Internet-Draft Randy Haagens

draft-haagens-iscsirqmts-~~xx~~01.txt Hewlett-Packard Co.

Expires 07 Jan 2001 design group draft ver 0.4, 07 July 2000

Changes from ver 0.3 are highlighted. These changes reflect discussion in the design group teleconference 6 Jul 00.

iSCSI (Internet SCSI) Requirements

~~This work defines a mapping of SCSI to TCP/IP, known as iSCSI.~~

Scope. We propose to define a mapping of SCSI protocol to TCP/IP so that SCSI storage controllers (principally disk and tape arrays and libraries) can be attached to IP networks, notably Gigabit Ethernet (GbE) and 10 Gigabit Ethernet (10 GbE).

Motivation. We seek timely adoption of a protocol mapping for block storage over IP networks. Accordingly, we have chosen to work with the existing SCSI architecture and commands and also the existing TCP/IP transport layer. Both these protocols are widely-deployed and well-understood. Using them means a minimum of new invention, the most rapid possible adoption, and the greatest compatibility with Internet architecture, protocols, and equipment.

The iSCSI protocol is a mapping of SCSI to TCP, and constitutes a "SCSI transport" as defined by the SCSI SAM-2 document [SAM2, p. 3, "Transport Protocols"].

1 Applicability

Traditionally, volume/block-oriented storage controllers (e.g., disk array controllers, tape library controllers) have supported the SCSI-3 protocol, and have been attached to computers through the SCSI parallel bus or through Fibre Channel. File-oriented storage controllers have supported the NFS and/or CIFS protocols, and have been attached directly to IP networks such as Ethernet.

The IP/Ethernet infrastructure offers compelling advantages for volume/block-oriented storage attachment compared to current approaches:

- Increasing performance and reduced cost driven by Internet economics and “IP convergence”
- Seamless conversion from local to wide area using IP routers
- Emerging availability of “IP datatone” service from carriers, in preference to ATM or SONET or T-1, T-3 services
- Protocols and middleware for management, security and QoS
- Economies arising from the need to install and operate only single type of network

The following applications for iSCSI are contemplated:

- Local storage access, consolidation, clustering and pooling (as in the data center)
- Remote disk access (as for a storage utility)
- Local and remote synchronous and asynchronous mirroring between storage controllers
- Local and remote backup and restore
- Evolution with SCSI to support of emerging object-oriented storage model

And the following connection topologies are contemplated:

- Point-to-point direct connections
- Dedicated storage LAN, consisting of one or more LAN segments
- Shared LAN, carrying a mix of traditional LAN traffic plus storage traffic
- LAN-to-WAN extension using IP routers or carrier-provided “IP Datatone”
- Private networks and the public Internet

The iSCSI standard will permit SCSI volume/block-oriented devices to be attached directly to IP networks such as Ethernet. The SCSI-3 command ~~protocol~~ sets (defined by the ANSI NCITS T10 committee) will be mapped to TCP. iSCSI is this mapping, and is analogous to (but not the same as) SCSI-FCP (aka "FCP"), which is the mapping of SCSI to Fibre Channel.

Local-area storage networks will be built using Ethernet LAN switches. These networks may be dedicated to storage, or shared with traditional Ethernet uses, as determined by cost, performance, administration, and security considerations. In the local area, TCP's adaptive retransmission timers will provide for automatic and rapid error detection and recovery, compared to alternative technologies.

IP LAN-WAN routers will be used to extend the IP storage network to the wide area, permitting remote disk access (as for a storage utility), synchronous and asynchronous remote mirroring, and remote backup and restore (as for tape vaulting). In the WAN, TCP end-to-end will avoid the need for specialized equipment for protocol conversion, ensure data reliability, cope with network congestion, and automatically adapt retransmission strategies to WAN delays.

The full realization of iSCSI will involve the following elements: (1) Completion of Requirements (this document) and Specification documents; (2) Development of Ethernet storage NICs and related driver and protocol software[1]; (3) Development of compatible storage controllers; and (4) The likely development of translating gateways to provide connectivity between the Ethernet storage network and the Fibre Channel and/or parallel-bus SCSI domains.

Products will initially be offered for Gigabit Ethernet attachment, with rapid migration to 10 GbE. For performance competitive with alternative SCSI transports, it will be necessary to implement the performance path of the full protocol stack in hardware. These new storage NICs will perform full-stack processing of a complete SCSI task, analogous to today's SCSI and Fibre Channel HBAs. They typically also will support all host protocols that use TCP, including NFS, CIFS and HTTP.

