

VISA: Netstation's Virtual Internet SCSI Adapter *

Rodney Van Meter[†] Gregory G. Finn and Steve Hotz
Information Sciences Institute
University of Southern California
Marina del Rey, CA 90292
{rdv,finn,hotz}@ISI.Edu

Abstract

In this paper we describe the implementation of VISA, our Virtual Internet SCSI Adapter. VISA was built to evaluate the performance impact on the host operating system of using IP to communicate with peripherals, especially storage devices. We have built and benchmarked file systems on VISA-attached emulated disk drives using UDP/IP. By using IP, we expect to take advantage of its scaling characteristics and support for heterogeneous media to build large, long-lived systems. Detailed file system and network CPU utilization and performance data indicate that it is possible for UDP/IP to reach more than 80% of SCSI's maximum throughput without the use of network coprocessors. We conclude that IP is a viable alternative to special-purpose storage network protocols, and presents numerous advantages.

1 Introduction

Storage system architectures are increasingly network-oriented, exploiting the ubiquity of networks to replace the direct host channel. Peripherals attached directly to networks are called network-attached peripherals (NAPs), or more specifically, network-attached storage devices (NASDs).

We have proposed that NAPs use the Internet protocol suite; in this paper we provide data on a sample implementation which supports the claim that the Internet Protocol (IP) can perform acceptably for NAPs.

In the experiments presented in this paper, we have used the User Datagram Protocol (UDP) as a transport protocol to send and receive SCSI commands and data to emulated network-attached disk drives. We have achieved data rates of 70+ megabits per second (Mbps) for read and write

through the file system over Myrinet before becoming CPU limited, and known optimizations could be expected to raise that to approximately 95 for write and 110 for read. TCP is predicted to be 10-25% slower than UDP. Fast ethernet is only 10% slower than Myrinet, with the difference caused primarily by ethernet's smaller MTU raising the amount of per-packet processing which must be done. Our analysis predicts that SCSI performance on the same hardware would become CPU-bound at 110 Mbps write, 133 Mbps read. Our conclusion is that more than 80% of maximum SCSI performance can be achieved using IP.

By demonstrating comparable performance, we open the door to the adoption of the Internet protocol suite by operating system vendors and disk drive manufacturers as an alternative to the numerous storage networks now being developed. This adoption would improve the sharability and scalability of storage systems with respect to numbers of network-attached devices and client systems, while substantially reducing legacy problems as technology advances, shortening development time and leveraging networking technology. In addition, TCP/IP enables such new wide-area uses as remote backup and mirroring of devices across the Internet.

This work was done in the course of the Netstation project, which concentrates on operating systems, network protocols, hardware mechanisms, and security and sharing models for network-attached peripherals. Some of the goals of the project are to demonstrate (1) that IP can provide acceptable performance in a host operating system when used to access peripherals, (2) that IP can be implemented efficiently inside network-attached peripherals, and (3) that our derived virtual device model enables efficient, secure use of NAPs. This paper addresses only the first point; the other two are presented in separate papers [HVF98, VHF96].

The paper begins with a brief description of Netstation, the network-as-backplane system architecture of which VISA forms a part. Section 3 describes the principles of networking as applied to network-attached peripherals, with special emphasis on the problems of scalability and host OS adaptation. We then describe related work, followed by the VISA architecture, performance, and possible performance improvements. Finally, we present our conclusions.

2 Netstation

Netstation is a heterogeneous distributed system composed of processor nodes and network-attached peripherals [Fin91, FM94]. The peripherals are attached to a shared 640 Mbps Myrinet network or to a 100 Mbps ethernet, as in Figure 1.

*This research was sponsored by the Defense Advanced Research Projects Agency under Contract No. DABT63-93-C-0062. Views and conclusions contained in this report are the authors' and should not be interpreted as representing the official opinion or policies, either expressed or implied, of ARPA, the U.S. Government, or any person or agency connected with them.

