13. Appendix B. Output Timestamps

This appendix discusses how output timestamps might be used to improve timekeeping accuracy with NTP. There are two approaches to this issue, depending on whether or not it is possible to uniquely identify the output timestamp returned from the output driver to the particular association that generated the message. In order to avoid having to make a special case out of a generic output-timestamp facility, it would be undesirable to utilize information in the message itself to do this, so the only way this would be possible would be to assign a unique number to each transmitted message somewhere in the buffer structure associated with the message, but not in the message itself, and then to return the output timestamp and this value to the application that generated the message. This may prove reasonably awkward in practice, especially in the case of multi-homed servers with a sizable client population.

If it is assumed that only the output timestamps are available and that these are returned in the order of transmission without duplication or loss, the output driver needs to know only the process that provided the buffer and return only the output timestamp itself. The daemon keeps a list of those generated messages that have been queued, but not actually sent, as indicated by the arrival of a timestamp from the output driver. The daemon then can do one of two things with the returned output timestamp: It can insert the timestamp as an additional field in the next following transmitted message (assuming with confidence that the queueing time is always less than the interval between transmitted messages). In this case the receiver can (optionally) use it to correct the next previous delay/offset calculations after the fact. Implementors would probably choose to pipeline these calculations and save the state variables resulting from one round of timestamp exchange to the next round when the final (output) timestamp is available.

Alternatively, the daemon could compute the difference between the original transmit timestamp and the actual departure time and interpret this as a correction factor to add to all subsequent transmit timestamps. This amounts to computing a statistical estimator of the checksum, authentication, queueing and context-switching delays. It may be the case, but is far from certain, that the error in this estimator may prove insignificant for most purposes and in any case is minimized by the filtering procedures used by the receiver. In this latter case, of course, the additional timestamp field in the transmitted messages is not needed.

If compatibility with previous NTP versions is to be preserved, it is not possible to alter the NTP packet header itself to indicate an additional timestamp field is present. When interoperating with older (Version 3) peers, or if it is to be optional with the new version, the presence of the output timestamp field can be determined from the UDP packet length. Since the use of PDU data beyond the fixed-length NTP header will be shared among the authentication, multicasting and output-timestamp data, a delimited PDU format, perhaps a subset of ASN.1 may be used.

Changes in the existing NTP daemon for Unix (xntp3) to incorporate this feature are not difficult; however, it has not yet been determined whether the various drivers can provided the needed functionality.

and a reasonably effective randomization procedure, for example, means that, on average, each client consumes only 1/16 the available space in the multicast PDU.

12.2. Route Management

In an extended LAN with filtering, spanning-tree bridges, each multicast message may be heard over a significant portion of the entire LAN; however, the spanning-tree routing algorithm may hide the existence of “back alleys” that are not represented in the spanning tree. Assuming that timing information is to be associated with minimum-distance (time, hops, etc.), this may be acceptable. However, when the spanning tree is constructed using a hop metric and the objective is to determine timing on a minimum-delay basis, this may be decidedly suboptimal. An example that comes to mind in involves satellite links such as USAN (if it still exists) and various interconnected VSAT systems.

12. Appendix A. Operation of the Multicast Protocol

The data to be computed by the protocol is an n x n matrix with components of the form $T_1, T_2]_{ij}$ representing the transmit and receive timestamps, respectively, of the last NTP message sent from peer i to peer j. In this formulation no provision is made to compute the dispersion, although this could be facilitated by the usual clock-filter and related variables. The rows and columns of this matrix can be dynamically allocated and deallocated as new peers arrive and depart. A time-to-live field can be associated with each entry in order to keep track of the peers and manage garbage collection.

Broadcast messages are sent at regular intervals depending on the stratum and accuracy expectations. It is probably not worth the complications of a variable-interval polling program, so the minimum polling interval of each peer may be disregarded. Each broadcast includes the T_3 timestamp and, for each peer message received during the preceding interval, its T_1 and T_2 timestamps and Internet address.

As multicast messages are received from peer i the T_1 and T_2 timestamps for peer j are saved in the matrix. When both the ijth and jith entries are available for the peer pair (i,j) the delay and offset between them can be calculates. In this way the delays and offsets between all peer pairs can be calculated. When at least two multicast messages have been transmitted by every peer and received by every other peer, the complete matrix of delays and offsets can be computed.

12.1. Traffic Management

A model for an extended LAN or WAN might operate as follows. A hierarchical collection of subnets arranged by stratum is created. This might involve several hosts at stratum-1 or stratum-2 using multicasting among themselves to insure redundancy and diversity. Each peer is configured for all of the others in the low-stratum subnet using private-key authentication, so a successful attack that disrupts timekeeping is very unlikely. Periodic multicasts are sent about once per minute containing all the data received during the previous minute. For client-server interactions outside the low-stratum subnet, the state information is purged when the message is sent and the storage reclaimed afterward. Since time is never calculated from these interactions, no clock-filter or reachability variables are required.

Clients at the next higher stratum periodically send a client message (mode 3) to a specific set of low-stratum servers. Note that these are not broadcast in the interest of minimizing the network resources required. Each server in the set preserves the Internet address, T_1 and T_2 timestamps until the end of the current polling interval, when these data are broadcast to the general population. Since all clients at this level are assumed to know the encryption key and will believe data only from their configured peers, a successful attack that disrupts timekeeping is highly unlikely.

In a large campus it is probably a good idea to use address filtering and a different multicast address at each level in the hierarchy. The servers at each stratum are expected to know the key of the next lower stratum, as well as the one for the next higher. Client-server interactions with the next lower stratum use the key of that stratum, while multicast messages to the same and next higher stratum use a different multicast address and encryption key.

This scheme only works to significantly reduce network resources if the polling interval used by the high-stratum clients is significantly longer than for low-stratum servers. Using a ratio of 1:16

2. Principal Investigator: “SAFENET Time Management,” University of Delaware, Electrical Engineering Department, for Northeastern Center for Electrical Engineering Education contract A3036-92 6192 43093 (June 1992 - January 1993, $26K).

List of Collaborators

Dissertation advisor: Prof. Bruce Arden, formerly with University of Michigan Computer and Communications Sciences Department

Graduate Students: Paul Schragger, Erik Perkins, Alden Jackson (joint with Charles Boncelet), Tim Hall (joint with Charles Boncelet), Ken Monington (joint with Gonzolo Arce)

Collaborators not listed in contracts, grants or co-authors

John Elias, University of Delaware Electrical Engineering Department (Highball Project)

David Clark, MIT Laboratory for Computer Science (chair of Internet Research Steering Group)

Robert Braden, USC Information Sciences Institute (chair of End-to-End Research Group)

Various participants in DARTnet experiments for Paul Mockapetris, DARPA Computer Systems Technology Office

Paal Spilling, Norwegian Telecommunications Administration Research Establishment (experiments characterizing network jitter on long Internet paths)

Alfred Ganz and Jordan Becker, Advanced Network Services (experiments characterizing network jitter on NSFnet T3 Network backbone paths)

Ashok Agrawala, University of Maryland Computer Science Department (NSFnet traffic characterization experiments)

Judah Levine, NIST Time and Frequency Division (NTP experiments and measurements)

Lars Mathiesen, University of Copenhagen Computer Science Department (joint project in implementing NTP Version 3 for Unix)

Dennis Ferguson, Advanced Network Services (formerly with University of Toronto Computer Science Department) (joint project in implementing NTP Version 3 for Unix)

4. Mills, D.L. Modelling and analysis of computer network clocks. Electrical Engineering Department Report 92-5-2, University of Delaware, May 1992, 29 pp.

