Internet Protocol storage area networks

by P. Sarkar
K. Voruganti
K. Meth
O. Biran
J. Satran

The sheer scale of the storage needs of most organizations makes block storage management an important system administration problem. Application servers, databases, and file systems rely on an efficient underlying block storage system. The storage area network paradigm is fast emerging as a desirable block storage solution, due to its performance, resource-sharing, and capacity-scaling benefits. This paper shows that the ubiquitous Internet Protocol (IP) network is technically well-suited to host a storage area network. The paper presents the storage protocol, management, and security building blocks that are necessary for making IP storage a reality. The paper then discusses performance issues that must be addressed in order to make IP storage area networks competitive with other storage area network technologies.

In the past, storage models assumed the presence of block storage attached to every host server. Block storage can be defined as raw storage volumes composed of and addressed in fixed-size extents called blocks. Block storage is the lowest form of logical storage and typically lies beneath file systems or databases that expose storage through a semantically richer interface. This paper does not deal with the topic of file storage systems and is restricted to the block storage model. A comparative analysis between file and block storage access models can be found in Reference 1.

The delivery of block storage relied primarily on the Small Computer System Interface (SCSI) command protocol.[2] The SCSI command protocol attained prominence in this field primarily because this protocol was the most clearly defined among its peers, leading to superior interoperability over a wide range of devices, from disks to tapes. In addition, the protocol used messaging primitives that provided modularity and layering for fast prototyping. Later advancements such as support for command queuing and overlapping commands led to superior performance.

The preferred transport for the SCSI command protocol in the server-attached storage model was *parallel* SCSI, where the storage devices were connected to the host server via a cable-based parallel bus. However, as the need for storage and servers grew, the limitations of this technology became obvious. First, contention for access to the parallel bus limits the number of storage devices that can be attached to each cable. Second, the physical characteristics of the cable also limit the distance of the storage devices from the host server. These limits imply that the addition of new storage devices might require the purchase of a host server for attaching the storage. Third, attaching storage to every host server means that the storage must be managed on a per-host-server basis, a costly implication for sites with a large number of host servers. Finally, the protocol

0018-8670/03/$5.00 © 2003 IBM

limitations of parallel SCSI do not allow for an easy sharing of storage between host servers. This means that a server cannot take advantage of unutilized storage resources on another server, leading to potentially lower storage resource utilization.

The lack of scalability of the host-server-attached storage model led to the evolution of the model of a *storage area network*. In this model, storage devices are assumed to be independent machines that provide storage services via a network to a multitude of host servers. The fundamental premise of a storage area network is the ability of host servers to share storage resources in the network, enabling a higher degree of utilization than that achieved in the host-server-attached storage model. The advent of networking infrastructure capable of gigabit speeds, as well as the development of transport protocols capable of sustaining such speeds, further facilitates the sharing of storage over the network. In addition, the distance limitation of the host-server-attached storage model is removed.

With storage being made available as a service over a network, security becomes a very important management consideration; it is important to defend the storage service against attacks. Another issue that becomes more pressing is the need to provide reliability and performance guarantees to the end-users of the storage service, particularly when this service competes for resources in a network of undetermined quality.

The focus of this paper is to show that the pervasive Internet Protocol (IP) networking technology is well suited for hosting storage area networks. In this regard, this paper addresses the following issues:

- The need for storage area networks and, in particular, IP storage area networks
- The necessary protocol, management, and security building blocks required for building an IP storage area network
- The performance challenges associated with IP storage area networks

The paper focuses on the key concepts underlying IP storage area networks. We begin with an overview of the concept of a storage area network and show why IP storage area networks are needed. The next section provides details about Ethernet, IP, and Transmission Control Protocol (TCP) to demonstrate their value as the basic building blocks of an IP storage area network. The following two sections provide insight into the storage management and security challenges of an IP storage area network, respectively. Details about standardization efforts are given next, after which we highlight some of the performance challenges in building an IP storage area network (SAN). Finally, we present our conclusions.