[†]Author's current address: Quantum Corp., Milpitas, CA, Rodney.VanMeter@qntm.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ASPLoS VIII 10/98 CA,USA
© 1998 ACM 1-58113-107-0/98/0010...\$5.00

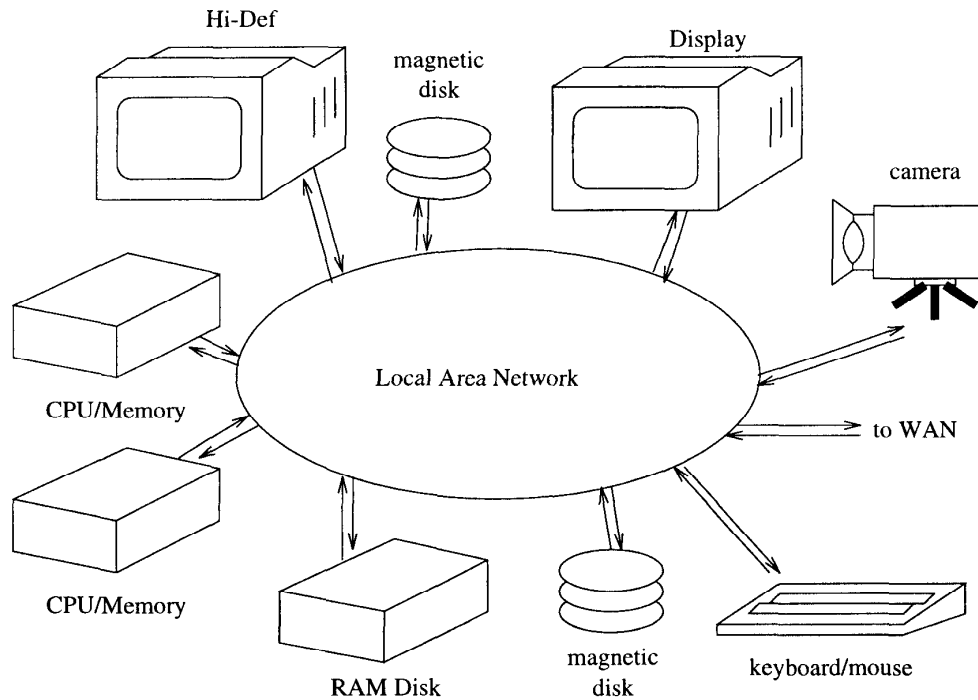


Figure 1: A Netstation network

The display and camera NAPs have been built; the other NAPs are currently emulated. The CPU nodes are Sparc 20/71 workstations running SunOS 4.1.3; the ideal Netstation CPU node would have only a CPU, memory and a network interface.

Connecting peripherals directly to the network allows sharing of resources and improves system configuration flexibility. Network clients can access peripherals without the intervention of a server.

Because the devices are attached to an open network with both trusted and untrusted nodes on the net, security at the NAPs is critical. We have developed a model we refer to as the *derived virtual device*, or DVD [VHF96]. DVDs provide a protected execution context at the device, allowing direct use of the devices by untrusted clients, such as user applications. The owner of a device *defines* the security policy, downloads a description to the NAP, and the NAP *enforces* the policy. This allows the owner to define a set of resources and operations allowed. Thus, a camera can be granted write access to only a specific region of a frame buffer, or a user application can be given read-only access to a DVD which represents a disk-based file or disk partition.

The only network-attached peripheral used during the experiments described in this paper is *IPdisk*, our emulated network-attached disk drive. *IPdisk* will be described in more detail in section 5.1.

VISA, our Virtual Internet SCSI Adapter, is the primary topic of this paper. It is the OS mechanism for supporting access to storage peripherals via the network.

3 Networking for NAPs

Netstation's shift to network-attached peripherals has a profound impact on the overall system architecture. In this section, we explore the technological and architectural motivations to make the change and its effect on host operating systems. We discuss the problems that must be solved to build large systems on heterogeneous networks, and show how choosing the TCP/IP suite solves these problems. Finally, we briefly discuss the appropriate high-level interface NAPs should present.

