Survivable, Real Time Network Services

Defense Advanced Research Projects Agency
Contract F30602-98-1-0225, DARPA Order G409/J175

Quarterly Progress Report
1 April 2002 - 30 June 2002

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 student Harish Nair and undergraduate research student John Conner. MS student Tamal Basu has graduated and taken a position at Acorn Networks. Phd student Qiong Li has graduated and taken a position with Phillips Research.

The project continues research in network time synchronization technology funded by DARPA, US Army ARL and NASA/JPL. 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. Specific applications benefiting from this research include multicast topologies, multimedia, real-time conferencing, cryptographic systems, and management of distributed, real-time systems and sensor networks.

This quarterly report is submitted in traditional report form on paper. As the transition to web-based information dissemination of research results continues, almost all status information and progress reporting is now on the web, either on pages belonging to the principal investigator or to his students. Accordingly, this and future progress reports will contain primarily schedule and milestone data; current status and research results are reported on web pages at www.eecis.udel.edu/~mills in the form of papers, technical reports and specific briefings.

2. Timekeeping in the Interplanetary Internet

For NASA/JPL we are working on issues involved with timekeeping in the Interplanetary Internet, in particular the upcoming Mars missions. Harish Nair is poking through the JPL SPICE library and documentation which may be necessary to predict position and velocity vectors describing moving objects in space and on the surface of Mars. This will require procurement of epheremis data in some form or another, perhaps assisted by range and range rate measurements by NTP itself. Some of this technology is as old or older than the DARPA SATNET program some 25 years ago, but at that time the earth stations really had only to measure the roundtrip time and the satellites stayed pretty much where they were told.

2.1 Approach and Discussion

Our model is that the timescales for all platforms, be they in space or on the surface, run at a constant rate relative to atomic time (TAI). This creates somewhat of a problem in that for an event

like a supernova explosion the flash should be observed by all platforms consistent with the quasi-planar wavefront from the source. This requires all platforms to calculate the offset relative to barycentric time (TDB) with respect to their particular position in space, which changes continuously. Thus, only the local clock rate is disciplined and the apparent time relative to TDB is calculated only when needed.

The scheme of choice goes something like this. Upon transmitting a NTP message, a client determines the apparent TDB time, position vector and velocity vector from local ephemeris. For symmetry, as required in NTP symmetric mode, These values are included in an extension field. The message includes this extension field along with the transmit timestamp, which when the clock is not synchronized may not agree with TDB time.

Upon receiving this message, the server determines the apparent TDB time, position vector and velocity vector from local ephemeris. For stateful servers, these values are saved for the next transmission opportunity. When transmitting the next NTP message, the server again determines the apparent TDB time, position vector and velocity vector from local ephemeris. The message now includes three extension fields corresponding to the position and velocity vectors for each of the three timestamps included in the NTP header.

Upon receiving this message, the client again determines the apparent TDB time, position vector and velocity vector from local ephemeris. The client now has four timestamps, as in ordinary NTP, plus four sets of position and velocity vectors, one corresponding to each timestamp. There are two unknowns, time offset and roundtrip propagation delay, to be determined, but these depend on the actual ephemeris time, which is not known at this point. An appropriate solution involves an iterative approach where the apparent time and delay are calculated from the server values, assumed accurate, and the apparent client values. New apparent position and velocity values are determined from the ephemeris and the process iterates.

Complicating the above, the residual clock frequency offset may introduce considerable error if the time between updates is relatively long, as would be expected during communication opportunities between Earth and mission spacecraft. After a few measurements the frequency can be disciplined in the usual way, but this affects the position and velocity vectors and residuals with respect to the ephemeris. What makes frequency-induced errors more nasty is that the frequency may fluctuate due to spacecraft thermal cycles and power management. Assuming primary servers on Earth together with ephemerides of the transmitter location, the above scheme continues to refine the residuals and develop global time. Kalman filters or ARIMA methods might be a good tool to deal with the residuals and steer to the best time.

There is some concern that the expense of these calculations, both in processor cycles and thermal management, may not be justified in all cases. For instance, NTP between Mars orbiters and the surface is no different than NTP between Earth orbiters and the ground. In fact, NTP has flown in space before on an AMSAT satellite where the embedded Intel processor ran the same code as used on Earth. This assumes the satellite doesn't move very far during the roundtrip propagation time for the NTP message and reply. If finer correction is required, orbital elements could be derived from radio rise and set times and corrections computed on the fly.