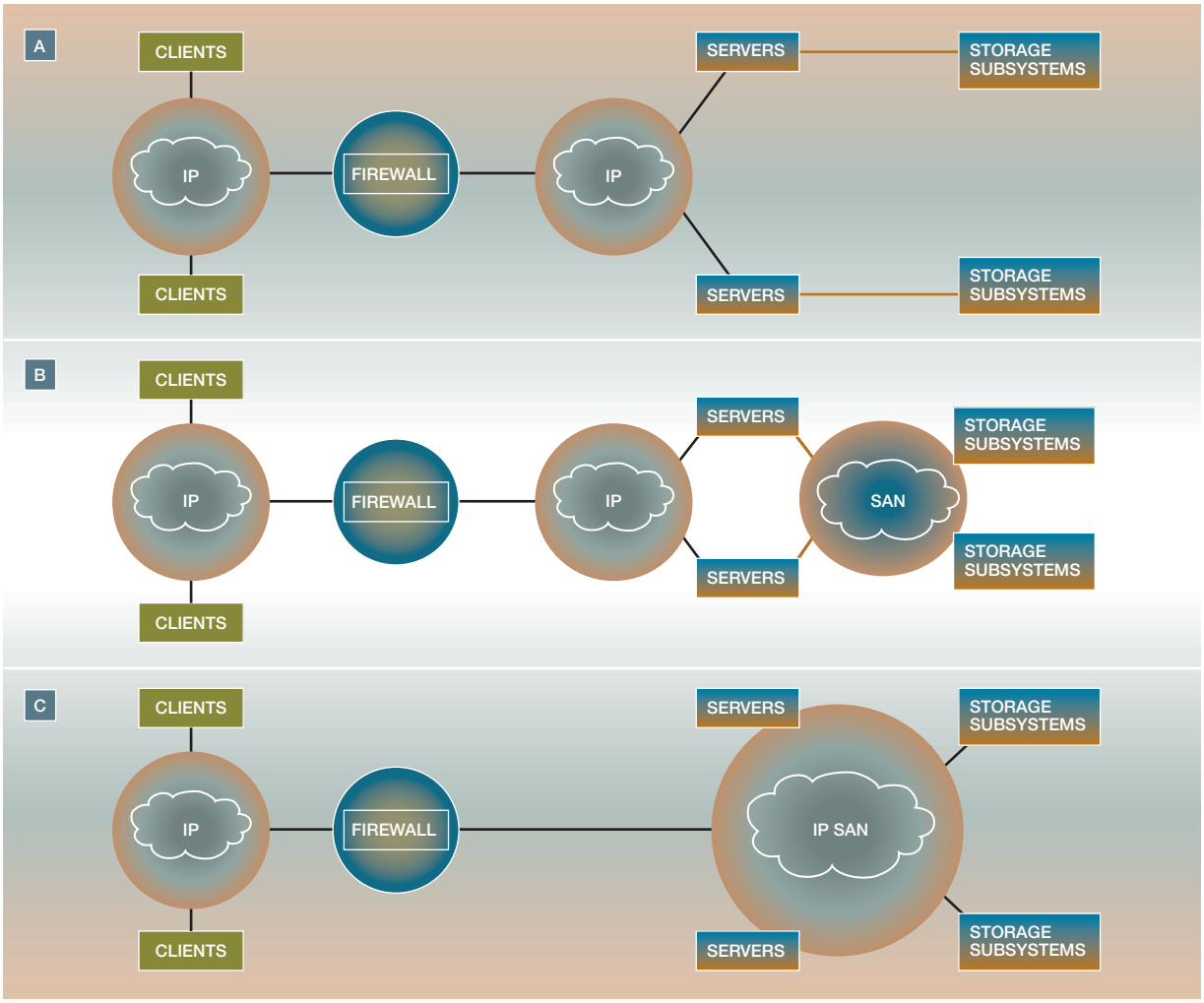
Storage area networks

This section first describes the basics of storage area networks and discusses their benefits in comparison to direct-attached storage systems. Storage area network protocols (such as SCSI) transport data and commands on top of a general-purpose network transport protocol. The requirements of the network transport layer of a storage area network are then described. Some existing non-IP-based storage area network transport protocols are analyzed. Finally, we discuss the need for IP storage area networks.

Definitions. A storage area network consists of a system of hardware and software components that interconnect host servers with storage systems. Figure 1A shows a direct-attached storage system. As shown in Figure 1B, a storage area network typically consists of multiple host servers and storage subsystems interconnected via a network. The storage subsystems can, in turn, consist of storage controllers and disk drives. The network can be either a network that is physically separate from the business's general-purpose network (as shown in Figure 1B), or it can be a separate logical network that shares the business's physical network infrastructure (as shown in Figure 1C). Note that in Figure 1C, the storage area network is physically the same as the general-purpose IP network.

Typically, the SCSI block storage protocol is used for communication between the hosts, storage controllers, and disks. SCSI uses a client/server model, where the hosts typically act as clients and are known as initiators, and the storage controllers or disks act as servers and are known as targets. The storage being managed by a storage controller is represented to the host as a number of contiguous storage areas called logical units (LUs). The logical unit number (LUN) identifies these contiguous storage areas. Thus, a SCSI initiator (host) sends a SCSI read or write command via a specific SCSI initiator port to a particular LU (identified by its LUN) residing on a particular SCSI target device (storage controller) via a particular SCSI target port.

Figure 1 Storage system models



Benefits. Storage area networks allow storage subsystem resources to be pooled and shared effectively among host servers. If storage is attached directly to host servers (as shown in Figure 1A), unused storage capacity in one host server is unavailable to another host server with a need for greater storage capacity. A storage area network solves this problem by making unused storage capacity available for use by any host server.

Another advantage of storage area networks is the separation of management and control of the storage subsystem from host server management. Since the storage subsystems are managed independently of the host servers, it is possible to add and remove storage capacity without causing significant host server down time. Furthermore, the number of available servers does not limit the aggregate storage capacity of an organization, allowing easier scaling of storage capacity.

Storage area networks also allow the separation of storage traffic from general network traffic. This is beneficial from a security, performance, and management standpoint. In direct-attached storage systems, backup operations usually involve moving data from disks attached to a particular server to disks attached to a different server, across the general-pur-

pose or local-area network (LAN). This backup traffic increases the load on the servers and the network, and it can potentially degrade the performance of other applications. In a storage area network, it is possible to perform LAN-free and server-free backup operations that copy data from a storage device directly to another storage device without transferring the data across the general-purpose network and the servers. In other words, data are sent across the dedicated storage area network directly between the source and destination storage devices. Having a separate storage area network also makes it easier to both secure and manage storage traffic, as there is no interference from the general network traffic.

Requirements. For a particular networking technology to be used successfully as a transport layer for storage area networking, the technology must provide:

- A high-bandwidth physical network interconnection
- A scalable networking infrastructure (in terms of distance and number of nodes)
- Reliable delivery of data in order
- An infrastructure to guard against various security threats
- Standardization of the storage transport delivery mechanism
- Network and storage management
- High end-to-end performance

Related technologies. Fibre Channel,[3] SSA[4] (Serial Storage Architecture), InfiniBand**,[5] VAXclusters,[6] and HIPPI[7] (High-Performance Parallel Interface) are some of the non-IP-based storage area network technologies that can be used to transfer SCSI command and data blocks. Of these storage area network technologies, Fibre Channel has emerged as the dominant choice, and constitutes the primary focus of this subsection.

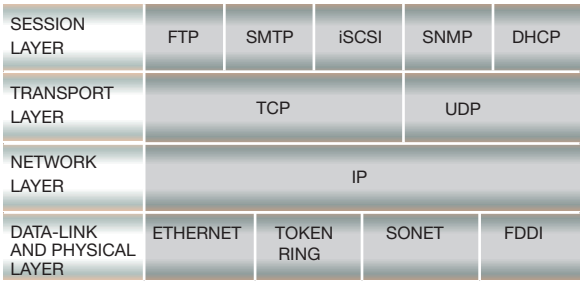
Fibre Channel networks provide a reliable, fast (2 Gbps), low-latency, and high-throughput transport mechanism for implementing the SCSI block storage protocol. The protocol for transferring SCSI blocks over Fibre Channel is known as FCP (Fibre Channel Protocol). Fibre Channel provides high end-to-end performance because it is a frame-based protocol, uses credit-based congestion control, and implements zero-copy send and receive (remote direct memory access [DMA]) semantics.[8] Credit-based congestion control ensures that frames are not dropped at switches during congestion. The zero-copy semantics ensure that the host CPU utilization in Fibre Channel environments remains low. Finally, the use of Fibre Channel frames reduces the memory requirements of Fibre Channel adapters for gigabit wire speeds.

Fibre Channel was not designed to be a wide-area network protocol, and is not scalable with respect to distance (it is limited to distances of 50 miles or less). Furthermore, Fibre Channel was designed to operate primarily in physically secure environments. Thus, its security infrastructure is not as well-defined as IP security mechanisms. The key drawback of Fibre Channel networks is that, in adopting them, an organization has to install a new and separate physical network infrastructure (wiring, switches, and adapters), and acquire a new set of network management skills, because Fibre Channel network hardware and management mechanisms are different from those used in IP network environments. Finally, to date, the interoperability record of Fibre Channel devices from different vendors has been a cause for concern.

The need for IP SANs. The notion of placing storage traffic on IP networks has been explored by numerous groups in the past.[9–13] In IP storage area networks, SCSI command blocks and data are encapsulated into TCP segments and transferred over TCP/IP/Ethernet networks. The choice of TCP/IP/Ethernet networks as the underlying transport mechanism for SCSI block storage is an attractive proposition for the following reasons:

- The emergence of Gigabit Ethernet and 10-Gigabit Ethernet allows one to utilize the commodity Ethernet layer for transferring high-bandwidth storage traffic in addition to general network traffic.
- IP networks have been shown to scale well with respect to distance, number of devices, and the amount of data.
- It is possible to leverage the elaborate security mechanisms that have been devised for IP networks.
- One can leverage the existing IP network management protocols and tools. Furthermore, one can also leverage the large pool of IP network management professionals.
- TCP is the most widely deployed reliable transport protocol that is supported by all of the major operating systems.
- Finally, it is also possible to leverage the existing general networking infrastructure (wiring,

Figure 2 Protocol stack for UDP and TCP



switches, network cards) for transferring storage traffic.

