Scalable, High Speed, Internet Time Synchronization

Advanced Research Projects Agency
Contract DABT 63-95-C-0046

Quarterly Progress Report
1 December 1997 - 28 February 1998

David L. Mills
Electrical Engineering Department
University of Delaware

1. Introduction

This report covers the work done in support of the DARPA Information Technology Office program in computer networking. Contributors to this effort include Prof. David L. Mills, graduate students Qoing Li and Robert Redwinski, and undergraduate student Douglas Miller. The project continues previous research in network time synchronization technology jointly funded by DARPA, NSF, US Navy and US Army. The technology makes use of the Network Time Protocol (NTP), widely used 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, cryptographic systems, and management of distributed, real-time systems.

Recent projects reported in papers, technical reports, project reports and technical memoranda include advances in precision timekeeping technology, improved clock discipline algorithms, and engineered security models and protocols for scalable, distributed server systems. Past projects include laboratory demonstrations using advanced DSP technology for LORAN-C and WWV timing receivers, as well as an optimum matched-filter SITOR receiver/decoder. Software developed with joint funding includes the NTP Version 3 implementation for Unix and Windows and a set of precision-time kernel modifications for major Unix workstation manufacturers. Finally, the joint projects involve the conduct of experiments designed to evaluate the success of the research and assist technology transfer to computer manufacturers and network providers.

2. Network Time Protocol Version 4

Our work in the design and implementation of the NTP Version 4 continued throughout the quarter; however, the departure of graduate student Ajit Thyagarajan left his work on autonomous configuration almost but not quite complete. On the other hand, implementation of the new security model and authentication scheme called *autokey* is now complete with an exception noted later. Initial experience and testing has confirmed the designs of both the autonomous configuration and autokey schemes operate as designed, are robust and efficient, and relatively simple to implement. In addition, the experience confirms both schemes should be adaptable to other real-time, distributed protocols with ubiquitous access models.

2.1 Security Model and Authentication Scheme

An interesting consequence of the implementation experience was the decision not to implement the scheme proposed by Steven Kent of BBN, which involves a stateless server generating a private key from the client IP addresses and a private value. In this scheme the server uses a hash of these values as the key value and shares it with the client over a secure channel. While the scheme is relatively fast, requires no persistent state at the server, and is easily implemented, the requirement for a secure channel is a significant drawback to its flexibility.

As described previously, the autokey scheme generates a key list by repeated hashing of a random server private value. The server uses this list in reverse order; the clients verify the identity of the server by checking that the hash of the current key equals the key most recently used. The autokey scheme was originally intended for use only in multicast modes, where clients do not regularly contact the server; however, it turned out to work as well for other NTP operating modes, including client-server mode and the symmetric modes. In particular, its use in these modes makes it virtually impossible to spoof a client request or server reply, since not only the server is authenticated by the autokey scheme, but the absence of the client private value makes a replay attack most improbable.

Implementation of the autokey scheme raised a number of issues in the area of local key management. Previously, keys were read from a file designated for that purpose, but with access permitted only to root. In the autokey scheme, keys are always generated as a random sequence and have an explicitly controlled lifetime. The key management infrastructure was rebuilt to fit this model, which required strictly policed lifetime enforcement, a new random sequence generator and provisions to avoid collisions when large numbers of associations are involved, as is the case with many public servers.

In order to preserve backwards compatibility, the space of key identifiers is partitioned into the symmetric-key space, where the key identifier has values less than 65,536, and the autokey space, where the key identifier has values greater than this. The original authentication scheme uses the symmetric-key space, where values are predetermined by bilateral agreement. The autokey scheme uses randomly generated key identifiers and keys. An interesting consequence of this model is what to do if a new random value happens to collide with an existing one which is still in use and what to do if the new one happens to have a value in the symmetric-key space. When a new value happens to be less than 65,536, the generator rolls again; while, if a collision occurs, the generator terminates the current key list. When the list is exhausted, the generator tries again with a new seed determined from the system clock.