5. Mills, D.L. Network Time Protocol (Version 3) specification, implementation and analysis. DARPA Network Working Group Report RFC-1305, University of Delaware, March 1992, 113 pp. Previous draft published as: Electrical Engineering Department Report 90-6-1, University of Delaware, June 1990.

Recent Publications Related to Other Work

1. Mills, D.L., and H.-W. Braun. The NSFNET Backbone Network. *Proc. ACM SIGCOMM 87 Symposium* (Stoweflake Vermont, August 1987), 191-196.

2. Mills, D.L. The Fuzzball. *Proc. ACM SIGCOMM 88 Symposium* (Palo Alto, California, August 1988), 115-122.

3. Mills, D.L., C.G. Boncelet, J.G. Elias, P.A. Schragger and A.W. Jackson. Highball: a high speed, reserved-access, wide-area network. Electrical Engineering Department Report 90-9-3, University of Delaware, September 1990.

4. Mills, D.L. A computer-controlled LORAN-C receiver for precision timekeeping. Electrical Engineering Department Report 92-3-1, University of Delaware, March 1992, 63 pp.

5. Boncelet, C.G., and D.L. Mills. A labeling algorithm for just-in-time scheduling in TDMA networks. *Proc. ACM SIGCOMM 92 Symposium* (Baltimore Maryland, September 1992).

Recent Contracts and Grants

1. Principal Investigator: "Research in Internet Architectures and Protocols," University of Delaware, Electrical Engineering Department, for DARPA Information Processing Technology Office contract N00140-87-C-8901 (June 1987 - May 1989, $282K).

2. Principal Investigator: "Support of the National Science Foundation Supercomputer Network Program" University of Delaware, Electrical Engineering Department, for NSF Division of Network and Communications Research and Infrastructure grant NCR-86-12015 (September 1986 - February 1989, $29K).

3. Principal Investigator: "Support of the National Science Foundation Networking Program," University of Delaware, Electrical Engineering Department, for NSF Division of Network and Communications Research and Infrastructure grant NCR-89-13623 (September 1989 - September 1992, $103K).

4. Principal Investigator: "SAFENET Time Management," University of Delaware, Electrical Engineering Department, for Southeastern Center for Electrical Engineering Education contract N609 21-87-D-A315 (January 1991 - January 1992, $40K).

Current Contracts and Grants

1. Principal Investigator (with C.G. Boncelet): "Performance and Policy Dimensions in Internet Routing," University of Delaware, Electrical Engineering Department, for DARPA Information System Technology Office contract NAG-2-638 (February 1990 - January 1994, $849K).

11. Biographical Information for David L. Mills

David L. Mills is Professor of Electrical Engineering at the University of Delaware and presently leads projects in high-speed networks and internetworking research sponsored by the Defense Research Projects Agency, National Science Foundation and Department of the Navy. He is a Principal Investigator for the Highball Project, which is studying high speed routing, switching and synchronization technology. His research activities have been concentrated in the areas of network architecture, protocol engineering and experimental studies using the DARPA/NSF Internet system.

He has for many years been an active contributor to the field of computer network time synchronization. Protocols he developed, prototyped and deployed evolved to the Network Time Protocol, which is now in widespread use in the Internet from the Pacific Rim to Eastern Europe to South Africa. One of his current interests is in extending the accuracy, stability and robustness of this technology to the submillisecond regime on desktop workstations.

He is a member of the Internet Research Steering Group, End-to-End Research Group and AT&T Network Architecture Advisory Panel, and formerly chaired the DARPA Internet Architecture Task Force and Gateway Algorithms and Data Structures Task Force. He served on the National Academy of Sciences Committee on Survivable Telecommunications Networks, DARPA Internet Activities Board and NSF Network Technical Advisory Group, as well as several groups involved with internetworking research coordination and the establishment of the National Research and Education Network and the High Performance Computer and Communications Initiative.

Before joining the faculty in 1986, Dr. Mills was a Director (Networks) at M/A-COM Government Systems Division (Linkabit) and led DARPA-sponsored R&D projects in packet-switching network architectures and application protocols. Before that he was a Senior Research Scientist at COMSAT Laboratories, where he worked in the areas of packet-switching satellite and internetworking technologies, and Assistant Professor of Computer Science at the University of Maryland, where he worked on several research projects in distributed computer networks and operating systems.

Dr. Mills earned a Doctorate in Computer and Communication Sciences at the University of Michigan in 1971 and has held postdoctoral positions at the University of Edinburgh (Scotland) and U.S. Defense Communications Agency. He has published and lectured extensively on data communications, computer networks and operating systems and has been a consultant to the United Nations, as well as a number of corporations and government agencies. He is a member of Sigma Xi, Eta Kappa Nu, Tau Beta Pi, Association for Computing Machinery, IEEE Computer and Communications Societies and Internet Society.

Recent Publications Related to Proposed Work

1. Mills, D.L. On the accuracy and stability of clocks synchronized by the Network Time Protocol in the Internet system. *ACM Computer Communication Review 20, 1* (January 1990), 65-75.

2. Mills, D.L. Internet time synchronization: the Network Time Protocol. *IEEE Trans. Communications 39, 10* (October 1991), 1482-1493.

3. Mills, D.L. On the chronology and metrology of computer network timescales and their application to the Network Time Protocol. *ACM Computer Communications Review 21, 5* (October 1991), 8-17.

27. Vasanthavada, N., and P.N. Marinos. Synchronization of fault-tolerant clocks in the presence of malicious failures. *IEEE Trans. Computers C-37, 4* (April 1988), 440-448.

Hardware Clock Synchronization

28. Krishna, C.M., K.G. Shin and R.W. Butler. Ensuring fault tolerance of phase-locked clocks. *IEEE Trans. Computers C-34, 8* (August 1985), 752-756.

29. Mills, D.L. The Fuzzball. *Proc. ACM SIGCOMM 88 Symposium* (Palo Alto, CA, August 1988), 115-122.

30. Ramanathan, P., D.D. Kandlur and K.G. Shin. Hardware-assisted software clock synchronization for homogenous distributed systems. *IEEE Trans. Computers C-39, 4* (April 1990), 514-524.

31. Ramanathan, P., K.G. Shin and R.W. Butler. Fault-tolerant clock synchronization in distributed systems. *IEEE Computer 23, 10* (October 1990), 33-42.

32. Shin, K.G., and P. Ramanathan. Clock synchronization of a large multiprocessor system in the presence of malicious faults. *IEEE Trans. Computers C-36, 1* (January 1987), 2-12.

