A Computationally Efficient and Secure Time Synchronization Scheme for Wireless Applications

C. A. Barnett[+] and K. J. Ray Liu[++]

[+]Hughes Network Systems
Germantown, Maryland
cbarnett@hns.com

[++]Electrical and Computer Engineering Department
University of Maryland, College Park
kjrliu@isr.umd.edu

***Abstract*—** Wireless mobile computing devices are used extensively to access the Internet for critical applications and services, which must be secured by computationally efficient algorithms and schemes. Computationally efficient security schemes, however, require accurate time synchronization between client and application servers, which is very challenging due to its susceptibility to stochastic noise on the transmission path and to message delay attacks where an adversary can intentionally delay measurement messages in order to compromise the scheme. In statistically noisy networks, accurate time synchronization requires filtering of a large number of measurements to reduce stochastic noise. Each measurement must be secured and, therefore, computationally efficient schemes are required to facilitate mobile computing. Efficient security schemes are also required to detect message delay attacks and to authenticate synchronization message sources (e.g. multicast host server and clients). We propose a computationally efficient time synchronization scheme, which achieves accurate and reliable time clock-offset estimates from a single measurement. This approach is based on a per node quality of service scheme, which can be easily incorporated into existing and emerging Quality of Service scheduling disciplines. It is designed to reduce stochastic noise by providing per node delay guarantee for synchronization. We also propose an efficient and reliable scheme to detect message delay attack. The proposed scheme compares synchronization messages round trip delay with predefined thresholds to ensure that they fall within expected limits, otherwise they are discarded.

Index Terms—Scheduling, Time Synchronization, Earliest Deadline First, Delay Guarantee, Quality of Service

I. INTRODUCTION

The explosive growth in Internet usage and the use of wireless mobile computing devices to access the Internet have created tremendous demand for secure realtime services. These services and applications require accurate and reliable time synchronization in order to provide computationally efficient schemes. By time synchronization we mean the setting of client's clocks so that they agree at a particular epoch at a particular server. Time synchronization is achieved by measuring clock-offsets between a client and timeserver and so is vulnerable to statistical delay variations (i.e. stochastic noise) on the transmission paths between them. In the Internet, transmission paths can have wide variations in delay and reliability. As a result, accurate time synchronization requires carefully designed filters to reduce stochastic noise. The accuracy of the clock-offset estimate is related to the filter algorithm and number of measurements used. As the number of measurements increase, the clock-offset accuracy also increases.

Time synchronization schemes are also susceptible to message delay attacks, where an adversary can intentionally delay measurement messages in order to compromise the scheme. In securing time synchronization protocols, mechanisms are required to detect message delay attacks and to authenticate synchronization message sources (e.g. multicast host server and clients).

Secure and accurate time synchronization involving mobile computing devices is very challenging, especially where securing each measurement involves computationally intensive cryptographic operation. Several time synchronization schemes [3], [5], [7],[8] have been proposed to synchronize clients and timeservers. These schemes are able to synchronize clients and timeservers to within a few tens of milliseconds, using a large number of measurements to achieve the desired accuracy and reliability. Computationally intensive cryptographic schemes are currently used to secure measurement messages. Also, they cannot guarantee a clock-offset bound between application servers and clients, since time synchronization is with a separate timeserver (i.e. third party) instead of between clients and application servers. They also provide no mechanism to detect and/or mitigate message delay attacks. These synchronization schemes are, therefore, not suitable for secure mobile computing.

In this paper, we propose a client initiated time synchronization scheme to synchronize clients and their application server, where the number of measurements required for accurate time synchronization and the computational overhead involved in securing each measurement at the client are sufficiently small. We also propose a reliable and efficient scheme, to detect message delay attacks in message based time synchronization schemes. In this scheme, the synchronization message round trip delay is compared with predefined thresholds to ensure that it falls within expected limits, if not, it is discarded.