A key goal is not to require modifications to the current IP and Ethernet infrastructure to support storage traffic over TCP. Nevertheless, the performance and security requirements of storage will create opportunities for improvement in security protocols and QoS implementations. The addition of storage traffic to local- and wide-area internets (and even to the public Internet) may introduce increased requirements for traffic monitoring and engineering in those environments.

It is contemplated that many organizations initially will choose to operate storage networks based on iSCSI that are independent of (isolated from) their current data networks except for secure routing of storage management traffic. These organizations will benefit from the high performance/cost of IP equipment and a unified management architecture, compared to alternative means of building storage networks. As security and QoS evolve, it will become more reasonable to build combined networks with shared infrastructure; nevertheless, it is likely that sophisticated users will choose to keep their storage subnetworks isolated, for the best control of security and QoS.

The proposed charter of the IETF IP SCSI Working Group (IPSWG) describes the broad goal of mapping SCSI to IP. Within that broad charter, many transport alternatives may be considered. Our initial work focuses on TCP, and this Requirements document is restricted to that domain of interest. At the current time, we do not seek a more generic requirements statement that would justify the choice of TCP (or

[1] iSCSI will be implementable using current SCSI application layer software, current TCP/IP network protocols stacks, and current NICs, with the addition of an iSCSI layer between SCSI and TCP. However, high-performance applications are expected to require hardware acceleration.

another protocol) as transport, since the merits of using TCP are readily evident to the working group participants.

2 Definitions

Certain definitions are offered here, with references to the original document where applicable, in order to clarify the discussion of requirements. Throughout the text, use of defined terms is emphasized by producing them in bold face type. Definitions without references are the work of the authors and reviewers of this document.

Logical Unit (LU): A **target**-resident entity that implements a device model and executes SCSI commands sent by an application client [SAM-2, §3.1.50, p. 7].

Logical Unit Number (LUN): A 64-bit identifier for a logical unit [SAM-2, §3.1.52, p. 7].

SCSI Device: A device that is connected to a service delivery subsystem and supports an SCSI application protocol [SAM-2, §3.1.78, p. 9].

Service Delivery Port (SDP): A device-resident interface used by the application client, device server, or task manager to enter and retrieve requests and responses from the service delivery subsystem. Synonymous with **port** (SAM-2 §3.1.61) [SAM-2, §3.1.89, p. 9].

Target: An SCSI device that receives SCSI command and directs such commands to one or more logical units for execution [SAM-2 §3.1.97, p. 10].

Task: An object within the logical unit representing the work associated with a command or a group of linked commands [SAM-2, §3.1.98, p. 10].

Transaction: A cooperative interaction between two objects, involving the exchange of information or the execution of some service by one object on behalf of the other [SAM-2, §3.1.109, p. 10]. [A **transaction** seems to be a smaller unit than a **task**.]

3 Requirements

In the attached, actual requirements statements are flagged with [R]. Related discussion is flagged with [D].

The requirements are somewhat arbitrarily grouped into categories. This is for convenience only. No semantic meaning is to be implied from the category names.

3.1 General

[R] Support block storage IO over IP networks.

> [D] Our initial approach uses SCSI for the block storage protocol, and TCP/IP for the network transport.

[R] Minimize optional features; but when allowed, (1) Allow for option negotiation at session establishment (login); (2) Provide for signaling an error (reject) when an unsupported feature is requested.

3.2 Performance/Cost[2]

In general, iSCSI must allow implementations to equal or improve on the current state of the art for SCSI interconnects.

[R] Low delay communication.

> [D] Conventional storage access is of a stop-and-wait or remote procedure call type. Applications typically employ very little pipelining of their storage accesses, and so storage access delay directly impacts performance. The delay imposed by current storage interconnects, including protocol processing, is generally in the range of 100 microseconds. The use of caching in storage

[2] Performance/Cost is frequently, but inaccurately, referred to as Cost/Performance. We prefer the Performance/Cost formulation, so that increasing Performance/Cost is goodness.

controllers means that many storage accesses complete almost instantly, and so the delay of the interconnect can have a high relative impact on overall performance.

[R] High bandwidth, bandwidth aggregation.

[D] The bandwidth (transfer rate, MB/sec) supported by storage controllers is rapidly increasing, due to several factors: (1) Increase in disk spindle and controller performance; (2) Use of ever-larger caches, and improved caching algorithms; (3) Increased scale of storage controllers (number of supported spindles, speed of interconnects). Not only must the iSCSI provide for full utilization of available link bandwidth, it also must exploit parallelism (multiple connections) at the device interfaces and within the interconnect fabric.