33. Shin, K.G., and P. Ramanathan. Transmission delays in hardware clock synchronization. *IEEE Trans. Computers C-37, 11* (November 1988), 1465-1467.

34. Vasanthavada, N., and P.N. Marinos. Synchronization of fault-tolerant clocks in the presence of malicious failures. *IEEE Trans. Computers C-37, 4* (April 1988), 440-448.

35. Wilcox, D.R. Backplane bus distributed realtime clock synchronization. Technical Report 1400, Naval Ocean Systems Center, December 1990, 52 pp.

36. Wilcox, D.R. Local area network distributed realtime clock synchronization. Technical Report 1466, Naval Ocean Systems Center, November 1991, 77 pp.

11. Winkler, G.M.R. Changes at USNO in global timekeeping. *Proc IEEE 74, 1* (January 1986), 151-154.

Time Synchronization Protocols and Services

12. Defense Advanced Research Projects Agency. Internet Protocol. DARPA Network Working Group Report RFC-791, USC Information Sciences Institute, September 1981.

13. Defense Advanced Research Projects Agency. Internet Control Message Protocol. DARPA Network Working Group Report RFC-792, USC Information Sciences Institute, September 1981.

14. Digital Time Service Functional Specification Version T.1.0.5. Digital Equipment Corporation, 1989.

15. Lamport, L. Concurrent reading and writing of clocks. *ACM Trans. Computing Systems 8,* 4 (November 1990), 305-310.

16. Lindsay, W.C., and A.V. Kantak. Network synchronization of random signals. *IEEE Trans. Communications COM-28, 8* (August 1980), 1260-1266.

17. Marzullo, K., and S. Owicki. Maintaining the time in a distributed system. *ACM Operating Systems Review 19, 3* (July 1985), 44-54.

18. Mills, D.L. Network Time Protocol (version 2) - specification and implementation. DARPA Network Working Group Report RFC-1119, University of Delaware, September 1989.

19. Mills, D.L. Measured performance of the Network Time Protocol in the Internet system. *ACM Computer Communication Review 20, 1* (January 1990), 65-75.

20. Mills, D.L. Internet time synchronization: the Network Time Protocol. *IEEE Trans. Communications 39, 10* (October 1991), 1482-1493.

21. Mills, D.L. On the chronology and metrology of computer network timescales and their application to the Network Time Protocol. *ACM Computer Communications Review 21, 5* (October 1991), 8-17.

22. Mills, D.L. Network Time Protocol (Version 3) specification, implementation and analysis. DARPA Network Working Group Report RFC-1305, University of Delaware, March 1992, 113 pp.

23. Mills, D.L. Modelling and analysis of computer network clocks. Electrical Engineering Department Report 92-5-2, University of Delaware, May 1992, 29 pp.

24. Postel, J. User Datagram Protocol. DARPA Network Working Group Report RFC-768, USC Information Sciences Institute, August 1980.

25. Postel, J. Time protocol. DARPA Network Working Group Report RFC-868, USC Information Sciences Institute, May 1983.

26. Schneider, F.B. Implementing fault-tolerant services using the state machine approach: a tutorial. *ACM Computing Surveys 22, 4* (December 1990), 299-320.

9. Opportunities for Undergraduate Research

It is very important in our research activities to provide a rich set of opportunities for undergraduate engineering research, especially in comprehensive design projects. In past projects undergraduate Mark Shaw and associates built two generations of computer-based, precision interval-timer devices for use in calibrating timecode receivers and evaluating the performance of NTP. Stefan Levie and associates worked on a software implementation of NTP for the ubiquitous PC, while Neil Corman constructed the prototype for a LORAN-C receiver and controller for the PC. Most recently, Michael Larson built a second-generation printed-circuit version of this receiver.

The work proposed provides many opportunities for undergraduate engineering research projects at the junior and senior level. Some of these, as in the past, can serve as projects for design courses in circuit analysis, digital systems and digital signal processing, including the design of phase-locked loops, timing generators and computer interfaces. There are many pending software projects such as improving the LORAN-C control program and integrating it with NTP support software written and MIT and Bellcore for the PC.

10. Bibliography of Pertinent Literature

Standard Time and Frequency Measurement and Dissemination

1. Allan, D.W., J.H. Shoaf and D. Halford. Statistics of time and frequency data analysis. In: Blair, B.E. (Ed.). *Time and Frequency Theory and Fundamentals*. National Bureau of Standards Monograph 140, U.S. Department of Commerce, 1974, 151-204.

2. Allan, D.W., J.E. Gray and H.E. Machlan. The National Bureau of Standards atomic time scale: generation, stability, accuracy and accessibility. In: Blair, B.E. (Ed.). *Time and Frequency Theory and Fundamentals*. National Bureau of Standards Monograph 140, U.S. Department of Commerce, 1974, 205-231.

3. Frank, R.L. Current developments in LORAN-C. *Proc. IEEE 71, 10* (October 1983), 1127-1139.

4. Johler, F.R. The propagation time of a radio pulse. IEEE Trans. on Anennas and Propagation (November 1963), 661-668.

5. Jones, R.H., and P.V. Tryon. Estimating time from atomic clocks. *J. Research of the National Bureau of Standards 88, 1* (January-February 1983), 17-24.

6. Montague-Smith, H. International time and frequency coordination. Proc IEEE 60, 5 (May 1972), 479-485.

7. Mungall, A.G. Frequency and time - national standards. *Proc. IEEE 74,* 1 (January 1986), 132-136

8. *NIST Time and Frequency Dissemination Services*. NBS Special Publication 432 (Revised 1990), National Institute of Science and Technology, U.S. Department of Commerce, 1990.

9. Smith, J. *Modern Communications Circuits*. McGraw-Hill, New York, NY, 1986.

10. Winkler, G.M.R. Path delay, its variations, and some implications for the field use of precise frequency standards. *Proc. IEEE 60, 5* (May 1972), 522-529.

7.3. Additional Timing-Receiver Drivers

Software drivers for xntp3 have been constructed for a number of radio timecode receivers, including U.S, Canadian, United Kingdom and German time-dissemination services. However, at this time there is no known implementation for a GPS receiver, which is likely to become the source of choice as the cost for the receivers themselves continues to drop. There are several GPS receivers on the market now, including those from TrueTime, Austron and Odetics, and several experimenters at MIT and elsewhere are adapting relatively inexpensive GPS navigation receiver printed-circuit modules as timecode receivers. A full implementation for the Austron receiver, which is being used now for timekeeping only with its one-pulse-per-second (pps) output is needed. Implementations for other types of GPS receivers are expected from collaborators at other institutions.

8. Related Collaborative Activities

In a separate effort funded by the US Navy Surface Weapons Center (NSWC), various timekeeping issues, including protocol standardization, are being presented to the cognizant ANSI and CCITT standards committees. While current activities in these committees are presently only in the preliminary stages, NTP has been presented to these committees along with other protocols, including DTSS from DEC and PCS from IBM. In fact, many discussions between the designers and implementors of these protocols have taken place via electronic mail. These had a significant effect on the design process of NTP itself. It is expected that collaborators at NSWC will advance this agenda using NTP as proposed for the Next Generation Computer program (NGCR), which involves ISO protocols, together with additional protocols intended for real-time control systems.