A detailed comparison of Fibre Channel, Infiniband, and IP SANs is provided in Reference 14.

The remainder of the paper provides details on the challenges associated with building IP storage area networks. Specifically, we discuss the issues associated with selecting the appropriate transport layer on top of IP, storage area network management, security, standardization efforts, and performance.

Ethernet and TCP/IP

This section describes the various properties of Ethernet and TCP/IP, and proceeds to justify the use of this technology for a scalable, high-bandwidth, and reliable storage area network.

Ethernet. A storage area network must provide high network bandwidths in order for storage to be made available as a service to applications residing on host servers. The need for bandwidth is important, because the goal is to provide performance competitive to that achieved with the server-attached storage model. In the IP world, Gigabit Ethernet and 10-Gigabit Ethernet can provide the necessary infrastructure for a high-bandwidth storage area network. Both the Gigabit Ethernet and 10-Gigabit Ethernet technologies have been widely adopted and Gigabit Ethernet is rapidly becoming the infrastructure of choice in many installations. This can provide the storage community with a cost-effective networking technology for storage area networks.

IP networks. The IP layer provides the network layer of the protocol stack. IP was designed to operate over a wide variety of physical transmission media varying in both speed and reliability such as Ethernet, Token Ring (IEEE 802.5), SONET (Synchronous Optical Network), FDDI (Fiber Distributed Data Interface), ATM (Asynchronous Transfer Mode) and even telephone lines. The IP protocol is itself connectionless and unreliable, lending itself to any type of networking infrastructure. An entire family of protocols has been developed to run on top of IP. Many of these IP-based protocols were developed and standardized through the IETF (Internet Engineering Task Force).[7]

The most prevalent protocols that run on top of IP are TCP and the User Datagram Protocol (UDP). Each of these protocols provides a particular type of end-to-end transport service that may be used by applications or by higher-level protocols. For example, File Transfer Protocol (FTP), the IP storage area network transport protocol (e.g., iSCSI [Internet SCSI]), and Simple Mail Transfer Protocol (SMTP) are built on top of TCP. Simple Network Monitoring Protocol (SNMP) and Dynamic Host Configuration Protocol (DHCP) are built on top of UDP. Each application that runs on top of TCP or UDP uses its own distinct application port for demultiplexing traffic between applications. The protocol stacks for UDP and TCP are shown in Figure 2.

TCP. A storage area network needs a reliable transport protocol to exchange control and data between the host servers and the storage devices. Fortunately, the IP networking community has invested a good deal of research into building a reliable in-order transport protocol called TCP. Years of deployment and experience have fine-tuned the behavior of TCP such that TCP traffic streams not only share available bandwidth responsibly but can also operate over a wide variety of network conditions. It must be noted that TCP is different from traditional storage transport protocols in that it is a streaming protocol, wherein application message boundaries are not recognized.

One of the key principles of TCP is that it is a connection-oriented protocol where the two endpoints of the network explicitly establish a connection and negotiate connection-specific parameters. Another important TCP property is reliable delivery, where every packet that is sent from an originating node to a destination must be acknowledged. If a packet is not acknowledged in due time as regulated by a timer, the originating node retransmits the packet to make sure that the destination node has received it. TCP further ensures in-order delivery of data to

the client, even if some data packets were delayed in the network and arrived out of order. A valuable attribute of TCP is its ability to be a responsible citizen in networks where bandwidth must be shared among multiple connections.

However, in contrast to the above-stated benefits, the weak checksum mechanism in TCP and the absence of built-in remote direct memory access (DMA) semantics pose challenges for the use of TCP as the transport protocol for storage area networks. The current TCP checksum data integrity detection mechanism may not provide the right level of protection in the presence of router-induced errors. Thus, the layers on top of the TCP layer need to utilize alternative data integrity strategies. The absence of built-in remote DMA semantics makes it difficult to avoid the TCP copy-and-checksum overhead as the data move from the network card to the end-user application space via an anonymous kernel buffer. This overhead, in turn, increases host CPU utilization. Efforts are currently underway to define remote DMA semantics for TCP networks.[8]

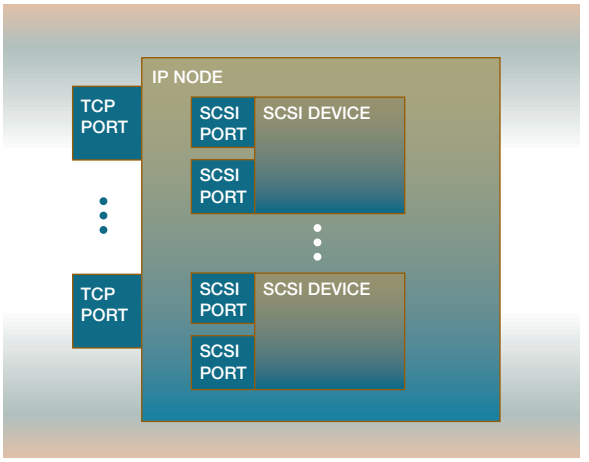
Storage area network management

Storage area network management consists of network and storage management components. Both network and storage management areas are, in isolation, well-understood fields. However, the combined area of storage area networking poses new challenges. This section briefly presents the management mechanisms that are available in the general networking and storage context and then analyzes naming, discovery, and monitoring and configuration issues within the context of IP storage area networks.

Background. In the general networking arena, the Domain Name Service (DNS) protocol[15] allows for the unique worldwide naming of IP network nodes. The Service Location Protocol (SLP)[16] allows for the discovery of resources on an IP network. The ICMP[17] (Internet Control Message Protocol), SNMP,[18] and SMI[19] (Structure of Management Information) standards allow for the monitoring and diagnosis of IP network nodes. Finally, DiffServ,[20] RSVP (Resource Reservation Protocol)/IntServ,[21] and MPLS[22] (Multi-Protocol Label Switching) allow for quality-of-service features in IP networks.

Traditional management of storage devices involves tasks such as configuring the RAID (redundant array of independent disks) levels and stripe size for the data to be stored, adding and managing storage capacity, configuring backup, managing restoring and mirroring schedules, monitoring the status and performance of storage devices, handling device errors and configuring storage volume parameters.

