Preliminary Measurement of the RMTP/RTIP*

Hui Zhang and Tom Fisher
hzhang, fisher@tenet.Berkeley.EDU

The Tenet Group
Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, California 94720
and
International Computer Science Institute
1947 Center Street
Berkeley, California 94704

Abstract. The Real-time Message Transport Protocol (RMTP) and the Real-Time Internetwork Protocol (RTIP) are the transport and network layer data delivery protocols in the Tenet Protocol Suite. We implemented the protocols in Ultrix on the DECstation 5000 workstations and in HP/UX on the HP 9000/7000 workstations. A preliminary measurement study has been conducted to evaluate the performance of the prototype implementation. Some of the results are: the throughput obtained by using RMTP/RTIP is comparable to that obtained by using raw IP; the rate control mechanism in RMTP/RTIP effectively enforces the traffic specification of communication clients; the scheduling mechanism in RTIP protects the real-time channel so that the performance of a real-time channel is not affected by the presence of IP traffic or other real-time channels in the network.

1 Introduction

There is an increasing demand to support real-time applications such as video conferencing, scientific visualization and medical imaging in an internetworking environment. These applications have stringent performance requirements in terms of delay, delay jitter, bandwidth and loss rate [Fer90]. The best-effort

* This research was supported by the National Science Foundation and the Defense Advanced Research Projects Agency (DARPA) under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives, by AT&T Bell Laboratories, Digital Equipment Corporation, Hitachi, Ltd., Hitachi America, Ltd., Pacific Bell, the University of California under a MICRO grant, and the International Computer Science Institute. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing official policies, either expressed or implied, of the U.S. Government or any of the sponsoring organizations.

service provided by the Internet Protocol [Pos81] is not adequate to support these applications.

The Tenet Group of University of California at Berkeley and International Computer Science Institute has proposed a new type of service called *guaranteed performance service*. By a guaranteed performance service, we mean that, before the communication starts, the client specifies its traffic characteristics and desired performance requirements; when the network accepts the client's request, the network guarantees that the specified performance requirements will be met provided that the client obeys the restrictions implied in its traffic description.

We believe that a reservation-oriented network architecture is needed to provide such a guaranteed performance service. The current Internet Protocol, which is connectionless and reservationless, cannot be used to support the service.

The Tenet Group has also designed and implemented a new protocol suite that provides guaranteed performance service [Fer92, FBZ92]. The protocol suite consists of five protocols: three data delivery protocols and two control protocols. The three data delivery protocols are: the Real-Time Internet Protocol (RTIP), the Real-time Message Transport Protocol (RMTP) [VZ91, ZVF92] and the Continuous Media Transport Protocol (CMTP) [WM91, MGWF92]. RTIP is the network layer protocol, while RMTP and CMTP are two transport layer protocols that provide message-oriented and stream-oriented transport services, respectively, on top of RTIP. The two control protocols are: the Real-Time Channel Administration Protocol (RCAP) [BM91], and the Real-Time Control Message Protocol (RTCMP). RCAP is responsible for establishment, tear-down and modification of the real-time channels, while RTCMP is responsible for control and management during data transfers.

In this paper, we will focus on the two data delivery protocols, RMTP and RTIP. Both protocols have been implemented on DECstation 5000 and HP9000 workstations. A preliminary measurement study has been conducted to evaluate the performance of the prototype implementation. Some of the results are: the throughput obtained by using RMTP/RTIP is comparable to that obtained by using raw IP; the rate control mechanism in RMTP/RTIP effectively enforces the traffic specification of communication clients; the scheduling mechanism in RTIP protects real-time connections so that the performance of a real-time connection is not affected by the presence of IP traffic or other real-time connections in the network.

The paper is organized as follows: Section 2 brief reviews the real-time channel scheme, on which the Tenet Protocol Suite are based; Section 3 describes the services, functions, software structure and programming interface for RTIP and RMTP; Section 4, which is the core of this paper, presents the simulation results; Section 5 briefly describes related work; Section 6 gives the conclusion and provides future work.

2 Background

The Tenet protocol suite implements the real-time channel scheme, which is a new communication abstraction proposed to support guaranteed performance service in a general packet-switched internetwork environment [FV90].