There has been some discussion on what the Mars orbiters can do with respect to antenna orientation. There is a limited fuel supply to point the antenna to Earth and it may not be a good idea spending fuel to point it at other orbiters or the ground in order to exchange NTP packets. Also, it

is not likely the orbiters can communicate with each other using an omnidirectional antenna and low power, at least most of the time. However, omnidirectional antennas would seem to be the choice when communicating with surface platforms. Assuming the surface platforms can discipline the local clock to some degree of precision, the surface clock could be used as a flywheel to synchronize orbiter clocks as they pass over the platform.

2.2 Future Plans

The general plan is to evaluate the above approach in detail, especially the impact of the required resources for processor cycles and power management. Of particular concern is the magnitude of frequency wander due to normal thermal housekeeping functions.

It is our eventual plan to implement a suite of algorithms which transform data such as the extension fields and timestamps described above to a TDB reference space, so that the existing NTP algorithms can be used without modification to discipline the local clock. We plan to sift through the SPICE library looking for tools and algorithms useful for this purpose. We plan eventually to test the algorithms using the NTP simulator described in the following section.

3. NTP Simulator

The various algorithms used in NTP have been evaluated using a special purpose simulator called ntpsim. The simulator uses algorithms somewhat simplified from those in the reference implementation, but behave very nearly the same under nominal conditions with typical network delay jitter and oscillator frequency wander. However, the NTP algorithms have grown in complexity over the years and some quirks of the reference implementation have cropped up from time to time. This has been most apparent when operators, fearful of dreaded backward time steps, have insisted that clocks always be slewed, rather than stepped, even if they are initially in error by a considerable amount, like a week.

The ntpsim algorithms include a number of heuristic defenses against low probability events, such as transient spikes, mode changes, “clockhopping”, server restarts and so forth. In principle, it would be possible to incorporate these features in ntpsim; however, this would require a good deal of effort and require verification that the features work the same in both simulation and practice.

3.1 Progress

Harish Nair looked into the question of whether the existing reference implementation could be modified to function as a simulator driven by a programmed script or the stochastic source model developed in previous research. This turned out to be easier than first imagined and, in fact, a preliminary version has been implemented in the NTP development version. The simulator uses the same code as the daemon, but surrounds it with a special purpose, discrete event simulator. The only changes are in the input/output, timer, clock reading and clock discipline code.

The simulator can be driven by explicit impulse generators selected on the command line or by a synthetic noise generator which closely duplicates the effects of network jitter and oscillator wander. In its present form, the simulator supports only one association where the synthetic source acts like a server and drives the NTP code as a client.

We expect to use the simulator to resolve some difficult scenarios, especially those which play out in a matter of days, like when the poll interval ramps up to a day or more and the clock discipline is in a frequency-lock mode. If the frequency wander suddenly increases, the poll interval must back off to a lower value in order to track the relatively rapid frequency changes. The code that does this is largely cut-and-try and yet remains to be optimized.

Another area needing simulation is the behavior of the algorithms when step is disabled and the clock discipline is forced to slew. A state machine has been implemented to quickly calculate major frequency and time changes and avoid long delays while the clock discipline slowly amortizes these changes. This causes big trouble when the frequency and frequency change clamps kick in as required by correctness assertions. The general behavior is best described as a unruly pinball machine which almost always (we think) eventually subsides to a stable regime. Problem is, in real life the pinball game can last a week. The simulator can run the course in a few seconds.

3.2 Future Plans

While a few minor touchups and features need to be done, the simulator is essentially complete and will shortly become available in the public software repository. As mentioned above, we expect to use the simulator for proof of concept testing in the Interplanetary Internet time synchronization project.

4. Autonomous Authentication

Autokey is a cryptographic protocol for the authentication of servers and clients in a hierarchical network. It uses certificates, timestamped digital signatures and reverse hashes to minimize clogging vulnerabilities, minimize resource exposure to public key computations and reduce packet sizes. The original intent of the design was for NTP; however, the same defense strategies can be applied to other services and in non-hierarchical network topologies, including sensor networks. This is the intent in DARPA and CTA supported research.