[R] Low CPU utilization, equal to or better than current technology.

[D] For competitive performance, the iSCSI protocol must allow three key implementation choices to be realized: (1) iSCSI must make it possible to build I/O adapters that handle an entire SCSI **task**, as alternative SCSI transport implementations do. (2) The protocol must permit "zero-copy" memory architectures, where the I/O adapter reads or writes host memory exactly once per disk transaction. (3) The protocol must not impose complex operations on the host software, which would increase host instruction path length relative to alternatives.

[R] Cost competitive with alternative storage network technologies.

3.3 SCSI

[R] Collaboration with ANSI NCITS T10 (SCSI)

[D] iSCSI is a new SCSI "transport" [SAM2]. Being the intersection of SCSI and TCP, iSCSI has potential impact on T10 as well as on IETF. However, a stated requirement (below) is that iSCSI shall have no impact on T10 architecture or command sets. Collaboration with T10 will be ~~required~~ necessary to achieve this requirement.

[D] Collaboration with T10 concerns three phases of T10 activity: (1) Past. For T10 work completed in the past, and well-document in T10 standards publication, we will seek assistance in properly interpreting those standards; (2) Present. For T10 work that is ongoing, or recently completed (but not widely published), we will seek review of our work by individuals active in T10, and/or the participation of those individuals in the IETF process; (3) Future. For compatibility with future T10 work, it is essential that iSCSI be a legitimate and recognized "SCSI transport", no less so than the several other SCSI transports. SCSI command standards must evolve within the context of all existing SCSI transports.

[D] Storage attachment to IP networks will engender an unprecedented potential for device sharing. This alone may impact future T10 work.

[R] Supported **SCSI Device** types. iSCSI shall support all SCSI device types. Our primary focus is on supporting "larger" devices: host computers and storage controllers (disk arrays, tape library controllers).

[D] Supported **SCSI Devices** will typically have adequate memory to implement the TCP transport and required iSCSI session state, and a cost structure that can support VLSI for full-stack protocol acceleration. Generally, a controller will be interposed between the iSCSI (typically Ethernet) connections and the drive interface (typically parallel SCSI or Fibre Channel). In the longer term, it will become feasible, due to the march of technology, to support iSCSI economically in disk spindle and tape mechanism controllers.

[R] Support SCSI SAM-2 architecture model.

[D] It would be helpful to produce a document discussing iSCSI with reference to SAM-2. No promises.

[R] Reliable Transport. The iSCSI mapping provides the SCSI-3 command layer with a reliable transport, equal to or greater in reliability than the parallel SCSI bus, and providing in-order delivery, as suggested by SAM-2.

[D] See [SAM-2, p. 17.] "The function of the service delivery subsystem is to transport an error-free copy of the request or response between the sender and the receiver…" [SAM-2, p. 22] "The manner in which ordering constraints are established is implementation-specific. An implementation may choose to delegate this responsibility...to the **service delivery port**. In some cases, in-order delivery may be an intrinsic property of the transport subsystem or a requirement established by the SCSI protocol standard. ¶For convenience, the SCSI architecture model assumes in-order delivery to be a property of the service delivery subsystem. This assumption is made to simplify the description of behavior and does not constitute a requirement.

[R] Support for SCSI Task Queuing.

[D] SAM-2 defines task queuing, and so strictly speaking, we don't need to call this out specifically. However, task queuing is not widely implemented today; and it will increase in importance with WAN IP networks, given speed-of-light delays. We are particularly interested in supporting task queuing of pipelined remote backup and asynchronous disk mirroring

[D] Just because iSCSI supports task queuing doesn't mean that the end SCSI node is required to do so also. Task queuing is an optional feature of SCSI.

[R] ~~Compatible with~~Supports all SCSI-3 command ~~protocols~~ sets [SPC-2, SBC, etc.]. There will be no requirement by T10 to modify the SCSI command documents. No modifications are required of the SCSI command layer implementation, except possibly to lengthen **task** timers to accommodate wide-area delays due to speed-of-light and switching.

[D] Note the restriction to SCSI-3 command ~~protocols~~sets. There are potential problems with gateways between iSCSI and SCSI-2 parallel bus devices. It may not be feasible to transport SCSI-2 commands over iSCSI. Gateways that wish to support older SCSI-2 devices may have to proxy for those devices, using SCSI-3 commands.

[R] Forward compatibility with future revisions of SCSI architecture and protocol. Attention to clean layering of protocols.

[D] This is a difficult requirement to achieve in practice, since we cannot predict how SCSI will evolve. However, careful attention to protocol layering principles will help ensure this result.

