Comparison of network attached peripherals with a distributed file system and FireWire

Satyajit Phanse
886-34-3508
CSci555 Term Paper
University of Southern California

April 26, 2002

1 Abstract

How does a Network Attached peripheral (NAP) for storage compare with a distributed file system? This is one question that arises when looking at NAPs for the first time.

This paper looks at how NAP compares with a representative distributed file system [Network File System (NFS)] as well as IEEE's P1394, known as FireWire. Points of comparison include:

Architecture: The physical and protocol relation between the NAP device and the system.

Performance and scalability: Speed, efficiency, and limits on connections, devices, hosts, and so on.

Possible security implications: Forging network messages, impersonating clients or servers, tapping.

2 Introduction

The term "network attached peripheral" often refers to peripheral devices accessible over a room- or campus-wide network[1]. A special type of NAP are the network attached disk or network attached storage (NAS) which are to block-oriented data storage devices accessible over the network. The objective is to have a common physical data repository. The next thought is immediately of the Network File System (NFS), and one wonders how they compare. As specified in [1], a NAP provides a block-oriented protocol while the NFS is a complete filesystem. A comparison to contrast the two is essential.

A new method of attaching peripherals is the IEEE P1394, "FireWire". This paper also attempts to compare FireWire with the NAP paradigm.

3 Architecture

For the purpose of this paper, NAP architecture can be divided into broad layers, lowest first:

The physical layer defines the physical interconnect. Traditionally HiPPI (High Performance Parallel Interface), also includes ethernet and P1394.

The transport layer. [1] puts this at a higher level than the command protocols, but NAP commands should be encapsulated in the transport layer packets. Everything but the NAP commands and the physical interconnections fall into this layer.

The NAP command protocols. These would be the device block commands and encapsulated in the network packets. NFS lies here or just above this layer.

[1] does not divide the layers in this way.

3.1 Physical layer

The most common NAP attachment seems to be HiPPI, which runs at 100-200MBytes/s.[1] This is, however, expensive, and the cables have to be short.[1] Realistically, NAPs should be run over ethernet as ethernet is ubiquitous. Note that at least one network attached storage system using ethernet is commercially available.[7]

NFS runs over RPC and therefore can be run on ethernet interconnects. The only problem is that ethernet is currently limited to 100Mbits/s, although one hopes Gigabit ethernet will be common soon.

FireWire is a serial bus protocol and architecture developed and trademarked by Apple Computer Corporation. It is a suite of IEEE standards, numbered 1394. The serial interconnect has 4-pin cables. Unlike ethernet, this bus may or may not be powered. Some FireWire devices may require bus power. The cables are limited to 4.5 meters.[8]

HiPPI would be faster than 1394, since 1394 is a serial interface and HiPPI is a parallel interface tweaked for maximum performance. But HiPPI is more limited in distance than 1394.[8][1]

3.2 Transport layer

A NAP can use many transport protocols, including ethernet and ATM (Asynchronous Transfer Mode). Probably the most used is IPI-3 over HiPPI. SCSI bus protocol can also be used.[1] The IP suite is a good choice for ethernet-based NAPs[6].

NFS can use just about any network that supports RPC, since NFS is built over RPC[3]. This certainly includes most IP-based LANs.

The link layer of the FireWire protocol is a transport layer of sorts. It breaks the data of the upper layers into packets.[8]

3.3 Command protocols

The list of physical and transport layers used by NFS and NAP have considerable overlap, but the command protocols naturally differ. Most NAP protocols are based on some form of SCSI[1] and provide access to what looks like a "traditional local file system"[7].

NFS "provides transparent, remote access to filesystems"[3]. The NFS protocol itself includes commands for filesystem manipulation such as mounting, and reading/writing of directories and inodes. NFS also seems to include block-level commands[3], but these appear to be file-level blocks and not disk blocks.

FireWire's transaction and bus management layers make up the command protocol layer. Configuration and power management happen in the upper, bus management layer. The transaction layer supports the bulk of the request-response protocol.[8]

4 Performance issues

HiPPI is fast but limited in distance[1]. SCSI is also quite fast, and has the same limitation. TCP/IP is usually considered inefficient, but can be shown to be fast enough for NAP purposes. Besides, if NAP is to replace NFS, it must be able to work with commonly-used protocols, and IP runs on just about anything. It has been shown that IP code efficiency is not a bottleneck.[6]

NFS uses a considerable amount of caching to increase apparent performance. Similar tuning could bring NAP to a comparable performance, especially on an ethernet.

At 400MBytes/s, FireWire has the fastest raw speed but may become limited due to network topology.[8]

For any network, there is always a trade-off between speed and distance. A network based on current ethernet technology can run for several meters but at slow speeds. A network based on FireWire is limited to short cable lengths (4.5 meters) but can achieve high speeds.[8]

FireWire network topology is similar to that of SCSI. Many devices can connect to a hub, and devices can be daisy chained. Connections are point-to-point.[8]

FireWire nodes can be daisy chained and star-connected to the hub as well as to each other in a tree structure[8]. This reduces the collisions and provides a certain amount of "cable division" multiplexing.

Ethernet connections are not point-to-point. Ethernet segments can be separated by bridges or switched hubs to separate collision domains. This helps to increase performance. FireWire cannot really be split like this without designing a bridge controller which may be expensive. The controller would have to work much like a Network Address Translator or a router.