Figure 3 IP storage area network element



Naming. One of the key tasks of a storage network management infrastructure is to uniquely name and identify storage devices on the network.[23] In IP networks, each network endpoint is identified using an IP address. Furthermore, each endpoint can be given a unique domain name that resolves to the corresponding IP address via the DNS infrastructure. Transport layers such as TCP typically add a port identifier to identify transport endpoints. In the SCSI realm, a target may have multiple ports, each of which is identified using a SCSI port identifier. Occasionally, a unique vendor-generated SCSI name may be associated with the target. The task of storage network management is to give a unique storage name to each target that can be mapped to the target's IP address, application port (such as TCP port), SCSI port identifier, and optionally a SCSI device name. (The SCSI device name is mandatory in the iSCSI standard.) This allows any client of the storage service in the storage area network to uniquely locate an IP network device in the network via the IP address and TCP port, and to identify the correct target on the network device using the SCSI port identifier and the SCSI device name. Thus, as shown in Figure 3, the IP node address, TCP port identifier, SCSI device name, and the SCSI port identifier are necessary for identifying a SCSI device in an IP network.

Discovery. Discovery of devices is a very important management function in a storage area network.[23] In a non-networked environment, an initiator selects an enclosed channel (like a bus) and queries each storage device attached to the bus. Such a mechanism is inadequate in the networked world, due to the large number of devices that can be present, potentially, on the storage area network. Discovery mechanisms also need to adapt to the size of the storage network. For example, enterprise-wide discovery mechanisms do not scale down well to the level of a small-sized storage area network (and vice versa). The following types of discovery mechanism are available for discovery of storage devices on an IP network.

- *Static discovery:* In this mechanism, the initiator *a priori* knows the addresses of the targets it wants to access and configures those particular target addresses in its discovery-related registers. This discovery mechanism is similar to users typing in a known URL (uniform resource locator) in their Web browsers to access a particular Web site. This discovery mechanism is useful in small environments with few storage devices.
- *Multicast discovery:* As the size of the storage area network increases, it becomes difficult to manage the configuration statically. In multicast discovery, messages are multicast by clients or servers to other devices in the network to discover the appropriate services. The Service Location Protocol (SLP)[15] provides the appropriate registration and multicast mechanism to perform this type of discovery.
- *In-band discovery:* An in-band storage protocol discovery mechanism is useful in environments where other types of IP discovery services are not available. In this approach, once the initiator has *a priori* knowledge about an IP network entity, the in-band storage protocol discovery mechanism can be used to query whether any targets are present at the network entity.
- *SNMP discovery:* If the storage devices contain SNMP/MIB (Management Information Base) support, SNMP messages can be sent (unicast or multicast) to the storage devices to query whether the devices are initiators or targets.
- *Storage resource server discovery:* None of the above discovery mechanisms scale to the enterprise level. Static discovery requires too much manual effort, and multicast mechanisms do not scale well beyond a local area network. In another approach, the target storage devices register their services at the storage resource server[23] along with access control information as to which initiators can access them. Similarly, the initiator devices can query a storage resource server to determine which targets they can access. Storage resource servers are essentially directories that keep track of the state of the storage devices in the enterprise. They can be organized hierarchically to scale across the enterprise.

Monitoring and configuration. Finally, storage network management solutions need to allow for the monitoring and configuration of storage area network devices.[25] In IP networks, the SMI mechanism[19] is used to describe and name entities that need to be managed. The SNMP message[17] protocol is used to transfer SMI-defined objects between the management console and the managed entities. The SMI-defined objects are accessed via the virtual information store known as the MIB. There are MIBs associated with various entities such as network nodes, protocol ports, and connections. These MIBs are defined in standards maintained by the IETF standards body. The MIB framework has been extended to the SCSI domain so that there are SCSI-level MIBs associated with storage entities such as initiators, targets, and ports. In addition, it is also necessary to have MIBs associated with the storage transport protocol layer. Currently, a new management framework, an alternative to the SMI/SNMP combination, is emerging for managing storage resources as part of the SNIA CIM (Storage Networking Industry Association Common Information Model)/Bluefin initiative.[26]

Security

This section deals with security considerations in a storage area network. A storage area network has several components that interact with data as the data flow from the point of creation and access to the point of storage.[27] Each component has certain defined data privileges, and accesses that are not part of the privilege set are considered as attacks. An adversary can invoke an attack on the transport protocols for the storage area network, or on one of the storage subsystems.

Security scenarios. One of the simplest attacks in a network infrastructure employs eavesdropping, where packets are observed "on the fly." This is considered a passive attack, because no bits are changed or sent; this increases its threat, because in many cases the attack is never detected. At a higher level of security threat are active attacks where the attacker tries to impersonate a legitimate entity in the

storage area network, modify the content of packets that were sent by a legitimate entity, or resend such packets (a replay attack). The goal of a security framework in a storage area network is to prevent these types of attacks. The framework should be able to properly authenticate and authorize the different components of the storage area network (storage servers, storage clients, and communication endpoints) and grant access to data based on the security requirements of the data storage subsystem and the entities authorized to access it.

Security protocols. The IP networking infrastructure has support for advanced security protocols to protect the transmission of storage data securely over the network. The development of these security protocols is largely due to the ubiquity of the IP networking infrastructure, which makes it a target for security attacks. Transport Layer Security (TLS),[28] Kerberos,[29] and IPsec[30] are some of the available IP security mechanisms. IPsec has been identified as the most suitable security framework for storage over IP, mainly due to its superior performance characteristics. IP storage implementations are expected to operate on 1 and 10 Gbps Ethernet networks, and possibly at higher speeds in the future. The other IP security mechanisms are software-based and do not come close to the performance offered by IPsec, which includes hardware offloads that work on the IP packet level. The IPsec data authentication mechanism also provides data integrity and thus detects communication errors at the IP level. This leads to much simpler and lower-cost recovery through retransmission at the transport level, when compared to error detection at higher levels.

IPsec provides a secure channel between two communication endpoints of a storage protocol connection. Protection against passive attacks is accomplished by IPsec encryption of packets, using cryptographically strong data transformation algorithms such as 3DES (Triple Data Encryption Standard) and AES (Advanced Encryption Standard). Impersonation at the machine level is prevented by the IPsec key management protocol IKE[31] (Internet Key Exchange) that provides mutual authentication using techniques such as preshared keys or certificates. Attacks modifying packets or sending false packets on behalf of one side are detected using an IPsec authentication transform, which provides both sender authentication and data integrity by placing a message authentication code (MAC) in each packet. Attacks that resend a packet that was legitimately sent are detected by the IPsec antireplay mechanism, which adds a sequence number (protected by the MAC) to each packet.

A secure channel between the two communication endpoints is sufficient for gateway protocols such as iFCP (Internet Fibre Channel Protocol), where the data are going from one gateway to another. However, in direct host-to-storage protocols such as iSCSI, where multiple client entities or storage server entities can share a single communication endpoint, it is necessary to have additional authentication between the end client entity and the end storage server entity. This authentication is only required at connection establishment, assuming that the connection is protected by IPsec.

Security management. The security characteristics that a storage endpoint expects from another endpoint can be set *a priori* by storage management or obtained via a discovery service. A discovery service has to deal with many of the same security threats described for the storage security framework above.