4.1 Progress

There have been several changes and additions in the Autokey protocol specification and reference implementation. Certificates are now managed in a LRU cache shared among all associations. Certificates can be used to sign and verify other certificates and regular garbage collection removes expired or revoked certificates. The ASSOC request can now retrieve a certificates by subject name. A new message SIGN has been added with which a client can ask that its certificate be signed by another server acting as certificate authority or introducer.

The key management procedures have been simplified and regularized. Key refreshment is now entirely automatic and can be driven by a script executed from a Unix cron job. The script rolls new keys and host certificates, including identity strings, and restarts the daemon. Everything else is automatic, even with symmetric peers which used to take a very long time to restart. Keys no longer have to be kept in a shared directory on NFS servers, so the root passwords of the clients do not have to be known to each other or the server.

The X.509 certificate format has been updated to Version 3, so that additional information can be included in a certificate extension field. These fields are intended to be used for related credentials

such as identification strings and multi-component capability metrics. Identity strings are intended as a unique identifier that binds credentials to the certificate. Two candidate schemes have been suggested, one based on symmetric keys and the other on the Schnorr identity scheme. While neither has been implemented, the intent at this time is to provide a framework in which a suitable scheme can be embedded, but without restricting the scheme to any particular design.

An simple symmetric key identity string scheme could work as follows. Select a particular private key which is shared by all members of a particular NTP group, such as a campus or department. Concatenate the value of this key with the subject name on the certificate and compute the MD5 message digest, then Include the result in the extension field before signing. Clients and servers can then verify the corresponding hash matches the extension field and discard bogus requests before wasting processor cycles. With this scheme the selected key is not revealed and intruders cannot manufacture an acceptable certificate. Defense against replay attacks is provided by the Autokey protocol itself.

The Schnorr identity scheme can be adapted as follows. Before deploying, every sensor is loaded with the public parameters and the normally private exponent. The exponent is public to the members of the group and never revealed outside the group. Hash the values with MD5, then Include the result in the extension field before signing. The hash can be used to verify the certificate is associated with the expected group and to avoid duplicitous signature calculations. Again, the Autokey protocol deflects old replays.

The client proves identity to the server and the server proves identity to the client using a three-way handshake where each entity generates a nonce and the other performs a computation on the nonce using the private exponent. This requires an extension to the Autokey protocol which has yet to be designed.

The single most worrisome vulnerability using strong cryptography in sensor networks is a hazard where an intruder runs down the battery by forcing needless signature encryptions and decryptions. The Autokey protocol is specifically designed to minimize such hazards. In order to improve the resilience of the protocol and harden it against clogging attacks, the protocol was subjected to yet another rigorous vulnerability analysis in the hope of simplifying the state machine, improving the error checking and making the protocol specification more regular and readable. The analysis resulted in much simpler and more direct header processing steps in both the protocol specification and reference implementation. All extension fields now go through the same data extracting and error checking procedures and the code is smaller, more straightforward and readable.

The analysis also suggested more effective defenses against clogging attacks. The only conditions under which an attack might be successful are while the certificate trail is being explored. Once the certificate is verified and the proventic bit is lit, it is not possible to provoke a decryption attack where the victim client consumes precious processor cycles. On the other hand, there are two remaining vulnerabilities involving encryption attacks. One is when requesting the cookie, in which the client presents its public encryption key and the server encrypts the random cookie and then signs the field. The stateless server has no defense against clogs of this type other than gapping; that is, dropping packets that exceed some preset rate. The client will of course drop the response, since the packet will fail the loopback check.

The other hazard is when the client presents a certificate to be signed. Presumably, the victim server would have to verify the request, sign the certificate and finally sign the extension field. There are two ways to deflect such attacks. The client includes an identification string in the X.509 extension field. In the scheme suggested above the identity check does not involve signature decryption, so a bogus certificate is detected and deflected. Second, the SIGN request is valid only when the client has synchronized to proventic time. This means that the field signature is valid, even if the server can't tell if a replay. While yet a victim of a decryption attack, at least the server will not have to verify the certificate, sign the result and sign the response field.

4.2 Future Plans

The Internet Draft on the Autokey protocol specification has been under major revision as well. It has been submitted to the STIME task force for final review. We expect it will be issued for last call by the IETF and launched on the standards track. An issue remaining for future work is a more refined scheme for proving identify using an X.509 extension file. The immediate issue is to find a scheme where compromised members can be expunged from the group and new ones be added without affecting working members.