A real-time channel is a simplex unicast end-to-end connection with performance guarantees and traffic restrictions. Once established, it guarantees that the performance bounds requested by the communication client are satisfied so long as the client does not violate the traffic restrictions. The performance parameters a client can request are:

- delay bound D
- delay violation probability bound Z
- buffer overflow probability bound W
- delay jitter bound J

The traffic parameters a client needs to specify are:

- minimum packet inter-arrival time x_min
- average packet inter-arrival time x_ave
- averaging interval I
- maximum packet size s_max

A channel with both Z and W being 1 is called a *deterministic* channel; a channel with either Z or W being less than 1 is called a *statistical* channel [Fer90].

The following paradigm is proposed in [FV90] to provide guaranteed services to clients in a packet-switching network: before communication starts, the client specifies its traffic characteristics and performance requirements to the network; the client's traffic and performance parameters are translated into local parameters, and a set of connection admission control conditions are tested at each switch or gateway; the new channel is accepted only if its admission would not cause the performance guarantees made to other channels to be violated; during data transfers, each switch or gateway will service packets from different channels according to a service discipline; by ensuring that the local performance requirements are met at each switch or gateway, the end-to-end performance requirements can be satisfied.

Notice that there are two levels of control in this paradigm: at the connection level, the admission control policy allocates and reserves resources for each channel; at the packet level, the service discipline allocates resources according to the reservations made. In the Tenet protocol suite, RCAP is responsible for the connection level control, and RTIP is responsible for the packet level control.

3 RMTP and RTIP

RMTP and RTIP are the data delivery protocols in the Tenet protocol suite. Together with the Real-Time Channel Administration Protocol, or RCAP [BM91], they provide guaranteed performance communication services in an internetworking environment.

3.1 RMTP Services and Functions

The Real-time Message Transport Protocol, or RMTP, is the transport layer message-oriented data transfer protocol in the Tenet protocol suite. It provides a simplex, end-to-end, unreliable, in-order, and guaranteed performance message service. RMTP is responsible for such end-to-end functions as message fragmentation and reassembly, regulation of traffic according to traffic specifications and optional checksumming.

3.2 RTIP Services and Functions

The Real-Time Internet Protocol, or RTIP, is the network layer data transfer protocol in the Tenet protocol suite. Operating at each host and gateway participating in the real-time communication, RTIP performs rate control, jitter control, packet scheduling, and data transfer functions. It provides a host-to-host simplex, sequenced, unreliable, and guaranteed-performance packet service. All the data are transferred in packets on a simplex channel from the sending client to the receiving client. A packet is not guaranteed to be delivered; it may be dropped for two reasons: the packet may be corrupted during transmission, or there might not be enough buffer space in the case of a statistical channel. Packets that are not dropped are delivered to the receiving client in the same order as they were sent by the sending client; the relative positions of the dropped packets are indicated. The client data are not checksummed; they may get corrupted due to transmission errors. If the sending client sends packets neither larger than the maximum packet size negotiated during channel establishment time, nor faster than the specified maximum rate, all the packets delivered are guaranteed to meet the delay and/or delay jitter requirements with certain probability (the probability is 1 for a deterministic channel).

3.3 Relationship Between RMTP/RTIP and RCAP

All real-time data is carried on real-time channels. Channel state information is kept in each node along the channel's path. The RMTP state information is maintained at the end-points of the channel, while the RTIP state information is maintained at every node along the channel's path (including the end-points). RCAP is responsible for initializing the state information of a newly created channel and for destroying the state information of a channel when it is torn down.

3.4 Software Structure

RMTP and RTIP have been implemented in Ultrix on DECstation 5000 and in HP/UX on HP9000/7000 workstations. Both Ultrix and HP/UX are Unix-like operating systems; the networking software of both was derived from BSD Unix [LMKQ89].