Standardization

This section focuses on the standardization of the mechanisms by which storage is transported and managed in IP networks. The section describes all current standardization efforts and then provides a description of the iSCSI protocol.

Protocols. The IETF community has attempted to standardize the transport of SCSI over IP networks using various approaches: iSCSI,[10] FCIP (Fibre Channel over IP),[32] and iFCP.[33]

iSCSI is a protocol to transport SCSI commands over TCP. FCIP is used to connect islands of Fibre Channel storage area networks over IP networks to form a unified storage area network, as if they were in a single Fibre Channel fabric. iFCP is a gateway-to-gateway protocol for the implementation of Fibre Channel fabric functionality on a network in which TCP/IP switching and routing elements replace Fibre Channel components. Whereas FCIP and iFCP were invented to allow existing Fibre Channel protocols and infrastructure to work with IP networks, iSCSI is completely independent of Fibre Channel. Of the three approaches, the iSCSI protocol has seen the widest adoption among vendors, although it is still too early to make any definitive conclusions. In addition, the IETF is also standardizing the mechanisms by which nodes in an IP storage area network will be discovered, named, addressed, and managed. The secur-

ity infrastructure for iSCSI, FCIP, and iFCP is also part of the standardization process. The remainder of this section focuses on the details of iSCSI.

iSCSI. Although attempts have been made to define SCSI over UDP, SCSI over IP, and even SCSI directly over Ethernet, the designers of iSCSI decided that it was best to define SCSI over TCP. There are several reasons to use TCP as a transport (rather than some other reliable transport such as SCTP[34] (Stream Control Transmission Protocol):

- TCP is a reliable connection protocol that works over a variety of physical media and interconnect topologies.
- TCP is field-proven and scalable and offers an end-to-end connection model independent of the underlying network.
- TCP is probably going to be well-supported on underlying networks for some time in the future.

Because TCP detects undelivered packets and retransmits them, iSCSI packets that are sent over TCP and are lost during delivery are automatically resent by TCP. If iSCSI were defined on top of a protocol that is not reliable and in-order, then iSCSI would have to provide these services itself. Internet traffic must also adhere to the congestion control regulations of the IETF, and this is already provided by TCP. Although TCP has additional features that are not needed for transport of SCSI, the designers of iSCSI felt that the benefits of using an existing, well-tested and understood transport like TCP justified its use.

Since iSCSI is defined on top of TCP, it is possible to write an iSCSI device driver that uses a host's ordinary TCP/IP stack. However, due to the large volume of network traffic that is generated by iSCSI I/O, there may be a large CPU burden, due to the extra TCP processing that is required for iSCSI traffic. In many environments, it may be desirable to offload TCP (and possibly also iSCSI) processing onto a TCP Offload Engine (TOE) or an iSCSI adapter, thus reducing the CPU load on the host machine.

Sessions. iSCSI defines the notion of a "session" between an initiator and a target, corresponding to a SCSI I_T_NEXUS (Initiator–Target Nexus). An iSCSI session is composed of one or more TCP connections that are used to communicate between an initiator and a target. Multiple TCP connections in a session may be used to aggregate the bandwidth of these connections, possibly spanning multiple physical interconnects. Multiple TCP connections can also be used to provide redundancy and failover capabilities, whereby a second TCP connection (possibly on a different physical interconnect) is used to continue a session's iSCSI processing after a first TCP connection has failed. iSCSI requests are numbered sequentially, and the target must handle the requests in the order of their sequence numbers. If there are multiple connections within a session, requests may be sent over any of the TCP connections in a session. The target uses the sequence numbers to ensure that the requests are processed in their original order, even if they arrive on different TCP connections. Data and responses that are associated with a request must be sent over the same connection on which the corresponding request was sent. This simplifies the implementation of iSCSI operations when multiple connections are used, especially if the endpoints of the connection are on separate iSCSI adapters.

Login. The iSCSI protocol prescribes a log-in procedure that must be performed for each TCP connection between an iSCSI initiator and target. The purpose of the iSCSI log-in is to enable creation of a TCP connection for iSCSI use, authentication of the parties, negotiation of the session's parameters, and marking the connection as belonging to an iSCSI session. The initiator opens a TCP connection to a target and attempts to log in by sending a list of log-in parameters. The initial log-in parameters identify the initiator, the intended target, the level of the protocol being used, and the session to which the connection is to belong. The initiator and target may insist on authenticating each other with one of several authentication schemes, depending on the system's configuration and the administrator's setup. After passing the authentication stage, the initiator and target may negotiate operational parameters, such as the number of connections allowed in the session, the length of packets, how to work with the "request to transfer" mechanism, and so on, again depending on the system's configuration, capabilities, and resources. After completing the operational parameter negotiation, the initiator and target enter the full-feature phase in which SCSI commands and data may be passed between the initiator and target.

Error handling. iSCSI defines several levels of recovery to provide resilience in the face of a wide range of possible errors and failures. iSCSI error handling and recovery is expected to be a rare occurrence and may involve a significant amount of overhead. It is anticipated that most computing environments will

not need all of the levels of recovery that are defined in the iSCSI specification.

The most basic recovery class is "session failure recovery." All iSCSI-specification-compliant implementations must implement session failure recovery. Session failure recovery involves the closing of all of the session's TCP connections, aborting all outstanding SCSI commands for that session, terminating all such aborted SCSI commands with an appropriate SCSI service response at the initiator, and restarting a new set of TCP connections for the particular session. Implementations may perform session failure recovery in response to any iSCSI error.

A less drastic kind of recovery option is "digest failure recovery." As data packets are routed over a network, it is possible that some packets may become corrupted. TCP has a checksum facility to help detect such transmission errors. Although the probability of the TCP checksum failing to detect an error is quite small, this is not sufficient for some storage environments. Also, the TCP checksum does not provide protection for corruptions that occur while a message is in the memory of a router (when header information might be recalculated, and the data are no longer protected by a checksum). iSCSI therefore defines its own CRC (Cyclic Redundancy Code) checksum to ensure the end-to-end integrity of its packet headers and its data. Initiators and targets may negotiate whether or not to use this CRC checksum. If a CRC checksum error is detected on iSCSI data, the data packet must be discarded. Instead of performing session failure recovery, implementations may use the digest failure recovery mechanism to ask the connecting peer to resend only the missing data. Similarly, if a sequence reception timeout occurs, a similar mechanism can be used to ask the connecting peer to resend missing commands, responses, or other numbered packets that had been expected.

Performance

The performance challenges of building a storage area network over IP are not trivial. Critics of IP storage area networks point out that the overhead of using TCP is significant enough to result in poor latency for transaction-oriented benchmarks. It is also pointed out that common network application programming interfaces such as sockets do not allow for zero-copy transmits and receives of data, resulting in the overhead of multiple data copying.[35] Such data copying is considered harmful for overall throughput and will affect bulk-data scientific and video applications. Finally, data are transferred from the network adapter to the host machine using frame-size transfers. This means that every bulk data transfer may involve multiple interrupts instead of at most one interrupt, as is the case with specialized storage area networks. Consequently, the interrupt overhead can be the limiting factor in peak throughput if the storage device or host server CPU spends the majority of its cycles processing interrupts.