We propose a per node Quality of Service (QoS) approach, which provides delay guarantee on a per node basis. The objective is to achieve accurate clock-offset estimation with only a single measurement, which makes this approach very challenging. The design of practical QoS schemes to achieve

0-7803-7954-3/03/$17.00 ©2003 IEEE.

the desired delay guarantee at each node is also very challenging and is the subject of significant research. Our proposed QoS mechanism achieves the desired per node delay guarantee for synchronization measurement messages.

The proposed QoS based approach to stochastic noise reduction in network time synchronization can be easily incorporated into existing and emerging QoS scheduling disciplines performance [2], [6], [9], [10] [11] without degrading performance as the simulation results show.

II. QOS-BASED TIME SYNCHRONIZATION

We propose a network QoS based approach to achieving client and server time synchronization in this section. The objective of our approach is to significantly reduce the delay variance of synchronization messages on a per node basis such that stochastic noise introduced at each node is minimized. In this approach, each client generates and sends a time synchronization request to the application server when activating the service (e.g. such as joining a multicast group), or the resynchronization timer expires. After the successful completion of clock-offset estimation, the client stores the clock offset of the corresponding server for future use.

A. Clock Offset Estimation

Fig. 1 describes a practical time synchronization scheme designed to estimate the clock offset between each multicast client and server. A synchronization request and response messages are required to estimate the clock-offset between the client and server. On initiating time synchronization process, the client establishes a connection with the server, and sends an authentication message to the server, which authenticates the requesting client and sends an encrypted response, containing a stream cipher master key, to the client. The clients use the master key in conjunction with the sequence number of synchronization response to generate the actual message keys used to authenticate response message from the trusted server. The client initiates and notes the transmission time (T_1^r) of the time synchronization request, after receiving the authentication response from the server. The server receives the request and notes the reception time (T_1^s). The server then sends the source authentication coded response (i.e. MAC is computed and included in the response packet) containing the time of the synchronization request reception time (T_1^s) and the transmission time (T_2^s) of the synchronization response.

On receiving a response from the server, the client authenticates and computes the clock offset as described below if the response is from the trusted server:

$$\Delta_1 = T_1^s - T_1^r \tag{1}$$

$$\Delta_2 = T_2^r - T_2^s \tag{2}$$

where

$$T_1^s = T_1^r + \delta_1 + \varepsilon \tag{3}$$

$$T_2^s = T_2^r - \delta_2 + \varepsilon \tag{4}$$

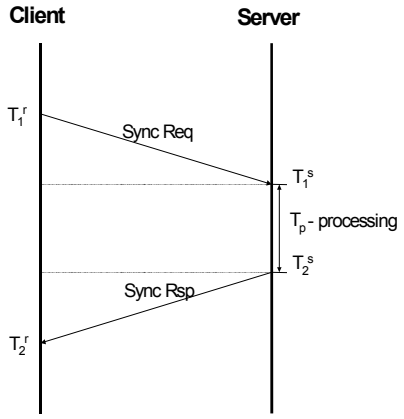

Fig. 1: Client Initiated Time Synchronization Scheme

The terms δ_1 and δ_2 are the forward and reverse path transmission delays, respectively, and ε is the clock offset estimate. We formulate the forward and reverse path transmission delays as follows:

$$\delta_1 = \sum_{j=1}^{n+1} \rho_j^1 + \sum_{k=1}^{n} d_k^1 \tag{5}$$

$$\delta_2 = \sum_{j=1}^{n+1} \rho_j^2 + \sum_{k=1}^{n} d_k^2 \tag{6}$$

where ρ_j^1 and ρ_k^2 are the transmission rate of the j^{th} link in the forward and reverse directions, respectively; and d_1^k and d_2^k are the nodal delays in the forward and reverse direction at the k^{th} node, respectively.