[R] Gateways to parallel SCSI ~~*[ref.]*~~[SPI-X] and to SCSI-FCP ~~*[ref.]*~~[FCP, FCP-2]. It will be possible to construct "translating" gateways so that iSCSI hosts can talk to SCSI-X devices; so that SCSI-X devices can talk to each other over a iSCSI network; and so that SCSI-X hosts can talk to iSCSI devices (where SCSI-X refers to parallel SCSI, SCSI-FCP, or SCSI over any other transport).

[D] This requirement is implied by support for SAM-2, but is worthy of emphasis.

[D] These are true application protocol gateways, and not just bridge/routers. The different standards have only the SCSI-3 command ~~protocol~~ set layer in common. These gateways are not mere packet forwarders. We need to look into their remote proxy behavior.

[D] Adequate liaison must be established with related standards bodies, principally ANSI T10 (SCSI).

3.4 iSCSI Session Layer

[R] SCSI command, data, and response transactions occur in a TCP ~~channel~~ connection that is determined by the initiator, in advance of starting the SCSI **task**.

[D] This requirement allows the initiator to assign the data transfer phase of a **task** to a given data transfer engine, at initiation of the **task**.

[R?] TCP ~~channel (~~connection~~)~~ allegiance. SCSI commands, data and status information for a given **task** shall flow within the same single TCP connection.

[D] This is a stronger statement than the one above, and is left here as a potential requirement, mostly so that it will be clear that the discussion topics below pertain to the notion of channel allegiance.

[D] SAM-2 seems to require this channel allegiance: "A **task** involving one initiator-target pair shall not specify a third **SCSI device** to participate in transmitting and receiving the remote procedure model elements for that **task**. Thus, an SMU initiator [e.g., a host computer] shall not create a **task** using one **service delivery port** with the expectation that the data transfer or status return for that task would occur via a different **service delivery port**" [SAM-2, § 4.10.7, p.33]. Of course, interpretation of this clause depends on the definition of **service delivery port**. If a **service delivery port** is a TCP connection, then channel allegiance is pretty clearly required. But if a **service delivery port** is an iSCSI session or an abstract **target** device, then the interpretation of this clause is less clear.

[D] We have found a number of other possible virtues in channel allegiance: (1) It supports multiple instances of the TCP protocol engine being controlled by a single iSCSI session layer; (2) Failure of a TCP connection will affect only a subset of the extant tasks (those that use the failed connection); (3) All TCP connections are used in exactly the same manner; (4) There is no need to have more than one IP port defined for the iSCSI protocol, which is firewall-friendly.

[R] Command striping (load balancing) across multiple host and device interfaces. It shall be possible to utilize multiple concurrent paths between hosts and devices for the purpose of load balancing.

[D] Load balancing refers to concurrent **tasks** from a single initiator. There is no ordering constraint among these **tasks**. We aim to distribute these **tasks** (commands and their related data and status) across multiple host ports, links, switch ports and device ports, in order to achieve aggregate performance equal to a multiple of single link performance.

[R] Command ordering for tape backup and asynchronous remote mirroring. It must be possible to pipeline commands to a device, and to have them executed in order by that device, as prescribed by SAM-2.

[D] Ordering can be maintained by allowing each command to complete before issuing the next. But that means there is no pipelining. For tape backup in the local area, this may be adequate, as the tape controller buffer can be made sufficiently large to cover the lower duty cycle of data transfers, and LAN speeds are fast enough to burst-fill the buffer. But in the wide area, a method of pipelining commands and responses is needed if the slower WAN link is to be filled continuously with data.

[D] This brings up an issue, if commands are sent in different TCP ~~channels~~connections. Although a single TCP ~~channel~~ connection delivers an ordered byte stream, there is no ordering constraint between TCP ~~channels~~connections. So command striping across TCP ~~channels~~ connections will result in the commands possibly being executed out of order, unless the commands themselves are numbered, and can be put back into order. SCSI does not provide a means for putting commands back in order, but requires that functionality of the "transport".

[D] We contemplate bonding multiple TCP ~~channels (~~connections~~)~~ into an iSCSI session for the purpose of ordered command striping. A command reference number (CRN) will allow iSCSI to receive commands in order from the initiator SCSI command layer, and deliver them in order to its peer command layer in the **target**. Note that this mechanism can be employed at all times, because delivering commands in order never hurts, even if the SCSI layer imposes no ordering constraints among them. This is the safest route, in fact, as it upholds the SAM-2 expectation of in-order delivery. We expect the ability to support a session consisting of multiple channels to be optional. ~~It will be possible for a target to refuse to add a channel to a session.~~

[R] Recovery at the session layer. The session layer specification shall explicitly address recovery at the session layer (from a failed TCP connection, for example).