To address these concerns, this section presents a performance evaluation of a software implementation of the IP storage area network protocol stack. In this implementation, the IP storage area network protocol and the TCP/IP stack are resident on the host computer system. The goal of the evaluation is to point out the performance characteristics that meet the requirements of storage area networks and those that do not. More details about this evaluation can be found in Reference 36. The implementation aims to determine the latency and throughput characteristics of a host server connected to a storage device over a Gigabit Ethernet network. Although the implementation used the iSCSI protocol, we expect that the results are also applicable to other IP storage area network protocol stacks.

Experimental setup. The storage device used for the performance evaluation is a dual-733 MHz Pentium** III system with 128 MB of memory running iSCSI server software on top of Linux** version 2.4.2. The host server is an 800 MHz Pentium III system with 256 MB of memory and running iSCSI client software on top of Linux version 2.2.19. The two entities are connected via a Gigabit Ethernet connection over an Alteon** 180 switch, as shown in Figure 4. The Ethernet frame size used was the regular 1500 bytes, and no jumbo Ethernet frames were used. In addition, TCP/IP zero-copy optimizations were not used. Instead, the standard socket interface was used, which meant that the TCP copy-and-checksum routines were performed on both the host server and the storage device.

The test application resided on the host server and read raw SCSI blocks from a SCSI volume exported by the storage device. Since the aim was to isolate the efficiency of the transport, the application always read the same block, ensuring a cache hit. A cache miss would have involved the RAID subsystem of the storage device, and made it difficult to analyze the results. Write performance was not measured, because writes can be done using various means (im-

Figure 4 Testbed for performance evaluation of an iSCSI implementation

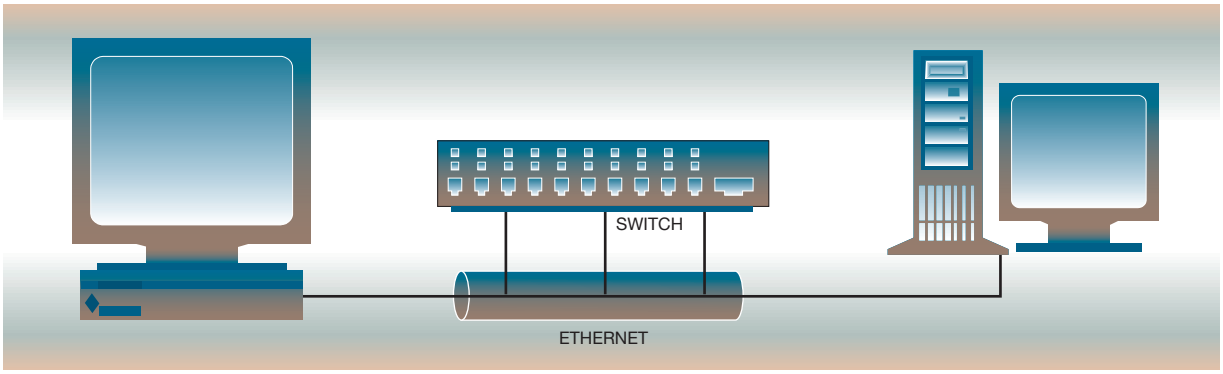
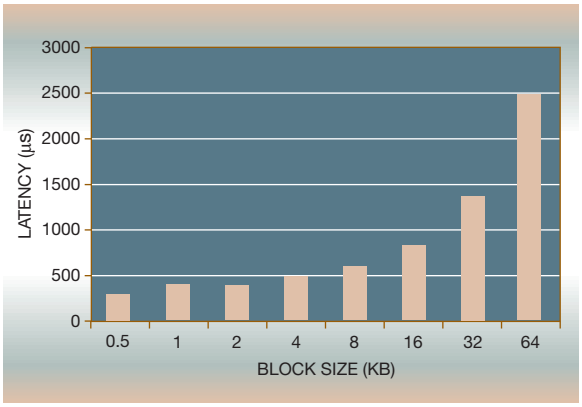


Figure 5 Latency measurements for the evaluation of a software iSCSI implementation



mediate, unsolicited, solicited), and add unneeded complexity to the analysis.

To measure latency, a single thread was used in the application to read raw SCSI blocks of various sizes from the storage device. For a particular block size, the same block was read 10000 times, and the average latency determined. To measure throughput, eight concurrent threads were used to read SCSI blocks of various sizes from the storage device. Eight threads were used because that is the concurrency limit imposed by the iSCSI client software in the host server. For a particular block size, each thread read a block 10000 times, and the throughput was calculated based on the time taken for all threads to finish reading the blocks. For the throughput experiment, the CPU utilizations of the host server and storage device were measured using the *vmstat* utility. The vmstat utility is a UNIX** system tool that reports statistics on processes, virtual memory, disk, trap, and CPU activity.

Results. The latency measurements (Figure 5) indicate a variation in average latency from 283 μs for a 512-byte block to a high of 2469 μs for a 64 KB block. The average latency values provide no meaning by themselves but are comparable (within 5 percent) to latency numbers obtained from the specification sheet of a Fibre Channel storage device for all block sizes.[37] There was an expectation that TCP/IP segmentation would have an adverse effect on latency for the larger block sizes, but it appears that the Gigabit Ethernet adapter does a reasonable job of interrupt coalescing and masks this effect. (Interrupt coalescing is a mechanism by which interrupt-generating events over a defined period of time are kept pending, and a single interrupt is generated for all of these events at the expiration of the time period.) This indicates that the TCP/IP fast path for transmits and receives does not impose a prohibitive overhead on latency. Consequently, it is not expected that IP storage (even in its software incarnation with no optimizations) will have an adverse effect on the performance of transaction-oriented applications and benchmarks.

However, the throughput measurements (Figure 6) indicate a different story. Although the average throughput from the storage device for the lower block sizes is similar to that obtained from a Fibre Channel storage device, the peak throughput is about 60 percent less than that obtainable from a Fibre Channel storage device. In these experiments, the

peak throughput is about 52 MBps for the 64 KB block size, and is constrained by the CPU of the host server, whose utilization is at 100 percent. A profiling of the CPU utilization of the host server indicated that its primary components were interrupt overhead (72 percent) and TCP copy-and-checksum (23 percent).

In addition, during the throughput experiments for the 64 KB block size, the CPU utilization of the storage device is at 51 percent, indicating that the storage device is capable of delivering additional throughput. In fact, by using multiple initiators, it was possible to obtain a throughput of 100 MBps at around 98 percent CPU utilization in the storage device. At this throughput, the constraining factor was the limit imposed by the network adapter. The CPU utilization figures were not available for the Fibre Channel storage device.