Using equations (1) – (4), we arrive at the following:

$$\begin{aligned} \Delta_1 - \Delta_2 &= T_1^r + \delta_1 + \varepsilon - T_1^r - T_2^r + T_2^r - \delta_2 + \varepsilon \\ &= \delta_1 - \delta_2 + 2\varepsilon \end{aligned} \tag{7}$$

Substituting (5) and (6) into (7) we arrive at the following formulation for the clock-offset estimate:

$$\begin{aligned} \varepsilon &= \frac{\Delta_1 - \Delta_2}{2} - \frac{\delta_1 - \delta_2}{2} \\ &= \frac{\Delta_1 - \Delta_2}{2} - \frac{1}{2}\sum_{j=1}^{n+1}\left(\rho_j^1 - \rho_j^2\right) - \sum_{k-1}^{n}\frac{d_k^1 - d_k^2}{2} \end{aligned} \tag{8}$$

If the links are symmetrical in transmission rates, (8) reduces to

$$\varepsilon = \frac{\Delta_1 - \Delta_2}{2} - \sum_{k=1}^{n} \frac{d_k^1 - d_k^2}{2} \quad (9)$$

In wireless networks, the transmission links at the client and server may not be symmetrical in transmission rates and, therefore, (8) is rewritten as follows:

$$\varepsilon = \frac{\Delta_1 - \Delta_2}{2} - \frac{1}{2}\left(\rho_1^1 - \rho_1^2\right) - \frac{1}{2}\left(\rho_{n+1}^1 - \rho_{n+1}^2\right) - \sum_{k-1}^{n} \frac{d_k^1 - d_k^2}{2} \quad (10)$$

The term $\sum_{k=1}^{n} \frac{d_k^1 - d_k^2}{2}$ in represents the total clock-offset error introduced by the intervening nodes, which is an unknown zero mean random variable or stochastic noise. The goal is to reduce or bound this error so that we can estimate the clock offset ε as follows:

$$\varepsilon \approx \frac{\Delta_1 - \Delta_2}{2} - \frac{1}{2}\left(\rho_1^1 - \rho_1^2\right) - \frac{1}{2}\left(\rho_{n+1}^1 - \rho_{n+1}^2\right) \quad (11)$$

which reduces to

$$\varepsilon \approx \frac{\Delta_1 - \Delta_2}{2} \quad (12)$$

when the transmission links are at the clients and server are symmetrical in transmission rates. In cases where the node delays in the forward and reverse direction are the equal, accurate clock-offset estimate results.

We, therefore, propose an approach based on EDF scheduling which significantly reduce the delay variance on synchronization messages. As a result, the error contribution by the term $\sum_{k=1}^{n} \frac{d_k^1 - d_k^2}{2}$ tends to zero (i.e. cancellation of stochastic noise).

B. Clock-Offset Estimate Error Control by Network QoS Guarantee

The client and server clock-offset estimate error addressed in section A must be as small as possible for secure Internet applications requiring time synchronization. As a result, we propose a scheme which makes use of the Internet Engineering Task Force (IETF) differentiated service mechanism. This QoS mechanism provides QoS on IP traffic flow and has been adapted to provide the QoS mechanism for time synchronization messages. The objective is to ensure that the processing delay at each node is constant.

The IETF's mechanism provides QoS on a flow basis instead of on per connection basis via the Differentiated Service (DS) Per-Hop-Behavior (PHB) groups. Two DS groups are provided for IP QoS, Expedited Forwarding (EF) and Assured forwarding (AF). We proposed the use of AF [1] to provide the QoS mechanism for synchronization message flows since it allows for extensions for local use. AF PHB group provides forwarding of IP packets in *N* independent AF classes. Within each AF class, an IP is assigned one of M different levels of drop precedence. An IP packet that belongs to an AF class i and has drop precedence j, is marked with the AF codepoint AF_{ij} where $1 \le i \le N$ and $1 \le j \le M$. Within each AF class, IP packets are marked with one of three possible drop precedence values. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested differentiated node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value.

Currently, four classes with three levels of drop precedence in each class are defined for general use. More AF classes and/or levels of drop precedence may be defined for local use. We recommend that these classes are extended so that a AF class and drop precedence is used to indicate the desired per hop behavior for synchronization messages and the corresponding node delay.