[D] TCP will recover from data loss due to bit errors or congestion. But what if a TCP connection fails (hangs)? The specification needs to address this issue.

[D] Another case that we should consider is loss of session state at either the target or the initiator, for example, when a target is power cycled. Should it be possible to restore the session in this case, or will we have to report service delivery failure to the SCSI layer, for recovery at that level? In the case of a recovered session, we're concerned about "ghost IOs" that may inappropriately linger from a previous session.

3.5 Transport, Network and Link

[R] Works with existing installed Ethernet and IP WAN infrastructure. iSCSI should not require any modification to Ethernet hubs, switches or WAN routers to achieve minimum acceptable performance, QoS and security.

> [D] Using existing and off-the-shelf technology will allow iSCSI to fully leverage the cost, performance and rapid improvement of widely-deployed IP LAN and WAN technologies. Therefore, iSCSI cannot require the installation of special, non-standard features in the underlying technology. However, it may be desirable to apply certain optimizations that will enhance storage protocol performance, or the performance of other protocols in the presence of the storage protocol.

[R] Joint operation (coexistence) with other IP protocols. iSCSI shall not preclude concurrent operation with any of the protocols in the IP protocol suite, and shall be a good Internet citizen.

> [D] Many organizations will choose to operate iSCSI storage networks as separate networks from their traditional data networks, by a router only for management traffic. This approach delivers the most manageable environment from a performance and security perspective, and is analogous to today's separate Fibre Channel storage networks, except for the obvious benefits that derive from using LAN technologies. On the other hand, some organizations will favor using fewer networks, and mixing storage with other types of traffic. This practice will be more prevalent in the wide-area, where dedicated storage links exact a high price. For these reasons, graceful co-existence is required. Over time, improved support for the QoS and security features inherent in IP and Ethernet protocols will make it more and more reasonable to combine storage with other types of network traffic.

> [D] When storage is transported over the wider Internet, it must be done in a way that respects TCP's bandwidth management and congestion avoidance algorithms. This is one of the reasons for selecting TCP as the transport. We feel that TCP itself is a good Internet citizen, and our best chance for compatibility.

[R] Uses TCP/IP. iSCSI is a protocol mapping from SCSI to TCP.

> [D] While we don't preclude consideration of alternative transports, we have focused our attention on TCP. Given wide-area functions in a storage controller, and the resulting need for TCP support, inclusion of an alternative local-area transport may imply an increment of cost, not a cost savings; and it certainly represents an increment of complexity.

[R] Link Independent. iSCSI is defined for all IP networks, and is link-independent. All IP-compatible LAN and WAN links are supported. Specifically, there are no dependencies on Ethernet.

> [D] We may nevertheless want to benefit from certain link capabilities like Ethernet port aggregation and PPP multi-link. But the spec should not depend on these capabilities for its viability.

[R] LAN, MAN and WAN –capable. **SCSI Device**s that implement iSCSI will be capable of communicating with similarly-equipped devices and host computers over any IP network, whether local, metropolitan, or wide-area in scale.

> [D] iSCSI is used not only for local area disk block access and tape operations. It also is used for remote disk access (as for a storage utility), remote disk mirroring, and remote backup and restore (as for tape vaulting). Using TCP in the iSCSI end nodes means that the protocol is scalable from the local to the wide area.

[R] Handles high bandwidth × delay fabrics.

[D] This requirement must be clarified further, as an extension of the WAN requirement. Consider that the TCP pipe at 10 Gbps × 200 msec holds 250 MB[3]. Will TCP sequence counts be up to this, or will they wrap too frequently?

[R] Recovery of data stream processing immediately after TCP segment drop.

[D] In a conventional TCP implementation, loss of a TCP segment means that stream processing must stop until that segment is recovered, which takes a network round trip to accomplish. Following the example above, we would be obliged to catch 250 MB of data into an anonymous buffer before we could resume stream processing; later, this data would need to be moved to its proper location. We seek some means of putting data directly where it belongs, and avoiding extra data movement in the case of segment drop.

[D] Two possibilities are known at this time: (1) A Remote DMA feature added to TCP headers (in the options field) would allow the data portion of subsequent TCP segments to be placed directly, even though the iSCSI protocol headers have not been parsed; (2) A means of recovering iSCSI framing is the TCP stream would allow iSCSI protocol processing to continue, and the data to be put in its proper location.