The CPU utilization of the host server is greater than that of the storage device because the host server is the receiver of bulk data. The receiving of data involves interrupting the host server every time a frame arrives and increases the interrupt overhead even if interrupt coalescing is used. This implies that if the experiments above involved writes, then the CPU utilization of the storage device would be higher.

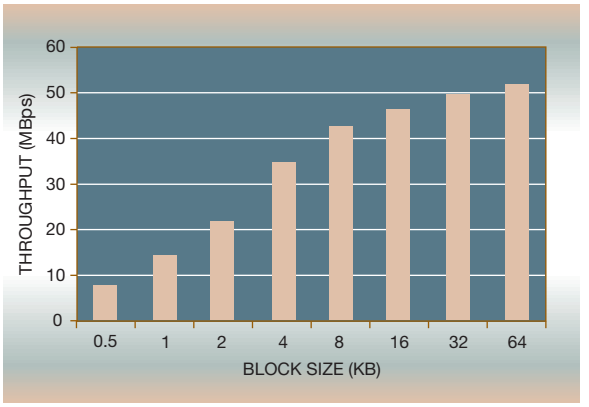
The results indicate that the main performance bottleneck in meeting the requirements of storage area networks is the high CPU utilization involved with bulk data transfers. The two main components of this high CPU utilization are:

- Interrupt overhead due to frame size transfers from the adapter to the host at high rates
- The overhead due to TCP copy-and-checksum in standard TCP/IP stacks for bulk data

Reducing CPU utilization. There are four potential avenues to reduce the high CPU utilization issues in IP storage subsystems. First, the interrupt overhead can be reduced by using 9 KB jumbo Ethernet frames, because this reduces the number of interrupts per bulk data transfer. For example, transferring a 32 KB data payload using the standard Ethernet frame may involve as many as 22 interrupts in the worst case, whereas in using the 9 KB jumbo Ethernet frame, only 4 interrupts may be involved. However, the jumbo Ethernet frames are not standardized and are not likely to be used in 10-Gigabit Ethernet.

Second, modified TCP/IP stacks with zero-copy transmit capability can be used to reduce the TCP copy-and-checksum overhead. The responsibility of generating the checksum is offloaded to the network adapter. However, zero-copy receives are not possible on such stacks because the network adapters are typically unaware of the final destination of any frame.

Figure 6 Throughput measurements for the evaluation of a software iSCSI implementation



Third, network adapters with TCP/IP offload engines have been released[38] where the entire TCP/IP stack is offloaded onto the network adapter. This also reduces the TCP copy-and-checksum overhead. However, zero-copy receives are not possible on such stacks because the TCP/IP stack is again typically unaware of the final destination of any TCP/IP packet. There is proposed work to add enough application hints to the TCP/IP header to make zero-copy receives possible.[39]

The fourth and most promising approach is the anticipated emergence of specialized adapters that have an iSCSI interface. This approach will reduce the interrupt overhead, because the iSCSI adapter will cause at most one interrupt per data transfer. In addition, offloading the protocol processing to the adapter will eliminate TCP/IP copy-and-checksum overhead. The disadvantage of this approach is that the use of such specialized adapters implies that commodity network adapters cannot be used in high-performance IP storage area networks. However, one can still use the existing switches and wiring present in commodity Ethernet networks in such cases. A detailed performance study evaluating these different approaches can be found in Reference 40.

Conclusions

Storage area networks are becoming an integral part of enterprise storage solutions because they provide resource sharing, storage capacity scaling, and performance benefits. With the emergence of Gigabit Ethernet technology, it is now possible to construct IP storage area networks which leverage an organization's existing IP infrastructure. We have presented the performance challenges that must be addressed in order for IP storage networks to be competitive with other storage area networking technologies. It is our expectation that the advantages of IP storage networks will motivate system designers, programmers, and standards bodies to address these challenges and greatly improve the performance of these networks.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Infiniband Trade Association, Intel Corporation, Linus Torvalds, Alteon, Inc., or The Open Group.

Cited references

1. R. Katz, "High-Performance Network and Channel Based Storage," *Proceedings of the IEEE* **90**, No. 8 (August 1992).
2. ANSI T-10 Working Group, *SCSI Primary Commands–2, ANSI NCITS.351:200x*.
3. A. Benner, *Fibre Channel: Gigabit Communications and I/O for Computer Networks*, McGraw-Hill Book Co., Inc., New York (1996).
4. I. D. Judd, P. J. Murfet, and M. J. Palmer, "Serial Storage Architecture," *IBM Journal of Research and Development* **40**, No. 6 (1996).
5. See http://www.infinibandta.org.
6. N. Kronenberg, H. Levy, and W. Stecker, "VAX-Clusters: A Loosely Coupled Distributed System," *ACM Transactions on Computer Systems* **4**, No. 2, 130–146 (1986).
7. American National Standard for Information Systems, *High-Performance Parallel Interface* (*HIPPI*), *X3T9.3/90-043* (1990).
8. S. Bailey, *The Architecture of Direct Data Placement (DDP) and Remote Direct Memory Access (RDMA) on Internet Protocols*, Internet Engineering Task Force (2002).
9. S. Hotz, R. Van Meter, and G. Finn, "Internet Protocols for Network Attached Peripherals," *Proceedings of the 6th IEEE/NASA Conference on Mass Storage Systems and Technologies* (March 1998).
10. J. Satran et al., *iSCSI, Internet Draft*, Internet Engineering Task Force (2002).
11. Wee Teck Ng, H. Sun, B. Hillyer, E. Shriver, E. Gabber, and B. Ozden, "Obtaining High Performance for Storage Outsourcing," *Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (2002).
12. G. Gibson, D. Nagle, K. Amiri, et al., "File Server Scaling with Network-Attached Secure Disks," *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems* (June 1997).
13. R. Van Meter, G. Finn, and S. Hotz, "VISA: Netstation's Virtual Internet SCSI Adapter," Eighth International Conference on Architectural Support for Programming Languages, San Jose, California (1998).
14. K. Voruganti and P. Sarkar, "An Analysis of Three Gigabit Networking Protocols for Storage Area Networks," *Proceedings of the 20th IEEE International Performance, Computing, and Communications Conference (IPCCC)* (April 2001).
15. P. Mockapetris et al., *Domain Names—Concepts and Facilities*, Internet Engineering Task Force RFC 1034 (1987); see http://www.ietf.org/rfc/rfc1034.txt?number=1034.
16. E. Guttman et al., *Service Location Protocol v2*, Internet Engineering Task Force RFC 2608 (1999); see http://www.ietf.org/rfc/rfc2608.txt?number=2608.
17. J. Postel, *Internet Control Message Protocol*, Internet Engineering Task Force RFC 792 (1981); see http://www.ietf.org/rfc/rfc792.txt?number=792.
18. J. Case et al., *Simple Network Management Protocol*, Internet Engineering Task Force RFC 1157 (1990); see http://www.ietf.org/rfc/rfc1157.txt?number=1157.
19. K. McCloghrie et al., *Structure of Management Information v2*, Internet Engineering Task Force RFC 2578 (1999); see http://www.ietf.org/rfc/rfc2578.txt?number=2578.
20. D. Grossman, *New Terminology and Clarifications for DiffServ*, Internet Engineering Task Force RFC 3260 (2002); see http://www.ietf.org/rfc/rfc3260.txt?number=3260.
21. J. Wroclawski, *The Use of RSVP with IETF Integrated Services*, Internet Engineering Task Force RFC 2210 (1997); see http://www.ietf.org/rfc/rfc2210.txt?number=2210.
22. E. Rosen et al., *Multiprotocol Label Switching Architecture*, Internet Engineering Task Force RFC 3031 (2001); see http://www.ietf.org/rfc/rfc3031.txt?number=3031.
23. M. Bakke, J. Hafner, J. Hufferd, K. Voruganti, and M. Krueger, *iSCSI Naming and Discovery, Internet Draft*, Internet Engineering Task Force (2002).
24. J. Tseng, K. Gibbons, et al., *Internet Storage Name Service, Internet Draft*, Internet Engineering Task Force (2002).
25. M. Bakke, J. Muchow, M. Krueger, and T. McSweeney, *Definitions of Managed Objects for iSCSI, Internet Draft*, Internet Engineering Task Force (2002).
26. Storage Networking Industry Association, *Bluefin Specification, Revision 1.0.0* (2002). See http://www.snia.org.
27. E. Riedel et al., "A Framework for Evaluating Storage System Security," *Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (2002).
28. T. Dierks and C. Allen, *The TLS Protocol*, Internet Engineering Task Force RFC 2246 (1999); see http://www.ietf.org/rfc/rfc2246.txt?number=2246.
29. J. Kohl et al., *The Kerberos Network Authentication Service (V5)*, Internet Engineering Task Force RFC 1510 (1993); see http://www.ietf.org/rfc/rfc1510.txt?number=1510.
30. S. Kent et al., *Security Architecture for the Internet Protocol*, Internet Engineering Task Force RFC 2401 (1998); see http://www.ietf.org/rfc/rfc2401.txt?number=2401.
31. D. Harkins et al., *The Internet Key Exchange*, Internet Engineering Task Force RFC 2409 (1998); see http://www.ietf.org/rfc/rfc2409.txt?number=2409.
32. M. Rajagopal et al., *Fibre Channel over TCP/IP, Internet Draft*, Internet Engineering Task Force (2002).
33. C. Monia et al., *iFCP—A Protocol for Internet Fibre Channel Storage Networking, Internet Draft*, Internet Engineering Task Force (2002).
34. R. Stewart et al., *Stream Control Transmission Protocol*, Internet Engineering Task Force RFC 2960 (1990); see http://www.ietf.org/rfc/rfc2960.txt?number=2960.
35. Hsiao Keng and J. Chu, "Zero-Copy TCP in Solaris," *Pro-*