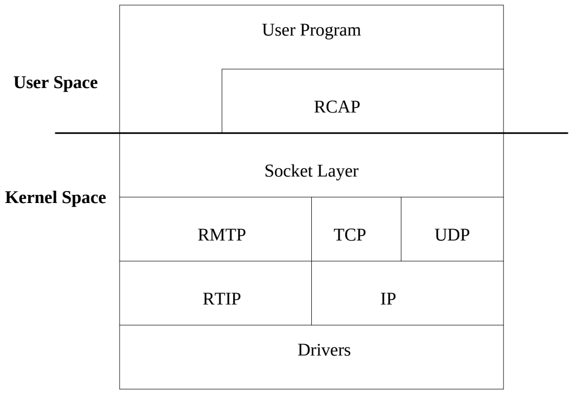


Fig.1. Software structure of RMTP/RTIP

The software structure of the RMTP and RTIP implementation is shown in figure 1. As can be seen in the figure, while RCAP is implemented in user space, RMTP/RTIP are implemented in the kernel, and co-exist there with TCP/IP. Through the socket layer software, RMTP and RTIP export interfaces to RCAP for control purposes and to user programs for data transfer purposes.

3.5 Programming Interface

RMTP and RTIP export programming interfaces to both user programs and RCAP agents. The following example illustrates how a user program sends data using the Tenet protocol suit.

```
DataSock = socket(AF_INET, SOCK_DGRAM, IPPROTO_RMTP)
RcapEstablishRequest(&ParametersBlock, &lcid, &Destination)
setsocketopt(DataSock, IPPROTO_RTIP, RTIP_ASSOC, &lcid, sizeof(u_short))
send(DataSock, msg, len, flags)
```

Fig.2. Programming interface for a user program

The user program first invokes the `socket` system call to open an RMTP socket; it then calls `RcapEstablishRequest` to establish a real-time channel, while `ParametersBlock` specifies the end-to-end traffic and performance parameters, `lcid` is used to hold a return value, and `Destination` specifies the destination IP address. `RcapEstablishRequest` is an RCAP library function; when invoked, it will contact RCAP agents at this node and other nodes on the path to reserve resources for the new request and establish a real-time channel. The agent on each node of the real-time channel, which resides in user space, will call

```
setsockopt(ControlSocket, IPPROTO_RTIP, RTIP_SPEC, &RtipSpec,
          sizeof(struct RtipSpec))
```

to set up the data structure in RTIP, where `ControlSocket` is a permanent socket opened for communication between the RCAP agent in user space and the RTIP module in kernel space, and `RtipSpec` has the local traffic, performance parameters, and amounts of resources to be reserved by the RTIP module.

If the channel is successfully established, `RcapEstablishRequest` will return a small integer `lcid`, which is the unique local channel identifier of the real-time channel at the source. The user program then calls `setsockopt` to associate the opened data socket with the established real-time channel. Finally, the user can start sending data to the data socket using the `send` system call.

A similar interface is provided on the receiving side. There are also other `setsockopt` calls that RCAP uses to communicate with RMTP and RTIP.