Recently, a collaboration has begun with Alfred Ganz and Jordan Becker at Advanced Network Services for an investigation of timekeeping issues involved in the NSFnet T3 backbone routers. One of the reasons for this is to improve the precision of delay measurements in order to help find and resolve persistent small idiosyncrasies in hardware and software behavior. Much experience in these matters suggests precision delay measurements are highly useful in diagnosing misbehavior of the routing and queueing mechanisms. Another reason for this is to explore the opportunity to use the T3 backbone to provide precision timekeeping services for use by the regional service providers, perhaps assisted by the T3 communication substrate.

In previous collaborations with Judah Levine of the NIST Time and Frequency Division at Boulder, CO, and Paal Spilling of the Norwegian Telecommunications Administration Research Establishment at Oslo, Norway, an NTP service has been established with direct access to the national time and frequency services of the two countries. In another collaboration with Lt. Cmdr. Rex Buddenberg of the US Coast Guard applications of NTP to precise time and frequency recording for LORAN-C navigation transmitters was explored. We have collaborated with many researchers in the use of NTP in various other research programs, including DARPA-funded experiments with the DARTnet network.

Another area of collaboration is with Jeffery Mogul and David Boggs at DEC Palo Alto Research Center. DEC has donated two high performance workstations for use in timekeeping experiments and protocol development. These and ancillary communication and utility input/output devices are resources available at no cost to the proposed project.

While the bulk of the work proposed involves specific features and extensions described previously, some work needs to be expended to improve the operation of xntp3 itself. It is expected that much of this work can be accomplished by collaborators at other institutions, including the many users of the software. A few of the specific needs are detailed in following sections.

7.1. Lessons on Randomization

One of the odd facts not clearly recognized in the design and implementation of NTP daemons was that precision timekeeping can result in destructive message bursts where a busy NTP server can bunch several messages to be transmitted during a one-second interval, for example, and queue them for transmission precisely on the second's epoch. When a number of NTP timekeepers share the same network or backbone, this can result in a nasty spasm where multiple peers transmit messages to each other at substantially the same instant. This happens even after careful consideration of randomization procedures that scatter the times of transmission over the seconds in a poll interval. Unfortunately, the precision of the interval-timer facility of the sending host may not have the granularity to spread these transmissions over the poll interval with granularity finer than one second. This means that, for busy servers serving over 250 clients, the likelihood of several messages stacked up for the same second is uncomfortably high.

The solution to this problem requires the granularity of the interval-timer facility to be substantially reduced; however, at least for Unix systems, the granularity is bounded from below at the tick interval, typically 10-20 milliseconds. In practice, decreasing the granularity from one second to 10 ms reduces the probability of two messages sharing the same timer bin by a factor of 100, which should be enough for the next few years. The extra overhead of bin searching should not be a problem, assuming a good hash algorithm is used. This problem will be addressed as part of a general software update of xntp3.

7.2. Exportable Message-Digest Algorithm

Currently, NTP uses the Digital Encryption Standard (DES) as a message-digest (MD) algorithm to generate a crypto-checksum, called the message authenticator in the specification, uniquely identifying the message. This provides a means to detect message-modification attacks, since a party not in possession of the relevant cryptographic key cannot forge a valid message. (Provisions for other kinds of attacks, such as replay, are discussed in the specification [22].). It is highly unlikely the authenticator can be forged, nor is it likely any other message could result in the same checksum.

However, in the present state of Government regulations, it is forbidden to export DES implementations without a cumbersome license procedure, which results in much consternation and resentment on the part of overseas NTP operators. Since NTP was designed, newer algorithms have been designed in which the authenticator is more properly called the message digest, which provides the same function. In particular, the MD5 algorithm is on its way to international standardization and, so far as known now, is exportable. While this is not exactly a drop-in replacement for DES (MD5 uses 128 bits, while DES uses 64) and the computational cost for it is not known compared to DES, there is every reason to include MD5 in the suite of algorithms usable with NTP. For this, an implementation of MD5 compatible with the xntp3 is expected to become available as a collaborative project with Louis Mamakos of the University of Maryland.

radios have a precision only to the millisecond. Even if these radios have on-second pulse outputs, which can be used with present NTP Version 3 implementations, the jitter on these outputs, 100 microseconds or so, considerably exceeds the jitter due to the interrupt latency of a typical modern RISC workstation, 20-30 microseconds.

Notwithstanding the accuracy problems due to radio errors and various jitter sources, the greatest threat to accuracy is oscillator frequency stability. For reasons mentioned above, it is desirable to use an external frequency source, either in the form of a temperature-compensated crystal oscillator and bus interface card or an external frequency standard. However, as shown in recent experiments, it is not practical to expect frequency stabilities better than 1 ppm in the case of onboard uncompensated oscillators or 0.1 ppm in the case of commonly available and relatively inexpensive compensated oscillators [23]. At a frequency error of 0.1 ppm, and an accuracy expectation in the low microseconds, phase corrections would have to be available at intervals less than a minute.

It would be highly desirable to reduce or eliminate the requirement for frequent interaction between the NTP server and its peers in order to achieve the highest accuracy. One way to do this is to employ NTP or a relatively imprecise timecode receiver to supply the seconds numbering and resolve the precision on-time seconds epoch using a precision frequency source and counter. This approach has in fact been implemented using both a cesium clock, a Global Positioning Service (GPS) timing receiver and a LORAN-C timing receiver. Each of these devices furnishes a precise on-time seconds pulse; however, they are frightfully expensive and not likely to find wide acceptance in the Internet.

We have experimented with the construction of a relatively inexpensive LORAN-C receiver with some success [4]. It consists of a dedicated IBM-compatible PC with a special digital interface card and an outboard analog receiver. With a good oven-controlled crystal oscillator this equipment produces a on-time seconds pulse to a precision of 200 nanoseconds with an oscillator stability of a few parts in 10^{10}, which is comparable to a GPS timing receiver, but costs far less. In fact, timing from LORAN-C is expected to be better than GPS itself, since the latter signals are intentionally degraded for non-military users.

Encouraged by the success of the prototype implementation, we have designed an etched circuit board including both the digital and analog circuitry and are in process of constructing it. While the present design is for the PC, the goal of this enterprise eventually is to construct an interface for a modern workstation, such as Sun, DEC or IBM, which could be replicated and deployed in places such as the current regional networks.

7. NTP Version 3 Software Upgrades

The original reference implementation for NTP Version 3 was constructed in assembly code for the Fuzzball operating system [29] and is not readily portable. This implementation was the vehicle for the development of the protocol and the algorithms described in the version 3 specification RFC-1305. Subsequent to this implementation the NTP Version 2 implementation for Unix built by Dennis Ferguson of the University of Toronto (Canada) was modified to conform to the version 3 specification by Lars Mathiesen of the University of Copenhagen (Denmark). Bancomm, a division of Datum, Inc., is now marketing a packaged, standalone unit which combines an NTP Version 3 implementation, GPS timing receiver and Ethernet interface. The Unix implementation, called xntp3, has become the reference implementation and development platform for the improved protocol and algorithms proposed herein.

then passed to the software driver and NTP daemon for later processing. From this information the daemon can calculate the offset, delay and dispersion of the remote clock relative to the local clock. Assuming symmetrical transmission paths and the output-driver modifications proposed in the previous section, delays and offsets can be computed to a precision of about 3.7 nanoseconds.