Developers of storage network technologies have started with different goals and assumptions about important issues such as number and type of devices and hosts to be interconnected, physical distance, cost and bandwidth. The result has been a proliferation of technologies developed primarily for NAPs, including 1394 (Firewire), Fibre Channel fabrics and Arbitrated Loop, HiPPI, and Serial Storage Architecture (SSA), as well as vendor-specific networks. Most of these have included development of complementary new physical, link, network and transport layers [Van96].

3.1 Motivation

Netstation uses network-attached peripherals to better share peripherals and take advantage of the relative technological trends of buses, networks, and peripheral processors.

The architectural reasons to shift from host adapter-attached devices to network-attached are better sharing of devices and reduction of the server's workload [RG96]. By allowing clients to directly access the devices, the server is no longer in the data path, reducing latency and demands on its buses, memory and processors. Devices can also communicate directly with each other without sending data across

a single shared system bus.

Buses do not scale well. They do not scale in distance; shortening a bus can raise data rates, forcing a direct trade-off in design. They do not scale in the number of devices interconnected, generally having a firm upper bound below twenty. They do not scale in aggregate bandwidth with the number of devices, as bandwidth is shared among the connected devices, and, due to increased capacitance, available bandwidth may actually decrease as devices are added.

Networks, especially serial optical networks, are improving rapidly in speed, typically scale well to large numbers of nodes, and can stretch over significant distances. Such networks are pushing into the gigabit per second range, a speed comparable to popular low-end buses such as PCI. Multicomputers and clustered systems have interconnected processors via networks for many years; I/O systems are now adopting networks to receive the same benefits.

3.2 Host I/O System and Networking Interaction

There are several characteristics which differentiate I/O for file systems from interactive application-level network I/O such as HTTP or telnet. Netstation uses some of these to improve the operating system efficiency, but further improvements are possible.

File transfers are generally large, page-aligned multiples of the system's page size (in our case, 4KB). On write, they are already pinned into memory and mapped into the kernel address space by the file system before they are handed to the bus or network subsystem. This may reduce the mapping and copying operations typically used to move data from user buffers to kernel buffers. On read, pages have already been selected, so the memory destination of incoming data is known in advance.

The file system cares only about the completion of the entire I/O operation. Data buffers will not be released to the application or virtual memory (VM) system until the read or write is finished. Thus, it is unnecessary to send partial completion status up the protocol stack until the entire transfer has either completed or failed. Typically, modern SCSI host bus adapters post only one interrupt to the host on completion or error, with requests that may be megabytes long. The host OS maintains a simple timer for the completion of the entire I/O. Sophisticated cards maintain one context per target device on the bus, and are capable of multiplexing among them. Current Fibre Channel interfaces are approaching a similarly autonomous level of operation, implementing the FC transport layer in the network interface card.

3.3 Scaling Problems Faced by I/O Networks

Netstation adopted the TCP/IP suite to leverage off the Internet community's experience with scale and heterogeneity. We believe that this experience makes the TCP/IP suite an increasingly appropriate choice and LAN-specific solutions increasingly inappropriate as more clients and servers are attached to more and larger heterogeneous storage networks.

The I/O network technologies deployed to date appear to offer only limited scalability, due to weakness in one or more areas. Media bridging, heterogeneity along many axes, security, latency tolerance, and congestion and flow control are several important areas that must be addressed.

Media bridging or routing becomes important as more hosts spread over more hops, and more legacy systems (both

hosts and nets) must be accommodated. In many environments, support for multiple networks and complex topologies, of the same or different types, is likely to be critical. This brings up issues of formatting, addressing, and especially underlying protocol interoperability.

3.4 Networking Protocols

In the Netstation project, we have chosen to work with UDP/IP and TCP/IP, reasoning that the more general architecture provides features that will be critical to network-attached peripherals in the long run. We expect that the performance shortcomings will be resolved by the networking research and development community.