C. Network Model

Each client generates synchronization request messages randomly and, therefore, we model the number of requests arriving at each network node and the server as a Poisson counting process. Synchronization request and response message are assumed to be the same fixed length and therefore the service time constant In addition, since the QoS is provided on a message flow basis, a single server is used in our model to process synchronization request and response messages. The synchronization process is initiated when the multicast client is first provisioned and when the resynchronization timer expires.

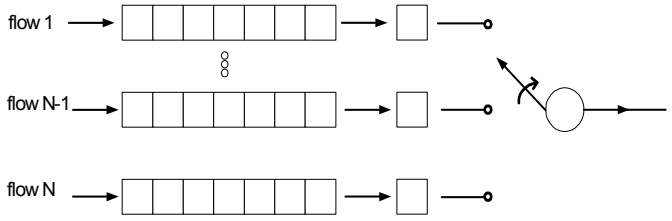

Fig. 2: Scheduling Model

Fig. 2 describes the scheduler model. In the model flow k, where $k = 1,2,....,N-1$, are service based on the selected schedule discipline to achieve desired per node or end-to-end statistics. Synchronization messages, however, are scheduled for transmission at time $t + d_i$, where t is the arrival time and

d_i is a fixed node delay (i.e. each message experience a fixed delay), and preempt other flows at or have passed their deadlines. Synchronization messages whose deadlines are passed are scheduled in decreasing order of time past their deadline. Packets in the scheduler are serviced every Δ seconds. Packet processing at each node includes a random delay component due to inter-task communication, queuing and operating system overhead. In order to mitigate the effects of this random delay on synchronization message delay, the fixed delay experienced by synchronization messages is selected to exceed the random processing delay at each node.

D. Securing Network Time Synchronization from Message Delay Attacks

Our proposed time synchronization scheme for synchronizing multicast server and clients makes extensive use of message delay measurements. This approach is susceptible to message delay attacks. An adversary can delay synchronization messages so that error introduced into the clock-offset computation exceeds the maximum allowed.

We propose a message delay security scheme, which compares the round trip delay of synchronization request, and response messages to predetermined maximum and minimum delay. If the round trip delay falls within the maximum and minimum delay threshold, it is accepted as secure. Otherwise, it is discarded. In order to establish the delay threshold, we compute the probability density function of the round trip delay of synchronization measurement at the maximum distance from the server and use it to determine the delay thresholds. The thresholds are selected be within $\pm k\sigma_T$ of the mean μ_T, where σ_T is the standard deviation. The combined message transmission delay is computed as follows using (1) and (2):

$$T_D = \Delta_1 + \Delta_2 = \delta_1 + \delta_2$$
$$= \Delta_1 + \Delta_2 + \frac{1}{2}\sum_{j=1}^{n+1}\left(\rho_j^1 + \rho_j^2\right)\sum_{k=1}^{n}\left(d_k^1 + d_k^2\right) \qquad (13)$$

Modeling the combined delay as a Gaussian random variable with mean μ_T and variance σ_T, we compute the false alarm probability P_{FA} as follows:

$$P_{FA} = 2\int_{k\sigma_T}^{\infty} \frac{1}{\sigma_T\sqrt{2\pi}} e^{-\left(\frac{x-\mu_T}{\sigma\sqrt{2}}\right)^2} dx \qquad (14)$$

where $\pm k\sigma$ is the region around the mean within which synchronization measurement responses are accepted.

III. SIMULATION RESULTS

Simulations were performed to determine synchronization message round trip delay and false alarm probability density distributions, to assess the performance of the proposed QoS based scheme; and to demonstrate its advantage over the Network Time Protocol (NTP) approach. We modeled the distributions as Gaussian and validated this assumption via simulations. The simulation results presented in this section were achieved using the Optimum Network (OpNet) Performance simulation tool. The simulated network nodes are connected by 100 Mbps links, and unless otherwise stated the local delay bound for background traffic is 10 milliseconds, and synchronization message experience a fixed delay of 10 milliseconds.