Note that in this scheme there is no reference to the onboard frequency source, which can operate independently of the external clock. The onboard source can of course easily be disciplined to either the remote clock or the local (external) clock by simply driving the phase-lock loop with an error signal computed as the difference between the selected source and the onboard source. In the same way the onboard source can be disciplined to the remote reference source, although some way to vary the oscillator frequency is required, perhaps using methods suggested in [23].

6.4. Precision Timekeeping Hardware

At its current level of refinement the limits of accuracy in synchronization subnets involving NTP Version 3 are not due to the capabilities of the architecture or protocol, but due to the inherent stability characteristics of the onboard frequency source, usually an uncompensated crystal oscillator of dubious tolerance and stability. In some modern workstations a high frequency oscillator, perhaps the one controlling the bus cycles, is divided to the range of 100 or 250 Hz, providing a continuous source of interrupts at that frequency. In these workstations the minimum increment in time is 4 or 10 ms, which is too large for most real-time applications involving speech or video. Nevertheless, if continuously resynchronized using NTP and a stable source of local time, these systems are sufficiently stable for most other applications running on modern workstations sharing an Ethernet.

In some systems, including Sun Microsystems and DEC RISC-based workstations, a microsecond timer is available in addition to a 100-Hz or 250-Hz interrupt. With these systems it is in principle possible to time internal events to precisions in the microsecond range; however, there is no interrupt associated with the microsecond timer, so internal and external events cannot be generated with jitter less than the reciprocal of the interrupt frequency. Thus, when timekeeping accuracies better than a few hundred microseconds or frequency accuracies better than a few parts-per-million (ppm) are required, the onboard frequency source in modern workstations must be augmented.

There are several designs for special-purpose timekeeping devices available, including those described in [23], [35] and [36]. Other designs have been discussed with collaborators David Boggs and Jeffery Mogul at DEC Western Research Laboratory. These involve an interface card for the TurboChannel bus and include a time-of-day counter, software profiling counter and general-purpose interval timer. It is expected that DEC will fabricate some number of these devices, including some for a pair of DECstation 5000/240 workstations donated by DEC for use in our timekeeping activities. As our contribution to this effort, we plan to fabricate a complementary interface for the Sun SBus. Opportunities to explore the issues for fabricating a similar interface for other bus technologies will be explored with IBM, especially in connection with the NSFnet backbone routers.

6.5. Precision Reference Sources

Working with currently available timecode receivers can sometimes be a frustrating experience. Due to vagaries of the transmission medium and particular radio design, errors of various kinds sometimes occur. Most receivers depending on low-frequency radio propagation (WWVB, MSF, DCF77) can be relied only to the millisecond and sometimes produce much larger errors due to atmospheric or ionospheric disturbances. In addition to these errors, the timecode formats of most

subsequent message. While this amounts to some intrusion in driver code, it is probably not a major invasion and is useful anyway for system measurements and profiling. Assuming this timestamp is available to the NTP protocol daemon, it remains to be determined how the daemon processes this information and/or includes it in subsequent transmitted messages. Appendix B discusses various approaches to this issue.

6.2. NTP Protocol Modifications for Precision Delay Measurements

In the original application of NTP the accuracy expectations were in the tens of milliseconds, so a precision of one millisecond was considered adequate. At that time only the Fuzzball implementation [29] had the capability to resolve time to the millisecond. In this regime the representation precision of the delay and dispersion fields in the PDU, about 15.3 microseconds, was not a factor. However, timekeeping accuracies have improved to the 100-200 microsecond range and promise to improve even more as faster workstations and networks appear. Therefore, something needs to be done about the precision of these two fields. It is doubtful that the number of bits necessary to represent whole seconds; that is, the number of bits to the left of the fraction point (16 in the present design), will ever be used; and, in fact, the present specification requires no more than four bits.

In order to preserve compatibility, at least with "conventional" precisions, the format of the delay and dispersion fields cannot be changed. However, for those systems that can support precisions to the microsecond and less, a natural approach is to use the precision field of the NTP header as a scale factor in the following way. If the precision is greater than -16 (about 15.3 microseconds), then operate as in the NTP specification; otherwise, shift the fraction point one bit to the left for each increment of precision below -16. Assuming a minimum of four significant bits to the left of the fraction point, the ultimate precision of the delay and dispersion fields is about 3.7 nanoseconds, which is close enough for all except the most exotic surveying applications.

The existing NTP daemon for Unix (xntp3) can be modified to provide this feature. There are two arithmetic formats used by the daemon, 64-bit fixed-point, with the upper 32 bits representing seconds and the lower 32 bits the fraction, and 32-bit fixed-point, with the upper 16 bits representing seconds and the lower 16 bits representing the fraction. The 64-bit format is used for timestamps and offsets; that is, differences between two timestamps. The 32-bit format is used for delays and dispersions. In the proposed mode all calculations would use the 64-bit format and conversions to 32-bit format done only as the values were converted to and from the actual PDU.

6.3. Using NTP with Externally Derived Timestamps

The traditional application of NTP has been to discipline a frequency source onboard the NTP host itself. However, the same algorithms and protocol design can be used to exchange and process timestamps derived from other sources, such as external time and frequency standards. For this purpose the timestamps can be derived from an external clock, consisting of an oscillator and counter which can be read by the hardware interface and software driver. When an NTP message is to be sent, presumably over a low-jitter channel such as a DS1 superframe or FDMA satellite circuit, the hardware interface captures a timestamp and includes it in the media header, as well as inserting it in a FIFO for later processing by the software driver and NTP daemon in a manner similar to that described previously. This represents the T_1 or T_3 timestamps, depending on the direction of transmission.

Upon receipt of an NTP message, the hardware interface inserts a timestamp in the FIFO. This represents the T_2 or T_4 timestamps, depending on the direction of transmission. The timestamp is

associations, together with state variables, clock filters and so forth for all other peer paths in the subnet, up to $O(n^2)$ in a subnet of n peers. An strawman example of this approach is described in Appendix A.

In the other approach each peer operates as it does now using a subset of the subnet peers. The offset and delay values computed for these peers are inserted into the multicast message. Each receiver of this message would run a Bellman-Ford-like algorithm to calculate the delays between all other peers. This technique would usually result in a smaller message, since the data items for a particular PDU would be associated with only the (symmetric) peers of the sending host and not for all members of the subnet, assuming the immediate (neighbor) peers are distinguished from the general population (non-neighbor) peers. This scheme works only if the actual routing graph used by the application is a subgraph of the graph formed by the union of all NTP peer paths.