Media-specific development of network and transport layers leaves systems with potential interoperability and legacy problems. While it is possible, for example, to route Fibre Channel frames across HIPPI networks, creating such exchange protocols for every possible pair of network technologies results in $O(N^2)$ protocols. If other networks cannot provide in-order delivery and support the flow control mechanisms Fibre Channel expects (either for class 2 or class 3 service), interoperability will be difficult. Additions of new clients or devices to the network are limited to network technologies which interoperate, regardless of external forces such as economic constraints, availability of new network technologies, etc.

Most developers and users of such networks (and devices for them) have discarded the possibility of adapting the TCP/IP protocol suite for communicating with NAPs¹. Reasons cited include inappropriateness for the type of traffic, TCP's wide-area tuning, the complexity of TCP, and especially performance of the entire TCP/IP suite in both bandwidth and latency [G⁺97]. We argue that these concerns are misplaced for several reasons.

Complexity is inherent as systems become larger. TCP/IP's complexity was not created arbitrarily, but developed in response to particular external stimuli, solving the problems presented above. Fibre Channel and other network transport protocols will have to face similar decisions as they attempt to address the same problems.

Earlier analyses of TCP/IP performance may have been based on poorly tuned or now outdated TCP/IP implementations. The Fibre Channel standardization effort, for example, was begun in 1988; much of the research on efficient networking implementations cited here has been conducted in the last decade. Older implementations often impose penalties such as extra data copies, separate checksumming passes, and inefficient demultiplexing of incoming data. Some of the improvements will be discussed in section 7.

3.5 Device Command Model

Once the architectural decision has been made to connect the peripheral to a network, the most important choice is the command request interface. Disk drives traditionally use a block-level interface, while network file servers use a file model appropriate to the needs of a particular set of clients. In the course of rearchitecting distributed storage systems, other choices are possible, such as an "object" model in which the disk is responsible for layout decisions but not file naming and security [G⁺96].

We have chosen a block interface, enhanced for security with derived virtual devices, rather than a file or file-like

¹Most of these net technologies can carry IP traffic for host-to-host communication, however.

model. This allows the simplest reuse of existing file system and operating system technology (data layout, partitioning, inode manipulation, the `sd` SCSI disk driver, `fsck` and `format`, virtual memory interaction, etc.). This also allows the broadest range of uses of the device, providing clients with the choice to build a Fast File System (FFS) or log-structured file systems, non-Unix file systems, network RAID, and other uses of raw partitions such as swap space, hierarchical storage management cache and databases.

4 Related Work

The projects most similar to Netstation are MIT's ViewStation [HAI⁺95] and Cambridge's Desk Area Network (DAN) [BHMP95]. Both use ATM networks as their device interconnect, and establish a physical boundary to the system for security purposes, while Netstation uses protocol-based security. They have defined a useful taxonomy of dumb, supervised and smart devices.

Network-attached storage is an area of much current research and development. Fibre Channel disk drives, new distributed file server architectures, and custom development of storage networks all play a part.

A TCP/IP RAID controller was developed at Lawrence Livermore National Labs; it is the first TCP disk device of which we are aware [WM95].

Mainframe channels have been extended to run over phone lines and even WANs for remote device mirroring; products from CNT, EMC and others perform such functions. However, they typically use media-specific protocols.

Fibre Channel-attached disk drives utilize a simple SCSI block interface with no security on a moderately complex network. As described above, this raises concerns about security, scalability, legacy systems and interoperability with other types of networks.

The CMU Parallel Data Lab's Network Attached Secure Disk (NASD) project divides NFS-like functionality between a file manager and the disk drives themselves, so that not only read and write commands but also attribute set and get are executed at the drive [G⁺96, RG96, G⁺97]. The file manager is responsible primarily for verifying credentials and establishing access tokens.

Soltis' Global File System (GFS) uses Fibre Channel disk drives modified to support a lock primitive [SRO96, Sol97]. This provides simple, efficient distributed locking. The drive itself attaches no meaning to the locks; by convention among the clients, they are used to lock inodes and other data structures.