4 Measurement Experiments

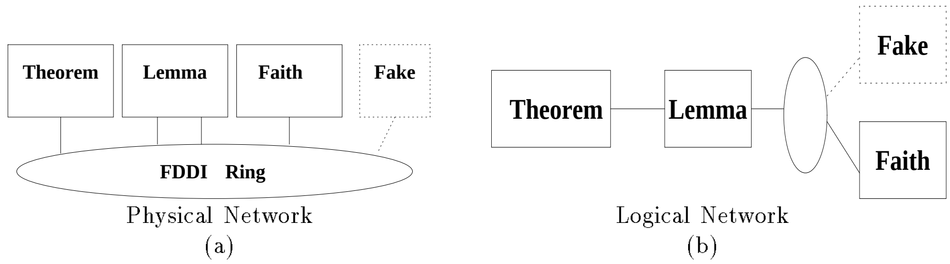


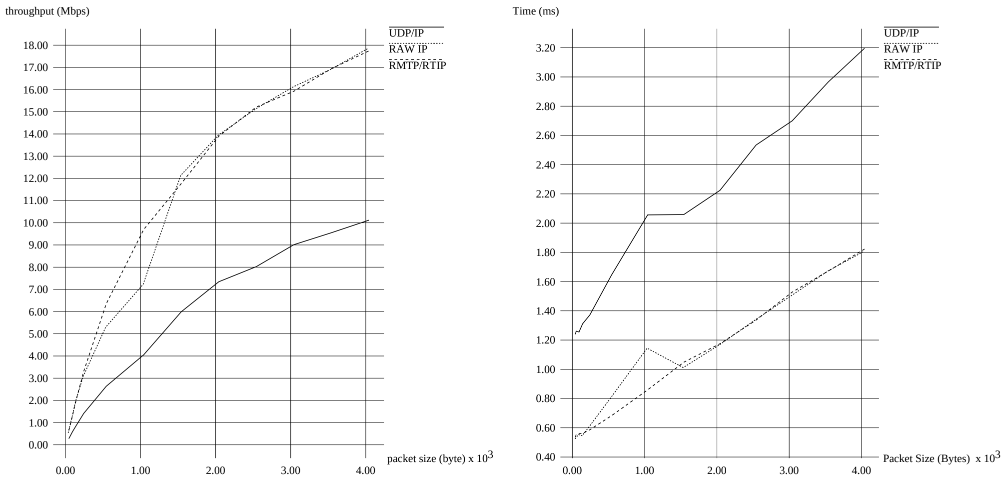
Fig.3. Testbed configuration

In order to evaluate the performance of the RMTP/RTIP implementation, we set up a local testbed and performed a set of experiments on the testbed.

The physical configuration of the testbed is shown in Figure 3 (a). Both *theorem* and *lemma* are DECstation 5000/125's, while *faith* is a DECstation 5000/240. *Fake* is a non-existent machine; it is used to be the receiver of network load. All three workstations are connected to one FDDI ring. While *theorem* and *faith* each have one FDDI interface attached, *lemma* has two FDDI interfaces. The routing tables on the three machines are set up to form a logical network as shown in Figure 3 (b).

4.1 Throughput and Processing Time

Our first experiment compares the throughput of RMTP/RTIP with those of UDP/IP and raw IP. The experiment was performed with one process on *theorem* sending 10,000 packets of same size in a tight loop to a receiving process on *faith*. No load was applied to either the gateway or the hosts.



Throughput   Time to Send One Packet

Fig.4. Performance of UDP/IP, raw IP and RMTP/RTIP

Tests were performed with different packet sizes and underlying protocols. For RMTP/RTIP, the rate parameters (Xmin, Xave, I) were set so that the "reserved rate" was higher than the achievable rate. This was just for experimentation purpose, so that packets could be sent back to back without being held by the rate control mechanism. Figure 4 shows the results of the experiment. The diagram on the left shows the throughput, and the diagram on the right shows the amount of time to send a packet during a tight sending loop on a DECstation 5000/125. It can be seen from the figure that the throughput achieved using RMTP/RTIP is almost the same as that using raw IP. This result is not surprising. It has been observed in [CJRS89] that, for bulk data transfers, the cost of operations on data such as copying or checksumming dominates protocol processing. Since neither RMTP/RTIP nor raw IP do checksumming, the operations on data are the same for both of them, thus the time for sending one packet and the throughput are approximately the same in both cases. The UDP throughput was measured with UDP checksumming turned on, which explains the discrepancy between its curves and the curves of raw IP and RMTP/RTIP.

Figure 5 shows the breakdown of the processing time of a RMTP/RTIP packet on a DS 5000/125. As can be seen, the RMTP/RTIP protocol processing time is about 68 μs per packet, and the driver software processing time is about 114 μs per packet. Socket level processing includes copying the data from user space into kernel space; transmission includes copying the packet from the kernel memory into the interface adaptor and transmitting the packet onto the FDDI ring. Both types of processing involve data movement and account for a larger fraction of the total processing time.