5. Autonomous Configuration

Manycast uses IP multicast to broadcast an invitation for potential servers to reveal themselves. This is current work supported by DARPA. Clients receiving responses to these invitations mobilize an association and, after a delay to groom the samples and suppress Byzantine falsetickers, outlyers are shaved from the population until only a maximum (three) are left. The scheme may appear straightforward, but it must be intricately engineered to minimize implosion hazards, repair damage when a server is lost, minimize network traffic and insure robust authentication.

Manycast is designed to operate in conjunction with Autokey in a completely fire-and-forget deployment mode. For a sensor network application, all sensors are configured as both manycast server and manycast client mode. Some sensors with more powerful radios are configured as primary servers and the network automatically nucleates around them. As each new association forms, it cranks the Autokey protocol. When the protocol completes there is no additional packet or processing overhead other than an occasional twitch to generate a new key list.

5.1 Progress

The new certificate provisions work very will with Manycast. The initial volley results in certificates from all servers and presents a rich mix of certificates that can be used to sign other certificates. In fact, unless specifically directed otherwise, each of a sizeable number of clients will accumulate all the certificates necessary to proventicate the trail to the primary servers. This makes an inviting target for multi-component trust metrics as described in the next section.

Several minor changes were made to the Manycast protocol and reference implementation to improve and smooth the response to transients, such as when a server refreshes its server seed or rolls a new certificate and restarts the protocol.

5.2 Future Plans

The Manycast algorithms and decision metrics are considered mostly mature. An important area that needs more work is a scheme to automatically select a stratum appropriate for the expected accuracy and available resources of the nearby server population. We expect to mine Ajit Thyagarjan's dissertation for applicable heuristics. It is likely that some sort of whisper campaign protocol will be necessary in order to balance the load among the available servers.

Load leveling can be implemented using an extension field which carries, for example, a list of the current servers mobilized by the Manycast client. A LRU list of recently heard servers is already available in the NTP reference implementation and used for access controls. Manycast servers can use the LRU list and combined extension field lists to compute a decision threshold for each server. Each server compares a random roll to its threshold to determine whether to respond to a client request. Details remain to be worked out.

6. Multi-Component Trust Metrics

This work is the primary area of interest in the CTA funded activity. The problem is to determine from various security related variables the level of trust accorded to some specific function. The approach involves the use of classic pattern analysis and classification (PAC) methods and defined operations to establish a trust vector where each component of the vector corresponds to the degree of trust for a security related function.

6.1 Approach and Discussion

PAC methods were first developed well over two decades ago. A set of feature vectors record a particular set of measurements for some experiment outcome. The vectors are members of an n-dimensional vector space, where each component is associated with a dimension. Sets of decision planes are devised to classify the space of acceptable and unacceptable outcomes for some defined function. The distance between a feature vector and a defined target vector can be defined using any of several measures. This distance will later be used as a trust vector component.

An example having nothing to do with cryptography, but illustrating the technology, is derived from original work done to decode manually sent Morse code some 25 years ago. A Morse code element consists of a mark (tone) followed by a space (silence). Marks can be dot or dash intervals; spaces can be element, character or word intervals, giving a total of six possible outcomes represented on a space of two dimensions: mark and space. A set of three decision planes, lines in this case, can be defined to classify a feature vector as one of these six elements and to determine a distance measure.

A similar approach can be used for cryptographic applications, although the decision planes are presumably fixed by design and not probabilistic. For the purposes of discussion, a peer-peer scenario will be assumed, although the scenario could just as well be client/server or broadcast. Each peer has a feature vector associated with some subset of the other peers. The vector includes a number of low level values, such as how many peer certificates have been verified, how many recommendations have been received from other peers, whether the peer identity has been verified (for instance, by the Schnorr scheme), whether the peer time has been synchronized, how

many clogging attack packets have been detected in the last minute, and so forth. Some of these components may age in the form of an exponential decay.

A trust vector for a particular peer defines a set of trust metrics, where each metric is associated with evidence that the peer can reliably perform some cryptographically protected function, such as sign a certificate request, read the clock, fire a missile, break radio silence, arm a booby trap, etc. Trust metrics for each of n functions are computed from the feature vector for each peer. In the traditional PAC model, a set of decision planes is defined over n-space for each function and this classifies the particular yes/no and don't-care outcomes for the function.