We ran simulations to analyze the combined transmission and false alarm probability delay for maximum network nodes *n* equal to 2 and 26, respectively. The false alarm probability distributions for these two values of *n* together with the zero mean unit variance probability densities are presented in Fig. 3. The three graphs are almost identical, which demonstrates that the Gaussian model is good approximation for the delay distributions.

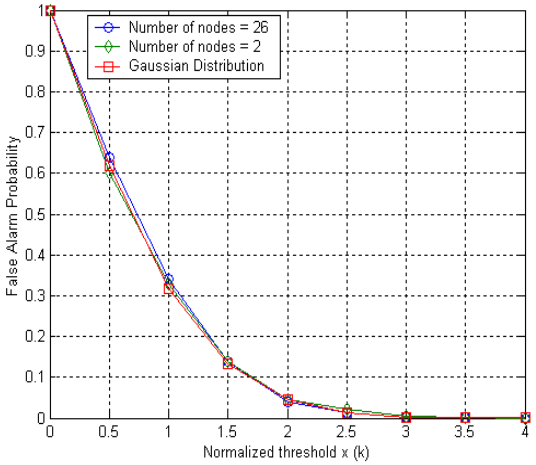

Fig. 3: False Alarm Probability

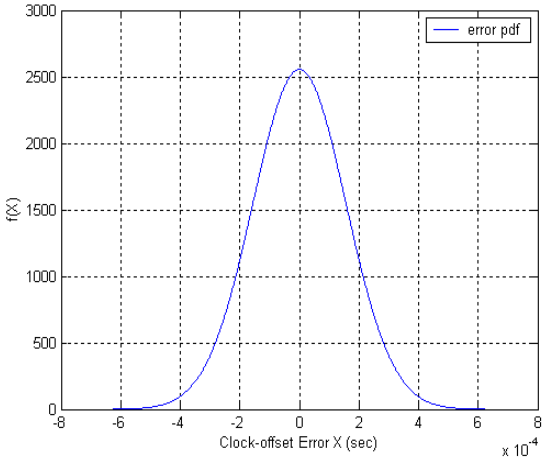

Fig. 4: Probability Density Function for QoS based Scheme

The clock-offset error probability density function for 26 tandem nodes is presented in Fig. 4. The result shows that the probability of the absolute value of the clock-offset error exceeding four times the standard deviation is practically zero. As a result, we select four times the standard deviation of the clock-offset error as the maximum error.

The simulation results for the maximum average clock-offset error for the QoS and non-QoS based time synchronization schemes are presented in Figure 7. For the NTP based scheme (non-QoS), synchronization messages are processed with background traffic, without being given any priority. As expected, the maximum average error performance of the proposed QoS based scheme is significantly better than the Non QoS based scheme, especially at low number of measurements. The performance of both schemes improves as the number of measurements increases. These results demonstrate that by controlling the random delay (i.e. stochastic noise) at each node, accurate time synchronization can be achieved.