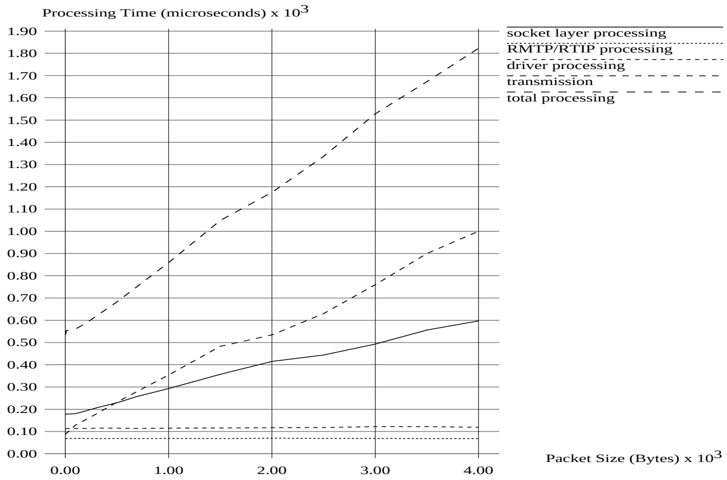


Fig.5. Breakdown of Processing Times In the Kernel

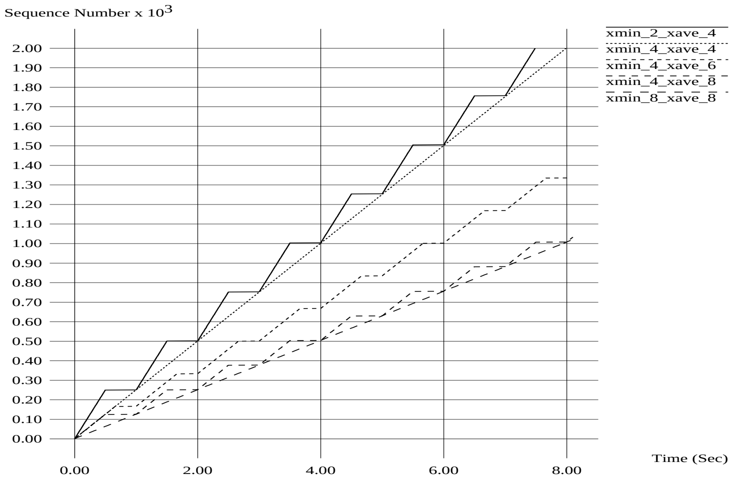


Fig.6. Effects of Rate-Control at the Source in RMTP/RTIP

4.2 Effects of Rate Control

One of the functions of RMTP is to perform rate control at the source so that the packets are injected into the network at a rate not higher than declared when the channel was established. This experiment was designed to check the effectiveness of rate control.

In the experiment, a process on *theorem* sent packets continuously to a process on *faith* through a real-time channel in a tight loop. The receiver program takes a timestamp upon receiving of each packet.

Figure 6 displays the relationship between packet sequences and timestamps. Each of the experiments whose results are reported in the figure was performed independently with different values of *xmin* and *xave*. *Xmin* and *xave* are expressed in milliseconds. The same value of 1 second is used for I in all experiments.

Although the sender program sends packets in a tight loop, the rate control mechanism in the RMTP/RTIP protocol ensures that the source never sends packets faster than what is specified by traffic parameters. As can been seen in the figure, the *(xmin, xave, I)* traffic restrictions have been indeed satisfied by the rate control mechanism.

4.3 Load on Gateway

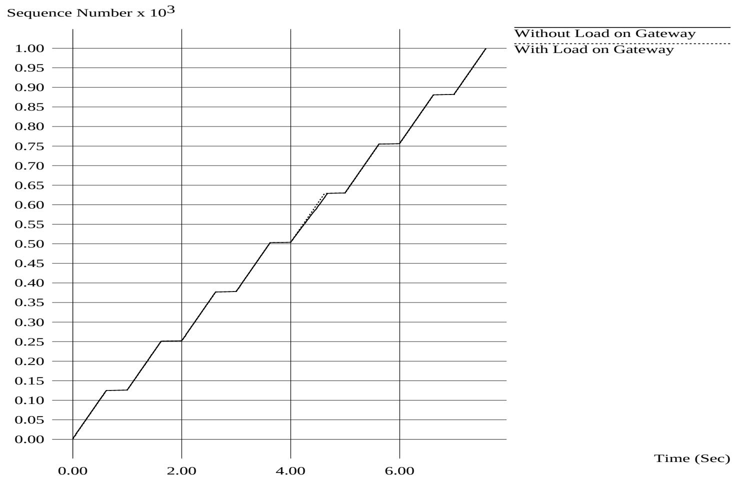


Fig. 6. Effect of gateway load on real-time channel

This experiment was designed to examine the effect of gateway load on real-time channels. In the experiment, a process on *theorem* sent packets to a process on *faith* via a real-time channel. We timestamped each packet and recorded its sequence number at the receiving end. Two tests were performed: gateway unloaded and gateway loaded. To load the gateway, a couple of processes were created on the gateway machine *lemma*, and each of them sent raw IP packets in a tight loop to the non-existing machine *fake*. Figure 8 shows both the loaded and the unloaded case. We can see that the load on the gateway did not affect the performance of the real-time channel.