While the autokey implementation is complete in the sense that it does generate, verify and manage keys used to generate packet cryptosums, there is still the matter of securely verifying the source of a received packet does in fact possess the private value used to generate the key list. In the original design, this was to be accomplished by a mechanism outside the scope of NTP, possibly a generic timestamping service involving certificated signatures. However, the folks at Coastek InfoSystems announced plans to market such a service which itself would require NTP as the delivery vehicle. This opened up the possibility of a collaboration in which the timestamping service would be integrated with NTP in the form of a certificated signature included in the NTP packet itself. After initial discussions, a revised format for the NTP packet header was evolved and published as an Internet Draft [2]. The format is backwards compatible with the existing formats in both authenticated and non-authenticated modes. It provides a variable-length extension field between the end of the NTP header and beginning of the Message Authenticator Code (MAC) field. It is now supported in the current NTP Version 4 distribution and used to carry the encrypted server private value used in the autokey scheme.

The Coastek timestamping service design calls for a certified signature, which is placed in the extension field; however, the exact format for these data have not yet been finalized. Appropriate hooks have been made in the current code which will support the required code when a final design has been evolved. It is expected to use the RSAREF cryptographic library in the US or its equivalent in other countries.

2.2 Autonomous Configuration

Perhaps the most valuable feature of the NTP Version 4 design is its capability to automatically organize and maintain the NTP timekeeping subnet. The design approach, rationale and mathematical foundations are to be published in Mr. Thyagarajan's dissertation, which is not yet complete. However, the intended implementation is substantially complete and usable in its present form.

As described previously, the scheme uses an expanding-ring multicast search to discover nearby servers, then configures the multiple-stratum client-server tree using an add/drop greedy heuristic designed to minimize the NTP synchronization distance (roughly equivalent to expected accuracy) under degree and distance constraints. The design spreads the load equally among the set of available servers, while at the same time insuring a high level of accuracy, redundancy and diversity.

In initial tests, the existing code operated correctly to discover new servers in both the multicast and manycast modes supported by the design. However, for the scheme to be successful in wide deployment, it must operate with cryptographically authenticated servers. However, initial testing turned up a significant problem with the Unix sockets implementation and NTP software interface. Specifically, the problem stems from the fact that the local socket IP address may not be known until the first packet from the destination arrives. This makes it impossible for a multicast server, for example, to construct the autokey session key, since it depends on the IP source and destination addresses.

In some cases workarounds have been crafted in which the first packet sent always fails the authentication test at the client, but the client is allowed to mobilize an association and return a packet, which then binds the server socket address. Subsequent packets will have the correct session key, so the client eventually authenticates as expected and the association continues. However, in some cases involving multi-homed hosts and routers, this scheme does not work correctly. The best remedy for the problem would be to overhaul the sockets interface, which may require a significant code expansion. Investigation of this issue will continue into the next quarter.

3. Global Survey of NTP Servers and Clients

Over the years a number of surveys have been conducted to estimate the total population of NTP servers and clients and the quality of service. The latest survey discovered some 230 primary (stratum-1) servers and over 34,000 servers and clients at higher strata. The survey, documented in a recent paper [1], presents the time and frequency error characteristics of typical servers and clients. It also revealed such statistics as the average number of servers and clients operating at each stratum level, the number operating in broadcast/multicast modes, and the number operating in an unsynchronized condition.

4. Presentations

5. Plans for the Next Quarter

Our plans for the next quarter include continued testing and refinement of the NTP Version 4 protocol model, specification and implementation. Specifically, we plan to resolve the problems with the Unix socket interface mentioned earlier, so that the autoconfigure feature is really useful. In addition, we plan to continue the collaboration with Coastek InfoSystems in the design and implementation of the cryptographic certification algorithm. The daemon is to be tested first in the research net, then the DARTnet/CAIRN community. As the extensions are backwards compatible, the new features can be activated and tested in regular operation without impacting current users.

6. Publications

1. Mills, D.L., A. Thyagarajan and B.C. Huffman. Internet timekeeping around the globe. *Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting* (Long Beach CA, December 1997). URL: See www.eecis.udel.edu/~mills/papers.hrml.

Abstract

This paper describes a massive survey of Network Time Protocol (NTP) servers and clients in order to discover statistics of time and frequency errors, as well as determine the health and welfare of the NTP synchronization subnet operating in the global Internet. Among the conclusions of the survey are that most NTP clocks are within 21 ms of their synchronization sources and all are within 29 ms on average. However, additional errors up to one-half the roundtrip delay are possible, but relatively rare. There is, however, a disturbing incidence of improperly operating servers and misguided server configurations.