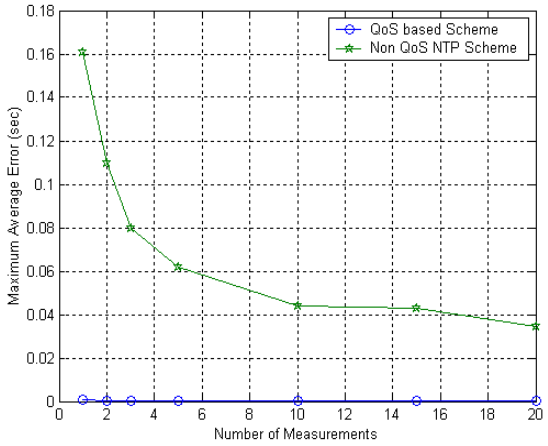

Fig. 5: QoS/Non-QoS NTP Performance Comparison

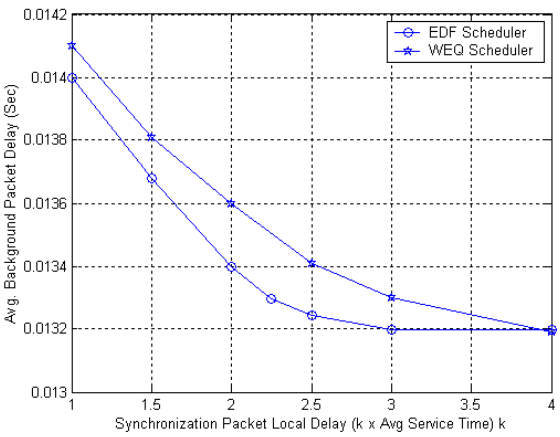

Fig. 6: Synchronization Message Delay Impact on EDF and WFQ

Synchronization packets are delayed a fixed amount within each node so that stochastic noise introduced at each node are cancelled. This delay must be selected so that it does not adversely affect the performance of other traffic traversing the node. In order to assess the impact of the proposed scheme on existing service disciplines, we incorporate the proposed QoS scheduling scheme into the Earliest Deadline First (EDF) and Weighted Fair Queuing (WFQ) scheduling disciplines. The results are presented in Fig. 6 and shows that as the local delay is increased the average delay of non-synchronization packets is reduced for both WFQ and EDF scheduling disciplines. These results demonstrate that the local delay for synchronization messages must be carefully chosen (i.e. large enough) so as not to affect other traffic traversing the node. It should be noted too that the local delay must be less than the average synchronization packet inter-arrival time so that synchronization packet buffer requirements can be met without discarding packets.

IV. Conclusion

We have proposed a secure time clock synchronization scheme, which is computationally efficient, reliable and accurate. The proposed QoS based scheme is very effective in controlling the variance of synchronization packet at each node. As a result accurate client/server time synchronization can be achieved from a single measurement

The simulation results also show that the proposed QoS can be incorporated into existing service disciplines without adversely affecting their performance. These approaches to secure packet-based time synchronization will positively influence future packet-based time synchronization schemes

References

[1] Assured Forwarding Per-Hop-Behavior Group, IETF RFC 2597

[2] F. M. Chiussi, and V. Sivaraman. "Achieving High Utilization in Guaranteed Services Network using Early-Deadline-First Scheduling," In *International Workshop on Quality of Service,* pages 209-217, Napa, California, May 1998.

[3] F. Christian, " Probabilistic Clock Synchronization," *Distributed Computing,* vol. 3, pp. 146-158, 1989.

[4] L. Georgiadis, R. Guerin, and R. Parekh, "Efficient Network QoS Provisioning Based on Per Node Traffic Shaping," *IEEE/ACM Transaction on Networking,* 4(4):482-501, August 1996.

[5] R. Gusella, and S. Zatti, "The Accuracy of Clock Synchronization Achieved by TEMPO in Berkely UNIX 4.3 BSD," *IEEE Trans. Software Engineering,* vol. 15, pp. 847-53. 1989.

[6] J. Liebeherr, D. Wrege, and D. Ferrari, "Exact Admission Control for Networks with a Bounded Delay Service," I*IEEE/ACM transaction on Networking,* 4(6):885-901, December 1996.

[7] D. L Mills, "Internet Time Synchronization: The Network Time Protocol," *IEEE Trans. On Comms.,* vol. 39, no. 10, pp. 1482-93.

[8] D. L. Mills, "Network Time Protocol (Version 3) Specification, Implementation and Analysis," Internet Request for Comments, March 1992.

[9] A. K. Parikh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control: The single node case," *In IEEE/ACM Trans. on Networking,* vol. 1. No. 3, pp. 344-357, June 1993.

[10] Vijay Sivaraman, "End-to-End Delay Service in High Speed Networks using Earliest Deadline First Scheduling."

[11] Z.-L. Zhang, D. Towsley, and J. Kurose, "Statistical Analysis of Generalized Processor Sharing Discipline," *IEEE Journal on Selected Areas in Communications,* 13(6):1071-1080, August 1995.