A similar experiment was performed for UDP. Under load, a significant fraction of UDP packets were dropped at the gateway.

4.4 Co-existence of Real-Time Channels

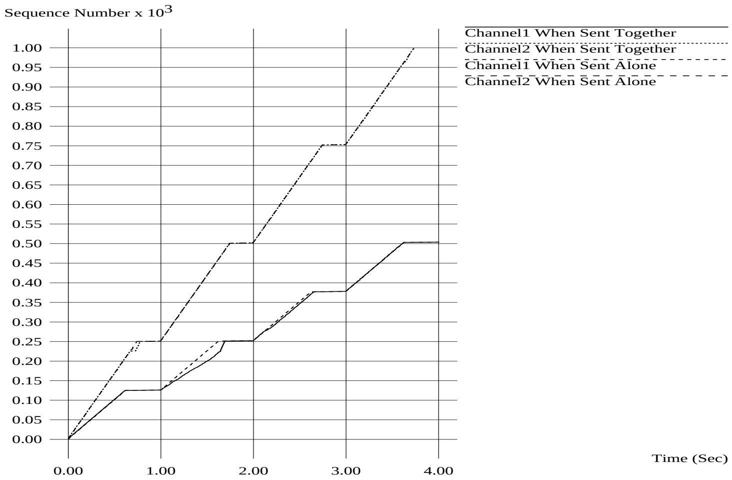


Fig. 7. Interference between two real-time channels

The previous experiment shows that RMTP/RTIP traffic was not affected by the presence of IP traffic. A more interesting case is that of multiple real-time channels.

We did three tests: channel 1 active alone, channel 2 active alone, and channel 1 and channel 2 active simultaneously. We timestamped each packet and recorded its sequence number at the receiving end in all the tests. Figure 7 shows the sequence number vs. time graph for all three cases. As can been seen in the figure, for both channel 1 and channel 2, the patterns of packet arrival are almost the same regardless of whether the other channel is active.

5 Related Work

Several other algorithms or protocolshave been proposed to provide Quality of Service in an internetworking environment. Among them are: the Stream Protocol Version II (ST-II) [Top90], the Multipoint Congram-oriend High-performance Internet Protocol (MCHIP) [Par90], the Flow Protocol [Zha89], a proposal for predicted service [CSZ92], and a proposal for resource management in datagram networks [Flo92].

There have been two implementations of ST-II [Seo91, PP91], but, to the best of our knowledge, no performance data have been reported. Also, to the best of our knowledge, none of the other proposed algorithms or protocols has been fully implemented yet.

6 Conclusion and Future Work

We have conducted a preliminary measurement study to evaluate our prototype implementation of RMTP and RTIP. The results obtained are encouraging: the throughput obtained by using RMTP/RTIP is comparable to that obtained by using raw IP; the rate control mechanism in RMTP/RTIP effectively enforces the traffic specification of communication clients; the scheduling mechanism in RTIP protects the real-time channel so that the performance of a real-time channel is not affected by the presence of IP traffic or other real-time channels in the network.

There are certain limitations to this research, which we would like to address in future works:

- The current testbed is a local area network. We would like to perform similar measurements in an internetwork such as the Xunet-2 [FKK+92] network and the SequoiaNet [FPP].
- All the experiments were performed with synthetic loads. In the future, we would like to experiment with real applications such as video conferencing or scientific visualization.
- The testbed consists of only DECstations. A heterogenous environment would be more interesting. We are in the process of moving the implementation of RMTP/RTIP to the Sun 4/280 and SGI IRIS-4D workstations.