6. Improving the Accuracy of Synchronized Clocks

While at the present stage of development, it is possible to synchronize computer clocks to the order less than a millisecond, this requires reasonably fast processors and lightly loaded network links. As workstations and networks increase in speed, there is every reasonable conjecture that the need for even higher accuracies in future, especially if NTP is used in conjunction with externally timed devices, such as precision time and frequency standards, digital telephone networks, very large baseline interferometers and similar devices.

There are three issues that arise in improving accuracy and stability of clock synchronization, timestamping NTP messages in the software driver, using NTP with externally derived timestamps and the use of special hardware devices to assist in precision timekeeping. Each of these issues is discussed below.

6.1. Output Driver Timestamps

In cases where the clocks to be synchronized are intrinsic to the measuring host itself, as in the ordinary applications of NTP, timekeeping errors can arise due to stochastic delays between when the time timestamps are captured and the actual time of arrival or departure of an NTP message. To the degree these differences are predictable or estimable with zero bias, they do not materially degrade the accuracy. However, especially in the case of output queues, they are not in general predictable and can be estimated only with suspicious bias. The mechanism to deal with this problem requires the timestamps be captured as close to the actual time of reception and transmission as possible, generally at interrupt time in the software driver for the actual interface over which the NTP messages are transmitted.

In general, the receive (T_2 and T_4) timestamps are more accurate than the transmit (T_1 and T_3) timestamps. This is because the receive timestamps are determined by the input device driver at the time of arrival and differ from the actual time by not much more than the interrupt latency of the device. However, the transmit timestamps are usually determined by the NTP protocol daemon before checksum, authentication and output-queueing operations, which can result in substantial errors. In principle, it would be possible for the output driver to insert these timestamps directly in the message or media-access header, but this makes assumptions on the lower-level driver and protocol which are probably unjustified in practice.

However, while the output driver may not be allowed to modify the message, it can capture the actual time of transmission and pass this information back to the daemon for inclusion in a

real-time simulation and multimedia conferencing systems requiring coordinated playback of audio, video and positioning data. As described later in this proposal, this is possible using multicast techniques and algorithms modified from the NTP Version 3 specification.

A third area of improvement involves the extension of the protocol to handle timing information delivered by external devices, such as auxiliary hardware interfaces or precision frequency standards. Using auxiliary interfaces such as described in [23] it is possible to avoid the notoriously unstable crystal oscillator which is the usual basis for system timing in most computers and provide not only improved frequency jitter and wander characteristics, but to provide precision profiling and time-of-day capture for application programs.

Other areas of interest include improvement in accuracy through the use of timestamps captured in the interface drivers, rather than in the application program or NTP daemon itself. This requires both a minor protocol change, as well as some investment in driver reprogramming. In collaboration with Advanced Network Services a program of experiments is proposed with the NSFnet T3 routers and existing time servers. This program is expected to improve the timekeeping accuracy of the T3 network to the order less than a millisecond. Finally, progress toward adapting NTP for use in the ISO connectionless protocol suite is to be pursued.

5.1. Improving Accuracy in Multicast Modes

In order to achieve the highest accuracy, NTP peers must exchange timestamps on a full-duplex basis with selected peers of the synchronization subnet. This is necessary in order to compensate for the effects of transmission delay. It would be highly desirable to disseminate time via multicast modes, rather than full-duplex ones, and NTP provides a multicast mode to do this. However, this is practical only in cases where the delays are small, such as in LANs, or can be accurately estimated. However, in the case of WANs the errors can be significant, especially in cases of shared backbone links.

In addition, in applications such as real-time, parallel, distributed, event simulation or multimedia conferencing systems, it is sometimes necessary to know the delays between each participant in the conference in addition to just the selected peers. In the present design, to know the delays between all neighbors in a subnet, NTP peers must exchange timestamps between every pair of participants and then pass this information along to the other peers in the conference using methods outside the scope of NTP. Obviously, distributing this information would be much more efficient within NTP itself and using multicast rather than point-point protocols.

In principle it is possible to distribute the delays and offsets computed by one peer relative to its neighboring peers using the NTP control-message (mode 6) protocol, or even to design a special protocol using mode 7, which is presently reserved for private use. However, it would be more efficient to build the information into the stock NTP protocol data unit (PDU) by including it at the end of the message itself and before the optional authentication fields, as described in the specification [22]. In this scheme the presence of the area and the number of (fixed-size) data entries can be determined from the UDP length field.

In the following the timestamp numbering is as shown in Figure 3. There are two approaches to resolve offsets and delays for all peer paths in the subnet. In one approach each data entry sent by a peer consists of the T_1 and T_2 timestamps as received by the peer for all other peers in the subnet. This message is also heard by all other peers in the subnet, so each can calculate the offsets and delays relative to any other peer in the subnet. However, this method requires each peer to maintain

$$\delta = a - b \quad \text{and} \quad \theta = \frac{a + b}{2}.$$

The errors in these quantities, both systematic and random, are discussed in [23].

Each NTP message includes the latest three timestamps T_1, T_2 and T_3, while the fourth timestamp T_4 is determined upon arrival of the message. Thus, both peers *A* and *B* can independently calculate delay and offset using a single bidirectional message stream. This is a symmetric, continuously sampled, time-transfer scheme similar to those used in some digital telephone networks [16]. Among its advantages are that reliable message delivery is not required (see [23] for an extended discussion of these issues).

The clock-selection algorithm determines from among all peers a suitable subset of peers capable of providing the most accurate and trustworthy time using principles similar to those described in [27]. This is done using a cascade of two subalgorithms, one based on interval intersections to cast out faulty peers [17] and the other based on maximum-likelihood principles to improve accuracy [20]. The resulting offsets of this subset are first combined on a weighted-average basis using an algorithm such as described in [5] and then processed by a phase-lock loop (PLL). In the PLL the combined effects of the filtering, selection and combining operations are to produce a phase-correction term, which is processed by the loop filter to control the voltage-controlled oscillator (VCO) frequency. The VCO furnishes the phase (timing) reference to produce the timestamps used in all timing calculations.

5. Approach and Research Plan

At the present time the NTP Version 2 described in RFC-1119 [18] is considered an mature protocol, with probably well over 10,000 hosts running one of three Unix daemons throughout much of the world. The NTP Version 3 specification described in RFC-1305 [22] has been specifically designed to provide provable error bounds, reduce errors and minimize network overhead. Daemons for both Fuzzball [29] and Unix have been deployed in deployed in several dozen domestic and overseas sites, as well as DARTnet and other experimental networks. A realistic evaluation would probably conclude that the protocol machine and various intricate algorithms are working to the original expectations and do deliver accurate and reliable time to within the limitations of the hardware and network paths available.

There are several areas not addressed in the current protocol specification that need to be considered for future applications, such as distributed, real-time simulation and multimedia conferencing. There are other areas that need to be considered in the interests of efficient use of network resources, such as the use of multicast techniques. This proposal attempts to collect some of these issues as topics for further study and possible incorporation in a future NTP Version 4.

One area of improvement involves the use of multicast techniques for efficient synchronization in wide-area synchronization subnets. While it has always been possible to employ these techniques to synchronize local-area networks with fair accuracy, in the case of wide-area networks the accuracy has been generally unacceptable. The protocol modifications suggested later in this proposal should provide the same accuracy with multicast as with traditional point-point techniques.