[R?] Framing. Some method of framing iSCSI protocol units within the TCP stream ~~must be defined~~may be required.

[D] We are unresolved as to whether this is a requirement. The more basic requirement, described above, is to be able to recover the processing of the data stream immediately after a segment drop. Framing is one way to recover processing.

[D] The conventional way to ~~do this~~locate higher-level protocol headers in the TCP stream is simply by parsing from the beginning of the stream, and never making a mistake. Is this sufficient? Or, should we use some other means such as byte stuffing or use of the push bit? Related, how do we ensure that data actually is transmitted, and doesn't languish in a TCP buffer somewhere?

[D] As an example of the problem: suppose a TCP segment is lost due to congestion, and it happens to contain an iSCSI header. At that point, stream synchronization will be lost, as we cannot find the next iSCSI header. Following the example above, we're obliged to catch 250 MB of data before we can resume iSCSI operation. If we could find the next iSCSI header, we could implement an optimization (non-traditional for TCP implementations) that would require us only to catch a single iSCSI message's-worth of data. Subsequent iSCSI messages could be decoded, and the data put where it belongs (even though command ordering constraints would preclude acting upon the data until the missing SCSI command is received and inspected for ordering constraints).

[D] Several methods have been discussed for providing framing by TCP: (1) A flag could be added in the TCP options that indicates that this segment begins a next-level Protocol Data Unit (PDU); (1a) Method 1 could be combined with a remote DMA mechanism for TCP; (2) The TCP transmitter function could be modified so that it emits a TCP segment for every next-level PDU, effectively turning TCP into a reliable, sequenced, datagram protocol. Protocols such as iSCSI would then need to limit their PDUs to less than the maximum TCP segment size (which is dictated by link considerations), if IP fragmentation is to be avoided.

[D] Other methods could work above TCP. (1) Byte stuffing is an old technique for framing within byte streams; its main disadvantage is that every byte must be processed by the framing mechanism, which would make software implementation impractical; (2) A special marker header could be placed periodically in the TCP stream. These headers would be found by doing

[3] Assume land-based communication with a spot half way around the world at the equator. Ignore additional distance due to cable routing. Ignore repeater and switching delays; consider only speed-of-light delay of 5 µsec / km. The circumference of the globe at the equator is approx. 40 000 km (we need to consider round-trip delay to keep the pipe full). 10 Gb/sec × 40 000 km × 5 µsec / km × B / 8b = 250 MB.

arithmetic on TCP sequence numbers. They contain information about the exact location of iSCSI PDUs.

~~[R] It has been noted that a remote DMA option for TCP possibly could provide the desired framing.~~

[R?] Error detection. Stronger CRC.

[D] The TCP checksum is rather weak as error detection goes. It is supported by the link layer check codes (CRC-32 for Ethernet). Is that sufficient? We don't have strong protection from re-assembly errors. Routers modify the frame and recompute the CRC. Even switches recompute CRCs ~~(for VLAN shifting)~~when adding VLAN tags, although good implementations do the CRC recomputation incrementally[4]. The TCP checksum is our only end-to-end protection. If the TCP checksum is not sufficient, do we introduce some kind of check on the SCSI data buffers by the iSCSI layer? Possibilities: byte count, CRC. Whatever we do, it must be possible to compute these check codes on the fly, as data is transferred from NIC to memory, without making a second pass over the data once it is in memory.

[D] We are considering using the IPsec messsage digest function for this purpose. It's already defined, and it could be used as a check code (only) using well-known keys; hence, without introducing the key distribution problem. Using IPsec in conjunction with TCP would not require a modification to TCP. A concern about using the IPsec message digest function is that it may be more difficult to compute at high speed than a simpler CRC.

[D] But is TCP truly an end-to-end protocol? The notion of an end-to-end error check is that it and the data it protects pass through the network unchanged, but possibly subject to errors while on a link or in a memory. At the receiving end node, checking the CRC verifies the correct receipt of data. In some cases, such as the use of a SOCKS proxy server or perhaps a NAT, the connection is not end-to-end, but is the concatenation of two end-to-end connections. In these cases, the iSCSI PDU (message) may be a better candidate for CRC protection.

[D] When considering a CRC at the iSCSI layer, we will give consideration to separate CRCs for iSCSI headers and data, and to the need to intersperse CRCs within long data messages.

[R] Selective TCP retransmission.

[D] Given the long delays in the WAN, using TCP selective retransmission must be supported by iSCSI, in order to minimize the bandwidth impact of retransmission.

[R] Firewall friendly. The protocol's use of IP addressing and TCP port numbers should be firewall friendly.