Some feature vector components may not be ideally suited to the traditional PAC model. Consider the case of geographic position or distance from a neighboring sensor and the uncertainty of the actual value. A handy interpretation may the distance in n-space from the particular feature to a defined outcome vector representing the uncertainty or distrust in the measurement.

From the above discussion it should be clear that the PAC operations and defined cryptographic functions produce a different trust vector for each peer. We need an operation that takes as argument the peer trust vectors and produces what might be called an inherited trust vector. Each component defines the trust involved in performing the associated cryptographic function. For instance, one of these trust components measures the confidence that the peer can legitimately sign a certificate request. Another might measure the likelihood an intruder is among the peer population.

The trust vector itself may be useful to other peers, perhaps contained in a protocol message. The components of trust vectors received from a peer can be combined in the feature vector for that peer and included in the PAC calculations. This provides a transitive mechanism where peers inherit trust from other peers and so on. Since such mechanisms typically involve a product of probabilities, for example, it may be appropriate to represent trust components as logarithmic values.

6.2 Future Plans

This activity has just been started and is not yet fully staffed. The discussion in this section is intended as marching orders for a grad student yet to be identified.

7. Infrastructure

The Windows PC used for utility word processing and multimedia presentation authoring is getting very old. A replacement has been purchased and installed in the laboratory. At the moment, there is no real alternative to Windows and Office, and these products cost as much as the hardware. The T1 line for the CAIRN tie circuit is being rerouted via a different carrier.

8. Publications

All publications, including journal articles, symposium papers, technical reports and memoranda are now on the web at www.eecis.udel.edu/~mills. Links to the several publication lists are available on that page, as well as links to all project descriptions, status reports and briefings. All publications are available in PostScript and PDF formats. Briefings are available in HTML, PostScript, PDF and PowerPoint. The project descriptions are cross-indexed so that the various

interrelationships are clearly evident. Also included are the documentation pages for various public software distributions. Links to other related projects at Delaware and elsewhere are also included on the various pages. Hopefully, the organization of these pages, which amount to a total of about 300 megabytes of information pages and reference documents, will allow quick access to the latest results and project status in a timely way.

Following is a retrospective list of papers and reports supported wholly or in part on this project and the immediately preceding project "Scalable, High Speed, Internet Time Synchronization," DARPA Order D012. The complete text of all papers and reports, as well as project briefings, status reports and supporting materials is at www.eecis.udel.edu/~mills.

8.1 Papers

1. Li, Q., and D.L. Mills. Jitter-based delay boundary prediction of wide-area networks. *IEEE/ ACM Trans. Networking 9, 5* (October 2001), 578-590

2. Levine, J., and D. Mills. Using the Network Time Protocol to transmit International Atomic Time (TAI). *Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting* (Reston VA, November 2000), 431-439.

3. Mills, D.L., and P.-H. Kamp. The nanokernel. *Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting* (Reston VA, November 2000), 423-430.

4. Mills, D.L. A brief history of NTP time: confessions of an Internet timekeeper. Submitted for publication; please do not cite or redistribute.

5. Li, Q., and D.L. Mills. Investigating the scaling behavior, crossover and anti-persistence of internet packet delay dynamics. *Proc. 1999 IEEE GLOBECOM 99 Symposium* (Rio de Janerio, Brazil, December 1999).

6. Mills, D.L. Cryptographic authentication for real-time network protocols. In: *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 45* (1999), 135-144.

7. Mills, D.L. Adaptive hybrid clock discipline algorithm for the Network Time Protocol. *IEEE/ACM Trans. Networking 6, 5* (October 1998), 505-514.

8. Mills, D.L., T.S. Glassey, and M.E. McNeil. Authentication scheme extensions to NTP. Internet Draft draft-mills-ntp-auth-coexist-01.txt, University of Delaware and Glassey-McNeil Technologies, September 1998, 9 pp.

9. Li, Qiong, and D.L. Mills. On the long-range dependence of packet round-trip delays in Internet. *Proc. IEEE International Conference on Communications* (Atlanta GA, June 1998), 1185-1191.

10. 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), 365-371.