NAPs can also be striped[5] to reduce network locality and thus allow at least some data to be stored closer to the stations requesting the data.

Given a suitable naming scheme, there may be more than one NAP in the network. A FireWire tree is limited to 63 nodes[8]. If any more nodes are required, more trees must be added.

Multiple clients can mount filesystems from multiple NFS servers. One does not usually need to mount more than a few filesystems. NFS-mounted filesystems can also be cascaded.[3]

5 Security implications

As the NAP interconnect is based on a general network medium, significant security concerns are introduced.[1] These include wire-tapping and client or server impersonation.

Network attached storage devices can use a message digest, shared secret, and key architecture to encrypt and sign messages.[4] Using assymetric key algorithms, it is possible to be fairly certain of the identity of the device sending the message. The digests allow for making sure that the message was not damaged or changed in transit.

Sequence numbers, as used in TCP[11], can be used to prevent a malicious entity wreaking havoc by repeating a message. Encrypting the messages allows for protection against tapping.

In NFS, host-based authentication is based on IP address or hostnames.[10] These are quite easy to spoof, allowing unauthorised nodes to impersonate a server, or clients to impersonate other clients.[3] Methods to improve security include operating NFS over Kerberos. RPC also includes support for security parameters.[10] To a certain extent, NFS security depends on RPC security.

Since FireWire is meant to be a serial bus and the devices connected to it are somewhat "trusted", it does not explicitly implement security. This is similar to saying that an ethernet LAN does not explicitly implement security. FireWire is a packet and data transmission mechanism. Authentication and encryption are best done at higher levels.

6 Conclusion

The principal difference between a NAP storage device and NFS appears to be the location of control. NFS implements most filesystem operations. A NAP is expected to implement the disk read/write commands and leave file management to the other nodes on the network.[2]

A distributed file system (NFS), NAP, and FireWire can be regarded as different levels of one system. NFS provides the file system semantics, NAP provides the device, and FireWire provides the network.

NFS can be adapted to be used with NAP,[4] replacing the NFS server with a NAP device. Essentially this reduces the processor and OS costs. This device can be connected to a network of workstations over 1394. This increases the speed of the network. This network would have to be limited to the size of a small specialised lab.

One application is to try running a system like Condor over such a network. With the data stored on NAP and moved by this high-speed network, Condor should get good performance.[9]

Another application is as an MP3 server for a house. The client devices could be MP3 players installed in various rooms, much like an intercom system. The NAP server can be installed in a closet and serve requests made by the MP3 players.

A very quick comparison:

	NAP	NFS	FireWire
Purpose	Peripheral	Remote filesystem	High-speed interconnect
Provides	Device	Filesystem	Data, network, physical layer
Application	Shared storage	Shared access	Shared peripherals
Performance and Scalability	Multiple NAPs possible	Tens of simultaneous connections	63 nodes maximum
Security	Kerberos-based	Kerberos and RPC	(none)

References

[1] Rodney Doyle Van Meter III, “A Brief Survey of Current Work on Network Attached Peripherals”, ACM Operating Systems Review, Jan 1996, pp. 63.

[2] Garth A. Gibson, David F. Nagle, Khalil Amiri, Fay W. Chang, Eugene M. Feinberg, Howard Gobioff, Chen Lee, Berend Ozceri, Erik Riedel, David Rochberg, Jim Zelenka, “File server scaling with network-attached secure disks”, Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (Sigmetrics), Jun 1997.

[3] Russel Sandberg, David Goldberg, Steve Kleiman, Dan Walsh, Bob Lyon, “Design and Implementation of the Sun Network File System”, Proceedings of the USENIX Conference, USENIX, Jun 1985, pp. 119-130.

[4] Garth A. Gibson, David F. Nagle, Khalil Amiri, Fay W. Chang, Howard Gobioff, Erik Riedel, David Rochberg, Jim Zelenka, “Filesystems for

Network-Attached Secure Disks", Technical Report CMU-CS-97-118, Jul 1997.

[5] John H. Hartman, John K. Ousterhout, "The Zebra Striped Network File System", ACM Transactions on Computer Systems, Vol.13 No.3, 1995, pp. 274-310.

[6] Steve Hotz, Rodney Van Meter, Gregory Finn, "Internet Protocols for network-attached peripherals", Proc. Sixth NASA Goddard Conference on Mass Storage Systems and Technologies in conjunction with 15th IEEE Symposium on Mass Storage Systems, Mar 1998.

[7] Garth A. Gibson, Rodney Van Meter, "Network attached storage architecture", Communications of the ACM, Vol.43 No.11, Nov 2000, pp.37.

[8] Don Anderson, "FireWire System Architecture IEEE 1394", Mindshare, Inc., Addison-Wesley, 1998, ISBN 0-201-69470-0, LOC catalog "TK7895.B87A52 1998".

[9] M. Litzkow, M. Livny, M. Mutka, "Condor - A Hunter of Idle Workstations", Proceedings of the 8th International Conference on Distributed Computing System, IEEE, Jun 1988, pp. 104-111.

[10] M. Eisler, "NFS Version 2 and Version 3 Security Issues and the NFS Protocol's Use of RPCSEC_GSS and Kerberos V5", RFC2623, Jun 1999.

[11] Jon Postel (ed.), "Transmission Control Protocol", RFC793, Sep 1981.