ceedings of the USENIX 1996 Annual Technical Conference (January 1996).
36. P. Sarkar and K. Voruganti, "IP Storage: The Challenge Ahead," *Proceedings of the Nineteenth IEEE Symposium on Mass Storage Systems* (April 2002).
37. Mylex Corp., *White Paper on the Performance of the Mylex SanArray Pro FF2 Storage Controller*, Mylex Technical Report (2001).
38. See http://www.gigabit-ethernet.org/.
39. See http://www.ietf.org/proceedings/01dec/246.htm.
40. P. Sarkar, S. Uttamchandani, and K. Voruganti, "IP Storage: When Does Hardware Support Help," *Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (2003).

Accepted for publication January 23, 2003.

Prasenjit Sarkar *IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, California 95120 (psarkar@almaden.ibm.com).* Dr. Sarkar is a research staff member in the Storage Systems department at the Almaden Research Center. He received a B.Tech. degree in computer science and engineering from the Indian Institute of Technology (Kharagpur) in 1992, and M.S. and Ph.D. degrees in computer science from the University of Arizona in 1994 and 1998, respectively. He subsequently joined IBM at the Almaden Research Center, where he has worked on storage networking. In 2002 he received an IBM Outstanding Innovation Award for his work on iSCSI.

Kaladhar Voruganti *IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, California 95120 (kaladhar@us.ibm.com).* Dr. Voruganti is a research staff member in the Computer Science Storage Systems department at the Almaden Research Center. He received a B.S. degree in computer engineering and M.S. and Ph.D. degrees in computing science from the University of Alberta, in Edmonton, Canada. Dr. Voruganti was the lead for the iSCSI Naming and Discovery IETF standard, and he made key contributions to the industry's first iSCSI storage controller (IBM TotalStorage™ 200i product). He received an IBM Outstanding Technical Achievement Award for these efforts. Subsequently, he has worked in conjunction with Adaptec on the industry's first iSCSI target host bus adapter card. He was responsible for adding iSCSI functionality to Tivoli products. He has also worked on iSCSI protocol performance analysis. He is currently doing research in storage systems management and in adding iSCSI support to IBM storage products. Previously, he conducted research on developing cache consistency, recovery, and data transfer algorithms for client/server database management systems. He has published on these topics in major database conferences.

Kalman Meth *IBM Research Division, IBM Haifa Research Lab, Haifa University, Mount Carmel, Haifa 31905, Israel (meth@il.ibm.com).* Dr. Meth received his Ph.D. degree in mathematics from the Courant Institute, New York University, in 1988, and was a lecturer at Temple University from 1988 to 1990. He has been a technical staff member at IBM's Haifa Research Lab since 1990, working in the areas of operating systems, distributed and parallel computing, real-time systems, file systems, and multimedia. Dr. Meth currently manages the Networked Storage Technologies group at IBM's Haifa Research Lab and is one of the authors of the draft of the iSCSI protocol specification.

Ofer Biran *IBM Research Division, IBM Haifa Research Lab, Haifa University, Mount Carmel, Haifa 31905, Israel (biran@il.ibm.com).* Dr. Biran received the B.Sc. and D.Sc. degrees from the Technion-Israel Institute of Technology, faculty of computer science, in 1985 and 1991, respectively. He subsequently joined the IBM Haifa Research Lab, where he is a research staff member, working in the areas of network design, systems management, security, and remote objects technology. He participated in writing the security components of the iSCSI protocol specification.

Julian Satran *IBM Research Division, IBM Haifa Research Lab, Haifa University, Mount Carmel, Haifa 31905, Israel (julian_satran@il.ibm.com).* Mr. Satran received B.Sc. and M.Sc. degrees from the Polytechnic Institute of Bucharest in Romania, faculty of electronics, in 1962. He has been active in industry and academia in Romania and Israel. Mr. Satran joined IBM in 1987 and is now a Distinguished Engineer at the IBM Research Laboratory at Haifa, where he works on system architecture, networking, and storage. He received two Outstanding Technical Achievement Awards and an Outstanding Technical Innovation Award for his work on iSCSI. He is the main author of the iSCSI protocol specification. Mr. Satran is a senior member of the Institute of Electrical and Electronics Engineers and a member of the ACM.