7 Acknowledgement

Domenico Ferrari guided and supervised the entire project. John Limb of Hewlett Packard Co. made it possible for Hui Zhang to implement the first prototype of RMTP/RTIP on a HP9000/700 workstation. Jim Hughes of Hughes Co. loaned us a 80 ns resolution timer board. Fred Templin shared with us his expertise in Ultrix. Bruce Mah helped us to set up the testbed and provided system administrative support.

References

[BM91] Anindo Banerjea and Bruce Mah. The real-time channel administration protocol. In *Proceedings of the Second International Workshop on Network and Operating System Support for Digital Audio and Video*, pages 160–170, Heidelberg, Germany, November 1991. Springer-Verlag.

[CJRS89] David D. Clark, Van Jacobson, John Romkey, and Howard Salwen. An analysis of TCP processing overhead. *IEEE Communications Magazine*, June 1989.

[CSZ92] David Clark, Scott Shenker, and Lixia Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceedings of SIGCOMM'92*, Baltimore, Maryland, August 1992.

[FBZ92] Domenico Ferrari, Anindo Banerjea, and Hui Zhang. Network support for multimedia: a discussion of the Tenet approach. Technical Report TR-92-072, International Computer Science Institute, Berkeley, California, October 1992.

[Fer90] Domenico Ferrari. Client requirements for real-time communication services. *IEEE Communications Magazine*, 28(11):65–72, November 1990.

[Fer92] Domenico Ferrari. Real-time communication in an internetwork. Technical Report TR-91-001, International Computer Science Institute, Berkeley, California, January 1992.

[FKK+92] Alexander G. Fraser, Chuck R. Kalmanek, A.E. Kaplan, William T. Marshall, and R.C. Restrick. Xunet2: A nationwide testbed in high-speed networking. In *Proceedings of INFOCOM*, Firense, Italy, May 1992.

[Flo92] Sally Floyd. Issues in flexible resource management for datagram networks. In *Proceedings of the 3rd Workshop on Very High Speed Networks*, Maryland, March 1992.

[FPP] Domenico Ferrari, Joe Pasquale, and Gorge Polyzos. Network issues for Sequoia 2000. In *Proceedings of COMPCOM 92*, pages 401–406.

[FV90] Domenico Ferrari and Dinesh Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications*, 8(3):368–379, April 1990.

[LMKQ89] Samuel J. Leffler, Marshall Kirk Mckusick, Michael J. Karels, and John S. Quarterman. *The Design and Implementation of the 4.3 BSD UNIX Operating System*. Addison-Wesley Publishing Company, 1989.

[MGWF92] Mark Moran, Amit Gupta, Bernd Wolfinger, and Domenico Ferrari. A continuous media communication service and its implementation, December 1992. to appear in GLOBECOM '92.

[Par90] Guru Parulkar. The next generation of internetworking. *ACM SIGCOMM Computer Communication Review*, January 1990.

[Pos81] Jon Postel. Internet protocol, September 1981. RFC 791.

[PP91] Craig Partridge and Stephen Pink. An implementation of the revised internet stream protocol (ST-2). In *Proceedings of the Second International Workshop on Network and Operating System Support for Digital Audio and Video*, Heidelberg, Germany, November 1991. Springer-Verlag.

[Seo91] Karen Seo. ST-II – new release, November 1991. Connection IP (cip) mailing list.

[Top90] Claudio Topolcic. Experimental internet stream protocol, version 2 (ST-II), October 1990. RFC 1190.

[VZ91] Dinesh Verma and Hui Zhang. Design documents for RMTP/RTIP, May 1991. unpublished internal technical report.

[WM91] Bernd Wolfinger and Mark Moran. A continuous media data transport service and protocol for real-time communication in high speed networks. In *Proceedings of the Second International Workshop on Network and Operating System Support for Digital Audio and Video*, pages 171–182, Heidelberg, Germany, November 1991. Springer-Verlag.

[Zha89] Lixia Zhang. *A New Architecture for Packet Switched Network Protocols*. PhD dissertation, Massachusetts Institute of Technology, July 1989.

[ZVF92] Hui Zhang, Dinesh Verma, and Domenico Ferrari. Design and implementation of the real-time internet protocol. In *Proceedings of the IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems*, Tucson, Arizona, February 1992.