The Petal distributed disk system provides a virtual disk model on which the Frangipani distributed file system is built [LT96, TML97]. Due to the virtualization of storage space, their model moves the actual backing store allocation to the disk, though the interface between Petal and Frangipani is a block-level one.

Various system vendors have developed their own networks on which distributed device sharing takes place. VAX-clusters [KLS86] and ServerNet [HG97] are two examples which use message passing between devices and hosts; the new SGI Origin series uses custom hardware to implement a shared address space on a switched network which includes many processors and I/O nodes [LL97].

5 VISA Architecture

VISA, Netstation's Virtual Internet SCSI Adapter, is an operating system module which makes Internet-attached pe-

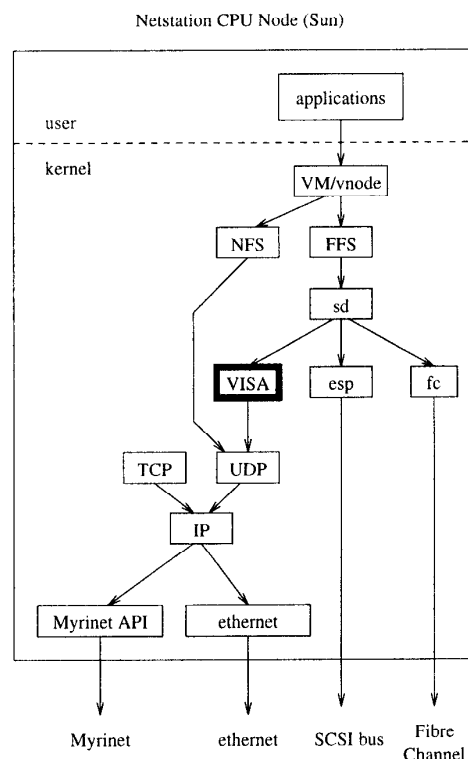


Figure 2: Netstation MPU node OS components. `sd` = SCSI disk device driver, `esp` = SCSI bus adapter driver, `VISA` = Virtual Internet SCSI Adapter, `VM` = virtual memory, `FFS` = Fast File System

ipherals appear as if they were attached to a local SCSI bus.

VISA implements an instantiation of the `scsi.transport` structure. SunOS uses a layered device driver model for SCSI devices. Device drivers which present the standard block or raw interfaces found in `/dev` are specific to a type of peripheral, such as `sd` for SCSI disks and `st` for SCSI tapes. For SCSI devices, these drivers in turn depend on lower-level services to communicate with the specific type of host adapter present. The `scsi.transport` structure consists primarily of pointers to nine high-level functions that implement a well-defined interface for sending commands to a SCSI device and managing the memory for returned data.

In figure 2, the `sd` SCSI disk driver is shown transmitting requests to VISA and to `esp`. `Esp` is the standard SCSI adapter type present on Sun SPARC workstations. Additional implementations exist for third-party SCSI adapters. It is at this layer that support for networked SCSI, such as Fibre Channel, is installed. Because we are using standard SCSI commands, no additional packaging or formatting, such as XDR, is required.

VISA transmits packets by calling UDP, which uses IP to send packets over any supported network medium. We have used both 100bT ethernet and Myrinet in these experiments.

We use UDP as the transport-layer protocol because we expected UDP to be faster than TCP as well as easier to work with inside the kernel. On top of UDP we found it necessary to build a simple reliability layer. While the network itself is highly reliable, SunOS provides only limited buffering in the sockets, and packets are discarded when the buffer is full. Our reliability layer works with a fixed-size window and assumes in-order delivery; on every 48KB transmitted, the sender pauses for an ACK. On receipt of out-of-order data, a NAK is sent and the sender rolls back. A timeout of one second is currently implemented only at the device (IPdisk). The host depends on either IPdisk to discover network glitches, or the higher-level I/O request to timeout and be restarted within the `sd` device driver. This is purely a convenience; a production-quality implementation would have timers at both ends.