[D] This probably means that all connection requests should be addressed a specific, well-known TCP port. That way, firewalls can filter based on source and destination IP addresses, and destination (**target**) port number. The source (initiator) port number also should be well-known for the initial TCP connection. Additional TCP connections would require different source port numbers (for uniqueness), but could be opened after a security dialogue on the control channel.

[R] Possible to move data directly from end-to-end, without having retransmission buffers in the middle.

[D] This is an important implementation detail. In an iSCSI system, each of the end nodes (for example host computer and storage controller) has ample memory; but the intervening nodes (NIC, switches) do not. We contemplate a WAN-scale retransmission requirementof 25 MB (1 Gbps) or 250 MB (10 Gbps, see earlier footnote). Therefore, it must not be necessary for ~~thee~~ intervening nodes to buffer data.

---

[4] Incremental CRC recomputation considers only the changed bytes in the frame, and the consequent change required in the CRC. The CRC is not recomputed in its entirety by making a pass over all the data. Because incorrectly copied data will not figure in the incremental CRC recomputation, the resulting CRC remains a valid check for these transcription errors.

[R] Conservative in use of TCP and session-layer connections. The number required should not scale directly with the number of supported **LU**s.

> [D] TCP connection and iSCSI session state is fairly expensive in terms of memory consumed both on- and off-chip (we contemplate VLSI implementation). At a minimum, we seek to support only the number of connections required to achieve required bandwidth and delay characteristics between hosts and storage controllers.

[R] Compatible with both IPv4 and IPv6.

> [D] We need to add a literal format for IPv6 addresses in ~~our~~ target domain name~~s field in urls~~.

3.6 Naming

[R] Naming. Whenever possible, iSCSI shall support the naming architecture of SAM-2. Deviations and uncertainties will be made explicit, and comment/resolution invited.

> [D] It may be necessary to provide a unique naming scheme for SCSI **LU**s. Fibre Channel does so using WWNs. There's some indication that the T10 Security work will complicate this problem through **LUN** renumbering. The manner of determining a unique, worldwide, unchanging **LU** name must be determined. We will attempt to make use of SPC-2 provisions for **LU** Identifiers (Vital product data page 83h [SPC-2, p. 203]).
>
> [D] We need to resolve whether the notion of "**target**" is relevant to iSCSI. Does an iSCSI session connect to a **target**? Can it subsequently address multiple **targets** and **LU**s or just a bunch of **LU**s?
>
> [D] We need to provide an understanding of just what a **Service Delivery Port (SDP)** is in the iSCSI context. Is it an IP endpoint? A session endpoint? A virtual device (**target**) that a session can be connected to? SAM-2 seems to equate an SDP with a **target** address, "…the application clients in each initiator have the ability to discover that logical units in the SMU **target** are accessible via multiple **Target** Identifiers (**service delivery ports**)…" [SAM-2, pp. 12-13]

[R] URLs. It shall be possible to name SCSI devices and possibly **LU**s using a URL syntax. These names shall be global (uniform) and suitable for passing as handles between SCSI application clients.

[R] Domain names. The Domain Name Service (DNS) shall be used to resolve the <hostname> portion of the url to one, or multiple IP addresses. When a hostname resolves to multiple addresses, these addresses shall be equivalent for functional (possibly not performance) purposes.

> [D] This means that the addresses can be used interchangeably as long as we don't care about performance. For example, the same set of SCSI **target**s and/or **LU**s (tbd) must be accessible from each of these addresses.

[R] Deal with the complications of the new SCSI security architecture [99-245r8].

> [D] Pay attention to the proxy naming architecture defined by the new security model. In this new model, SCSI **Logical Unit Numbers** (**LUN**s) can be mapped in a manner that gives each host (more correctly, each AccessID) a unique **LU** map. Thus, a given **LU** within a **target** may be addressed by different **LUN**s.

[R] Support SCSI 3rd-party operations.

> [D] The key issue here relates to the naming architecture for SCSI **LU**s. We need to determine a method of passing a name or handle between parties

3.7 Security

[R] Authentication. At a minimum, iSCSI parties shall participate in a simple principals authentication protocol. This protocol shall involve a minimum of encryption and no special hardware for implementation.

[R] Bootstrapping. It shall be possible to negotiate higher levels of security than the minimum, technique to be defined.

[R] Data encryption. Data encryption shall be optional, but when implemented, shall be done in a manner prescribed by iSCSI, by reference to other standards.

[R] Compatible with IP protocol suite security protocols for the present and future.

