Scalable, High Speed, Internet Time Synchronization

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

Quarterly Progress Report
1 September 1996 - 30 November 1997

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 for-

mats 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. SNMP project

As described previously, a project was started to develop a MIB for NTP. This is an issue which has been dormant for some years. Comprehensive provisions for remote monitoring and control of NTP servers and clients were included in the original NTP Version 2 software, which appeared

well before SNMP became available in any form. Prof. Adarsh Sethi of the Computer and Information Sciences Department volunteered his time and that of his graduate student to specify a MIB and implement prototype software to provide SNMP support usable by available management programs. A preliminary MIB specification has been completed and extensions to available SNMP agent software has been completed.

The new MIB and SNMP agent has been described in a technical report [1]. It is designed to operate as a SNMP proxy agent to a manager program running in another computer. However, in order to avoid a considerable rewrite of the existing NTP monitoring code, the proxy agent communicates with the NTP daemon using the native NTP monitoring protocol. The proxy agent thus translates SNMP requests into NTP requests and sends them to the NTP daemon, which in principal may run in some other computer. The proxy agent then translates NTP monitoring data received from the NTP daemon to SNMP format and forwards it to the manager program.

In the final version to be developed, the proxy agent software must be embedded in the native SNMP support for each operating system. The preliminary version uses available SNMP agent software for this purpose. Since it is not practical to modify software distributed with current workstations, and the IETF has not yet come to agreement on how to interface variables resident outside the kernel, full deployment of the NTP MIB within the standard SNMP must await resolution of these issues.

4. Presentations

Prof. Mills gave an invited talk at the DIMACS Workshop, held on 27-29 October 1997. The abstract for that talk is given below.

Mills, D.L. Wrangling a large herd of Internet clocks. DIMACS Workshop, Rutgers University, October 1997 (invited presentation). URL: See www.eecis.udel.edu/~mills.

> The Network Time Protocol (NTP) is allegedly the longest running, continuously operating, distributed application in the Internet. The first packet flew in 1979 and the protocol itself hasn't changed much since then. However, the algorithms have evolved in dramatic ways and reflect the chaotic place the Internet has become. With over 100,000 servers and clients ticking in the Internet of today, NTP is not only a good timekeeper, but also a highly useful watchtower to gauge the robustness, stability and utility of the Internet itself.
>
> This talk will focus on lessons learned (and relearned) in the analysis, design, implementation and operation of the various algorithms evolved for NTP. These include those which cryptographically authenticate valid servers, sift the messages from possibly many of them to find the most reliable clique and the best time, and nudge the client clock as close to true tick as possible. With modern workstations and fast local networks, NTP can usually nudge the clock closer than a few tens of microseconds.

There were three specific issues explored in the workshop context:

1. Robustness of the synchronization algorithms to many and varied kinds of failure, Byzantine, malicious and otherwise. Our approach is based on diverse network paths, redundant servers and a suite of intricately crafted algorithms.

2. Autonomous configuration, that is, the ability to automatically and with minimal manual intervention configure the (large) timekeeping network with respect to quality metric and overhead constraints, and to reconfigure as required due to failure or network congestion. Our approach is based on Internet multicasting, anycasting and manycasting, together with engineered drop-add heuristics which manage the load on the servers and network.

3. Authentication, that is, the ability for clients to independently authenticate the servers using both public-key and private-key cryptography. Our approach uses automatically generated and managed keys with controlled lifetimes and engineered algorithms designed to avoid loss of accuracy due to encryption jitter.

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. Sethi, A.S., H. Gao, and D.L. Mills. Management of the network time protocol (NTP) with SNMP. Computer and Information Sciences Report 98-09, University of Delaware, November 1997, 32 pp. URL: www.cis.udel.edu/~sethi/papers/97/ntp-mib-tr.ps>(PostScript)

2. Mills, D.L., T.S. Glassey, and M.E. McNeil. Coexistence and interoperability of NTP authentication schemes. Internet Draft draft-mills-ntp-auth-coexist-00.txt, University of Delaware and Coastek InfoSys, Inc., November 1997, 8 pp.

   Abstract

   This memorandum describes extensions to the Network Time Protocol (NTP) version 3, described in RFC 1305 [MIL92], and the Simple Network Time Protocol (SNTP) version 4, described in RFC 2030 [MIL96a], to create a framework for interoperability and coexistence of various cryptographic signature and authentication schemes that have been suggested as enhancements to NTP/SNTP [MIL96b]. Without describing any particular approach to authentication in detail, a framework is hereby established for these schemes and methods to coexist and interoperate on a single network or Internet.

   NTP v3, as specified in RFC 1305, provides support for a method of symmetric-key authentication. This model requires independent, out-of-band, distribution of cryptographic keys, with the associated risk of compromising those keys during distribution. This process is too costly and too great a security risk for universal application.

   The NTP protocol extensions described here allow for the creation of certified time from a certified and authenticated time source, for use in portable commercial-grade trust, non-repudiated logging, cryptographic synchronization, and other critical network management

services. Newer cryptographic techniques involving public-key certificates and authentication (see the series of Internet-Drafts by the Public Key Infrastructure X.509 [PKIX] working groups of the IETF) can elevate timestamping to a trustable resource.