We use 8 kilobyte data payloads, plus a small header including sequence numbers, on top of the UDP packet. Over Myrinet, this is a single packet. Over ethernet, this forces IP fragmentation in order to work within the 1500 byte MTU.

As is common with UDP, checksumming is turned off. The data integrity is still protected by the link layer checksum. Both hardware and software mechanisms for doing the TCP/UDP checksum with zero CPU cost are known. It can be done during data copy, DMA or transmission [PP93, FHV96].

We use one kernel pseudo-process per device attached to the VISA virtual bus. Much like NFS `biods`, these are responsible for the communication with the device, and are necessary because the SunOS kernel is not multithreaded.

5.1 IPdisk

IPdisk is our Netstation emulated disk drive. It runs as a user process on another Sun, emulating the SCSI block device command set running over UDP or TCP. It can be configured to use RAM to emulate disk storage, or to use a regular file, or access an actual raw SCSI disk. For the experiments described in this paper, IPdisk was used with a 32 MB RAM buffer, the fastest form of backing store, in order to stress the host operating system. In this mode, the

physical characteristics (rotational and seek latency, zone-variable transfer rates, etc.) of a disk are not emulated, so the host CPU becomes the bottleneck.

IPdisk supports our derived virtual device model. This allows the owner of the device to download small programs which act as filters on the SCSI RPCs before those RPCs are actually executed. This feature is primarily expected to be used for sharing devices among multiple hosts and executing third-party (direct device-to-device) copies. Because the purpose of these experiments is to saturate the host CPU, DVDs, which only affect execution time at the device, are not enabled for the experiments presented here.

6 Performance

We have run experiments to measure the performance of regular Berkeley fast file systems built on top of VISA-attached disks. The read and write throughput for sequential accesses are detailed in the following subsections, then in the next section we discuss potential improvements to this performance.

We measured the CPU utilization for file systems on VISA-attached disks and on a directly-attached fast, narrow (10 MB/sec.) SCSI bus. This provides a very direct comparison of the actual impact of different data transport mechanisms in a complete file system environment.

Our test configurations utilize a 75 MHz Sparc 20/71 with 64 MB of RAM and an 800 Mbps Sbus as the client. The CPU has 20KB/16KB I/D on-chip caches and 1MB of external cache. The STREAM benchmark reports memory copy bandwidth of 61.7 MB/s [McC98]. Identical 20/71s running IPdisk emulate the network-attached disk drives. Our absolute performance numbers are low by today's standards due to the age of the systems used, but the relative numbers and conclusions remain valid.

The biggest performance problem we encountered is excess calls to `bcopy`. On send, the Myrinet device driver copies the data from the `mbuf` chain to a special send buffer allocated and maintained by the device driver itself in main memory. From that buffer, the data is then DMAed to the buffer RAM on the network interface card, from where it is transmitted onto the network. The nominal reason for the copy is the expense and complexity of establishing DMA mappings for arbitrary memory addresses, as well as concerns about data alignment. However, when we are sending data from complete VM pages, as in VISA or NFS, the pages are already pinned in memory, properly aligned, and large enough to more than amortize the cost of creating the mapping, making it the proper choice.

The benchmark we are using is a modified version of Bonnie which reports additional CPU utilization and can loop on individual subtests to reduce the overhead of process creation. Bonnie, written by Tim Bray and available on the Internet, performs several tests to measure I/O operation overhead and throughput; we have concentrated here on throughput. 25MB files were written or read repeatedly until the total amount of data was one gigabyte.

6.1 Write Performance

Table 1 lists the measured and predicted performance of file writes on VISA-attached disk drives. Our write throughput is CPU-limited at 72 Mbps when using Myrinet, with kernel profiling disabled. Table 2 lists the measured write throughputs of other configurations and subsystems for comparison. Writing just to the file system buffer cache (tech-

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts



Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research



With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips



Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.