> [D] We anticipate incorporating IPsec (host-to-host) and SSL/TSL (TCP connection) security into the iSCSI protocol by reference, and as options. Adherence to good layering will ensure (as much as possible) that future security developments at the IP and TCP layers can be utilized by iSCSI.

[R] Permits use of firewall for security screening.

> [D] It's important to allow a firewall to be used to offload authentication from the end node. This is a possible means of defending against Denial of Service (DoS) assaults, from a less-trusted area of the network. We assume that the firewall(s) have much greater processing power for dismissing bogus connection requests than do the end nodes.

3.8 Topology Discovery

[D] OK, we said we'd leave this for later. But why not open the discussion?

[R] iSCSI shall have no impact on the use of conventional IP network discovery techniques.

> [D] IP discovery techniques are well-evolved. Various network management platforms have ways of discovering IP addresses, such a mining router caches. We assume that these techniques will be used, and will find all of the IP end points that contain iSCSI nodes.

[R] iSCSI shall provide some means of determining that a discovered IP end point in fact is an iSCSI node.

> [D] This requirement is just a placeholder. Generally in IP discovery, there is some way of determining the type of the discovered device. Possibly this is due to the presence of the SNMP protocol and specific MIB variables. In this case, SNMP is the bootstrap protocol. Alternatively, one could probe various TCP port numbers to determine if there exists a higher-level protocol at each port (the port number would tell you which protocol). To be determined. But in any case, some means is needed to determine that an iSCSI entity is present at an IP end point.

[R] When a device supports multiple IP end points, some means of determining the IP connection topology is needed.

> [D] A device may support multiple end points, yet it may not be reasonable to bind any combination of the end points together into an iSCSI session. For example, a port controller (aka channel group) card may have four ports that can be bound together. The storage controller may support four of these port controllers, yet not allow the binding together into a session of TCP connections made on different port controllers.
>
> [D] A really simple solution to this problem would be to define a means of describing port topology, and provide for reading that description either from a MIB or directly from the iSCSI layer (with a command).

[R] SCSI protocol-dependent techniques shall be use for further discovery beyond the iSCSI layer.

> [D] Discovery is a complex process. But SCSI provides specific hooks for doing the work, and all we need to do is transport the commands associated with this process. Generally the SCSI discovery process involves using the Report **LUN**s command to determine which **LU**s are addressable at a given **service delivery port**. Subsequently, the true identity of each **LU** (ie, name) is discovered by reading Vital product data page 83h. By comparing **LU** IDs, the discovery process can find that a given **LU** is accessible through multiple paths.
>
> [D] We need only verify that this SCSI mechanism is sufficient. Hopefully, we will not need to augment SCSI at the iSCSI layer.

3.9 Management

[R] IP-based management protocols. It shall be possible (but not required) to use IP-based management protocols such as SNMP and RMI in conjunction with iSCSI. However, the present effort will not define the management architecture for iSCSI networks.

[R] SCSI management protocols. It shall be possible to use SCSI commands for management (eg, SCSI Enclosure Services, SES commands) to manage iSCSI devices.

3.10 Interoperability

[R] It must be possible for hosts and devices that implement only those features specified in the RFC to interoperate.

[R] Software implementation is possible using conventional TCP/IP protocol stack.

> [D] Although some low-performance products may contemplate an all-software implementation, we expect the majority of iSCSI products to employ hardware protocol acceleration. This requirement really is here to solve two problems (1) Proof of interoperability, by compatibility with extant TCP implementations; (2) Prototyping, where the iSCSI protocol is first implemented in software using these conventional stacks. These prototypes will likely become the early reference implementations.

4 References

[SAM-2] ANSI NCITS. Weber, Ralph O., editor. SCSI Architecture Model -2 (SAM-2). T10 Project 1157-D. rev 13, 22 Mar 2000.

[SPC-2] ANSI NCITS. Weber, Ralph O., editor. SCSI Primary Commands – 2 (SPC-2). T10 Project 1236-D. rev 18, 21 May 2000.

[CAM-3] ANSI NCITS. Dallas, William D., editor. Information Technology – Common Access Method – 3 (CAM-3)). X3T10 Project 990D. rev 3, 16 Mar 1998.

[99-245r8] Hafner, Jim. A Detailed Proposal for Access Controls. T10/99-245 revision 8, 26 Apr 2000.

[SPI-X] ANSI NCITS. SCSI Parallel Interface – X.

[FCP] ANSI NCITS. SCSI-3 Fibre Channel Protocol [ANSI X3.269:1996]

[FCP-2] ANSI NCITS. SCSI-3 Fibre Channel Protocol – 2 [T10/1144-D]