A second area of improvement involves the capability to determine delays and offsets between all members of a synchronization subnet by every member of the subnet without requiring peer associations between all members. This capability is important for use in distributed, parallel,

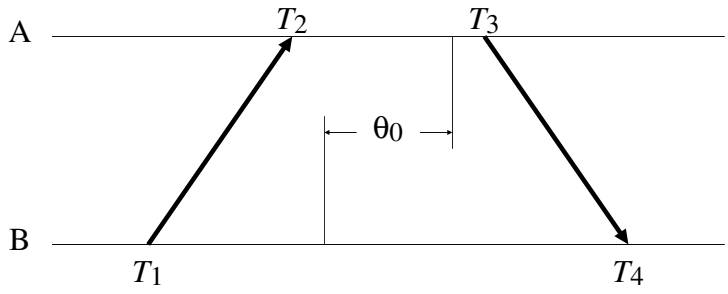

Figure 3. Measuring Delay and Offset

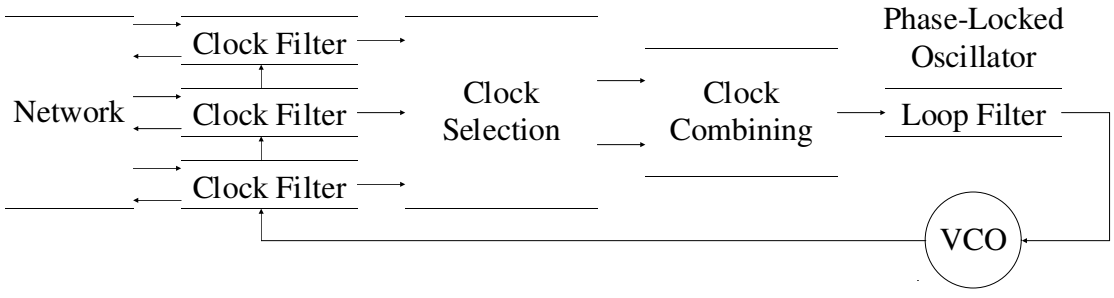

Figure 2. Network Time Protocol

subnet. A typical subnet is shown in Figure 1a, in which the nodes represent subnet servers, with normal level or stratum numbers determined by the hop count from the root (stratum one), and the heavy lines the active synchronization paths and direction of timing information flow. The light lines represent backup synchronization paths where timing information is exchanged, but not necessarily used to synchronize the local clocks. Figure 1b shows the same subnet, but with the line marked *x* out of service. The subnet has reconfigured itself automatically to use backup paths, with the result that one of the servers has dropped from stratum 2 to stratum 3. In practice each NTP server synchronizes with several other servers in order to survive outages and Byzantine failures using methods similar to those described in [32].

Figure 2 shows the overall organization of the NTP time-server model, which has much in common with the phase-lock methods summarized in [31]. Timestamps exchanged between the server and possibly many other subnet peers are used to determine individual roundtrip delays and clock offsets, as well as provide reliable error bounds. As shown in the figure, the computed delays and offsets for each peer are processed by the clock-filter algorithm to reduce incidental timing noise. As described in [22], this algorithm selects from among the last several samples the one with minimum delay and presents the associated offset as the output.

Figure shows how NTP timestamps are numbered and exchanged between peers *A* and *B*. Let T_1, T_2, T_3, T_4 be the values of the four most recent timestamps as shown and, without loss of generality, assume $T_3 > T_2$. Let

$$a = T_2 - T_1 \quad \text{and} \quad b = T_3 - T_4 \ .$$

If the subnet delays from A to B and from B to A are similar, the delay δ and offset θ of *B* relative to *A* at time T_i are close to

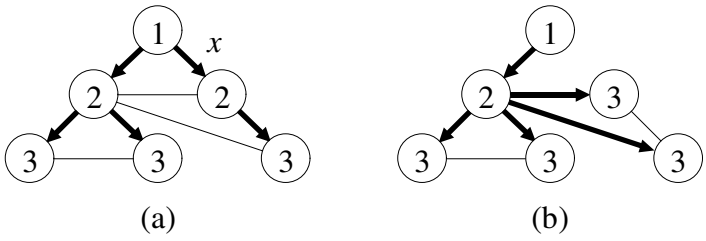

Figure 1. Subnet Synchronization Topologies

The time-synchronization protocol selected for use in the proposed work is the Network Time Protocol (NTP) developed for the Internet over the last several years [22]. In principle, others such as the Digital Time Synchronization Service (DTSS) [14] could be used as well. NTP has evolved over a course of three versions, with NTP Version 2 [16] an Internet Standard (Recommended) and Version 3 [22] advancing to Draft Standard.

The local clocks of present operating systems such as Unix and Fuzzball can be synchronized to within a few tens of milliseconds using NTP in the Internet of today [19]. However, as workstations and networks become faster, there is every expectation that future applications will require timekeeping to the submillisecond regime. This requires in essence a complete reexamination of all elements of the timekeeping apparatus, including the clock design and its synchronization mechanism. Before advancing to detailed plans in this proposal, it may be helpful to summarize the architecture and algorithms of this protocol.

4. Network Time Protocol

The Network Time Protocol (NTP) is used by Internet time servers and their peers to synchronize clocks, as well as automatically organize and maintain the time synchronization subnet itself. It is evolved from the Time Protocol [25] and the ICMP Timestamp Message [12], but is specifically designed for high accuracy, stability and reliability, even when used over typical Internet paths involving multiple gateways and unreliable networks. This section contains an overview of the architecture and algorithms used in NTP. A formal description and specification of the protocol is contained in [22]. A detailed description of the NTP architecture and protocols is contained in [20], while a summary of operational experience and performance is contained in [19] and a detailed discussion on timescales is contained in [21]. A comprehensive analysis of the clock models and error processes found in NTP is contained in [23].

NTP and its implementations have evolved and proliferated in the Internet over the last decade, with NTP Version 2 adopted as an Internet Standard (Recommended) [18] and NTP Version 3 now Proposed Standard [22]. NTP is built on the Internet Protocol (IP) [12] and User Datagram Protocol (UDP) [24], which provide a connectionless transport mechanism; however, it is readily adaptable to other protocol suites. The protocol can operate in several modes appropriate to different scenarios involving private workstations, public servers and various subnet configurations. A lightweight association-management capability, including dynamic reachability and variable poll-interval mechanisms, is used to manage state information and reduce resource requirements. Optional features include message authentication based on crypto-checksums and provisions for remote control and monitoring.

In NTP one or more primary servers synchronize directly to external reference sources such as radio clocks. Secondary time servers synchronize to the primary servers and others in the synchronization

of hosts over major portions of the global Internet. This work is relevant in expanding the scope of timekeeping technology to such applications as real-time, parallel, discrete-event simulation, multimedia conferencing, retrieval of digitized maps and experiment control. It is also relevant to improving the performance and efficiency of the technology itself in regimes including gigabit networking and superscaler workstations.

The effect of the proposed research on the infrastructure of science and engineering impacts both computer-network timekeeping technology in general and current and future real-time networking applications in particular. A continuing experiment program in precision timekeeping has in the past proved highly useful in the identification and resolution of insidious misbehavior in the NSFnet backbone and regional networks and, as proposed, is expected to continue in future. Joint experiments are expected using the T3 nodes of the NSFnet Backbone network in collaboration with Advanced Network Services. Other collaborative effort[1] is expected to continue with the NIST Time and Frequency Division, Norwegian Telecommunications Administration Research Establishment, DARPA Computer Systems Technology Office (DARTnet), US Naval Surface Weapons Center (SAFEnet and NGCR Program), US Coast Guard Electronic Engineering Center and several other universities, including MIT Laboratory for Computer Science and University of Maryland Computer Science Department.

The intrinsic merit in this research lies in the ongoing improvement in algorithms and models developed for precision network timekeeping. In a continuing series of refinements, timekeeping accuracy has been improved at least two orders of magnitude by using nonlinear signal-processing algorithms, while stability has been improved to the intrinsic limits imposed by typical workstation hardware by incorporating adaptive-parameter, phase-lock techniques in the local-clock model.

3. Background

This proposal involves computer network timekeeping technology and the synchronization of computer clocks. A *local clock* is an ensemble of hardware and software components used to provide an accurate, stable and reliable time-of-day function. It includes an oscillator, clock counter and software support to provide the time in some format to the operating system and client processes. In [23] various hardware and software models based on two different operating systems are considered and detailed analyses of their operating characteristics and performance envelopes are presented.

In order that multiple, distributed computers sharing a *synchronization subnet* can synchronize their operations with each other, it is necessary that a time-synchronization protocol be provided to exchange time information and adjust the time and, in some systems, the frequency of the local-clock oscillator in response to corrections computed by the protocol. If these computers are to agree with Universal Coordinated Time (UTC), a means must be provided to synchronize the network time to UTC as disseminated by various means [8].

It may happen that the accuracy and reliability of the timekeeping system as a whole can be improved when each computer exchanges timekeeping information with two or more *peers*. In these cases the information from redundant peers connected by diverse paths can be combined to produce an accuracy better than any one of them. Techniques to accomplish this are described in [22].

1 A list of collaborators is given in another section of this proposal

Part of the effort funded by the grants have been to watch and maintain several LSI-11 Fuzzball time servers located at various NSFnet backbone network sites. These machines are the direct descendents of the original NSFnet Phase-I backbone routers. They now provide primary and secondary time synchronization for about 2,000 sites in the Internet. Maintenance includes regular monitoring and software upgrades as they become available.

Much of the original research supported by the grants is in the area of network time-synchronization technology. The most widely used tool to synchronize computer clocks in the Internet is the Network Time Protocol (NTP), which has evolved over four versions and a dozen years and is now in use in many thousands of hosts distributed over major portions of the globe. This protocol and its implementations for PDP11 and Unix systems have been significantly improved in both reliability and accuracy. Recent improvements have resulted in better fault tolerance using agreement principles and better timing accuracy for use on high speed, wide area networks. A comprehensive formal specification document for NTP Version 3 has been published as RFC-1305 [22]. Other RFCs, technical reports and papers document the error analysis, timescales and clock modelling applicable to NTP [21], [23]. A review and analysis of the algorithms and models developed for NTP have been presented in a transactions paper [20] and a detailed evaluation of its operation in the Internet given in [19].

In a collaborative effort with colleagues at University of Toronto, University of Copenhagen and University of Maryland a full-featured implementation of NTP Version 3 for Unix has been completed and is now in operation in many sites from Australia to Europe and South Africa. This implementation is equipped with comprehensive measurement facilities enabling the routine collection of timekeeping data such as clock offsets and one-way delays. In many cases these data have been processed to reveal idiosyncratic behavior of the transmission links and routers in various paths in the Internet. Timekeeping data such as this serve as delicate instruments for the detection of oscillating routes, low-level loss rates and queue-management problems.

In addition to the work specifying and implementing NTP, a continuing program to enhance timekeeping measurement accuracy has been pursued. This involves routine comparison and calibration of cesium clocks, GPS satellite receivers and LORAN-C radionavigation equipment, as well as the calibration of NTP mechanisms with respect to national standards. Much of the expensive precision timekeeping equipment was provided by the US Coast Guard and DARPA Computer Systems Technology Office at no cost to the project. In an effort to develop low-cost alternatives to this expensive equipment, a computer-controlled LORAN-C timing receiver has been constructed and evaluated. It is documented in a technical report listed in the bibliography of this proposal.

2. Project Description

This is a proposal to the National Science Foundation Division of Network and Communications Research and Infrastructure (DNCRI). It proposes efforts advancing the state of the art in computer network synchronization and timekeeping, experiments and measurements involving the NSFNET backbone network and maintenance of certain NSFNET backbone network timekeeping services. It requests funds for one graduate student and three months of the Principal Investigator's time over three years, plus a nominal amount for travel and construction costs for designated interface devices.

The work proposed complements work supported by current and previous NSF grants, DARPA contracts and US Navy contracts listed elsewhere in this document. It continues a program of research, development and evaluation of Internet timekeeping technology now in use by thousands

Advances in Computer Network Timekeeping

National Science Foundation
Division of Network and Communications Research and Infrastructure

Proposal

David L. Mills
Electrical Engineering Department
University of Delaware
22 September 1992

Project Summary

This project continues previous research in network time-synchronization technology and proposes new algorithms adapted for use in high speed, wide area networks including supercomputers and superscalar workstations. The technology makes use of the Network Time Protocol (NTP) widely used now in the Internet, together with engineered modifications designed to improve accuracy in high speed networks, including SONET and ATM, expected to be widely deployed in the next several years. Specific applications benefiting from this research include multicast topologies, multimedia, real-time conferencing and management of distributed, real-time experiments.

The project also involves the design and construction of precision timekeeping devices that provide improved timekeeping accuracy and stability consistent with the capabilities of the time-synchronization technology. The effort includes the design, construction and evaluation of relatively inexpensive precision timing receivers based on the LORAN-C radionavigation system, as well as an interface for a commonly available workstation. Using the improved technology and these devices, it is expected that reliable distributed synchronization can be improved to some tens of microseconds in modern RISC-based workstations.

Finally, the project involves the conduct of experiments designed to evaluate the success of the research and assist technology transfer to computer manufacturers and network providers. A joint effort is proposed with Advanced Network Systems involving upgrade of the existing T3 router software in order to achieve end-system timing accuracies better than one millisecond.

1. Results from Prior NSF Support

Previous NSF grants have supported a graduate student and a modest amount of travel, equipment upgrades and maintenance. The travel funds were used to attend meetings of the Internet Research Steering Group, End-to-End Task Force and to provide assistance and advice on topics related to the Internet infrastructure. Results from these activities led to the publication of several papers, technical reports and request-for-comments (RFCs) listed in the bibliography of this proposal. Contributions related to infrastructure include papers on the NSFnet Phase-I backbone network and the Fuzzball networking software used by its nodes. Other contributions have been in the areas of authentication, routing and monitoring mechanisms.