Secure Pairwise Broadcast Time Synchronization in Wireless Sensor Networks

Chafika BENZAID
LSI, USTHB, Algérie
cbenzaid@usthb.dz

Amin SAIAH
LSI, USTHB, Algérie
Email:btsaiah@gmail.com

Nadjib BADACHE
LSI, USTHB, Algérie
badache@mail.cerist.dz

Abstract

Most wireless sensor network (WSN) applications require a synchronized local clock in sensor nodes. One promising approach to achieve time synchronization in WSN, with significantly reduced energy consumption, is the Receiver-Only *synchronization approach. However, to the best of our knowledge, there has been no previous published work addressing the security issue of this approach. So, our contribution consists in proposing a secure version for this approach. We show that our scheme is resilient to various attacks, namely sybil attacks, message manipulation attacks, replay attacks, delay attacks, wormhole attacks, and misleading attacks. These security features are achieved by the use of public key cryptography-based authentication scheme, random nonce, and threshold-based delay attack detection.*

1. Introduction

Technology advancement in wireless sensor networks is enabling a wide range of applications. However, achieving security in sensor networks has long been known as a very difficult task [14]. Among many security concerns, securing time synchronization protocols is recently recognized as an important problem [14].

Most wireless sensor network (WSN) applications, such as data aggregation, target tracking, and TDMA scheduling, require a synchronized local clock in sensor nodes. Several time synchronization protocols have been proposed for WSNs. However, none of them were designed with security in mind, thus leaving them vulnerable to security attacks.

The proposed protocols rely on one of the following pairwise synchronization approaches: Sender-Receiver (SRS), Receiver-Receiver (RRS) or Receiver-Only (ROS). In SRS (e.g. TPSN [5], FTSP [9], GTSP [17]), a receiver adjusts its clock according to the timestamp received from a reference node. In RRS (e.g. RBS [3]), receivers within one hop use a number of synchronization pulses initiated by a sender to synchronize among themselves. While in ROS (e.g. PBS [13]), a group of nodes can be synchronized by only overhearing the timing message exchanges of a pair of sensor nodes. ROS shows a promising approach to achieve time synchronization with a significantly reduced number of timing messages, that is, with reduced energy consumption.

Recently, several schemes [4, 20, 21, 18, 2, 8, 11] have been proposed to secure time synchronization protocols in WSNs, but none of them have addressed the security issue of ROS approach. The existing solutions defend attacks by using authentication schemes, adding a random nonce, and/or detecting outliers. The proposed authentication schemes rely on symmetric cryptography using one of two approaches: shared pairwise secret keys or a single shared group-wise key (e.g. μTESLA). While the former approach incurs a heavy storage and communication overhead and lacks scalability, the latter is vulnerable to a single node compromise, and in some cases requires itself prior time synchronization (e.g. μTESLA). All these shortcomings make such schemes unsuitable for broadcast authentication. Public key cryptography (PKC), on the other hand, is desirable for broadcast authentication. Although there is a prejudice against its feasibility in WSN, it has recently been reported that PKC is *possible* in WSNs [23]. Letting us to ask the following question: Is not it time to reconsider the proposed solutions by integrating PKC-based authentication schemes?

This paper will focus on proposing a secure version of PBS using a PKC-based authentication scheme. The rest of this paper is organized as follows. Section 2 presents some background and the related work. We briefly describe the basic principle of PBS in Section 3. Section 4 identifies the various attacks an adversary can launch against the PBS protocol. Next, in Section 5, we present our approach to secure PBS in order to defeat those attacks. Finally, Section 6 outlines our concluding remarks, current and future work directions.

2. Background and Related Work

2.1. Time Synchronization in WSNs

Over the years, many protocols have been designed for maintaining synchronization of physical clocks over computer networks (e.g. [10, 7, 1]). However, these protocols are unsuitable for WSNs, due to the peculiar characteristics, limitations, and the dynamic nature of these networks [22]. Recently, there have been many time synchronization protocols adapted to WSNs [15]. The Reference Broadcast Synchronization (RBS) protocol [3] is developed for pairwise as well as multi-domain time synchronization. RBS exploits the broadcast nature of wireless communication to eliminate all sender side non-determinism; it eliminates the send time, and access time from the clock reading error. However, for a single-hop network of n nodes, this protocol requires $O(n^2)$ message exchanges, which can be computationally expensive in the case of large neighborhoods. The Timing-Sync Protocol for Sensor Networks (TPSN) [5] employs a SRS approach, and uses MAC-layer timestamping to reduce non-determinism in message delivery at both sender and receiver sides. The protocol achieves a global timescale by first creating a spanning tree structure and then performing a pairwise synchronization between parent and children nodes. TPSN achieves two times better synchronization precision than RBS [5], and requires $O(n)$ timing message exchanges. The shortcoming of TPSN is that it does not estimate the clock skew, leading to frequent resynchronization compared to protocols (e.g. RBS) that estimate both the clock offset and skew. Furthermore, TPSN does not handle dynamic topology changes. As a variation of RBS and TPSN, the Flooding Time Synchronization Protocol (FTSP) [9] extends the MAC-layer timestamping to further eliminate the non-determinism in message transmission so that one single broadcast time-stamped message is sufficient to synchronize the sender and the receivers. Although, FTSP achieves a higher level of synchronization accuracy than either RBS or TPSN, Its message complexity still of $O(n)$. The Time Diffusion Protocol (TDP) [19] achieves a network-wide equilibrium time based on the diffusion of messages, thereby involving all the nodes in the synchronization process. More recently, the Gradient Time Synchronization Protocol (GTSP) [17] is designed to provide accurately synchronized clocks between neighbors. GTSP works in a completely decentralized fashion: Every node periodically broadcasts its time information. Synchronization messages received from direct neighbors are used to calibrate the logical clock. There are many other synchronization schemes presented in the literature such as TSS, Tiny-Sync, LTS, TSync, IBS and so on. A short description of each of these algorithms can be found in [22, 16]. The main challenge in designing time synchronization protocols for WSNs is how to achieve the best tradeoff between synchronization accuracy and energy consumption. For this purpose, Noh et al. [13] proposed the Pairwise Broadcast Synchronization (PBS) scheme, which combines the merits of SRS and ROS approaches to achieve global synchronization with a significantly reduced number of synchronization messages; that is, with reduced energy consumption. PBS relies on the idea that while two nodes perform synchronization using two-way message exchanges, other nodes lying nearby can overhear the messages and can also synchronize themselves without any additional timing message transmissions. ROS shows a promising approach to achieve time synchronization with a significantly reduced number of timing messages, that is, with reduced energy consumption.

2.2. Secure Time Synchronization in WSNs

Among many security concerns, securing time synchronization protocols is recently recognized as an important problem. However, the above protocols all assume benign environments and cannot survive in hostile environments. To meet the security requirements, several secure synchronization schemes [4, 20, 21, 18, 2, 8, 11, 24] have also been proposed. Ganeriwal et al. [4] analyzed the attacks from the external attacker, especially the pulse-delay attack. Then they proposed a suite of secure single-hop and multi-hop pairwise as well as group-wise time synchronization protocols. The pulse-delay attack is prevented by checking if the end-to-end delays exceed a pre-defined threshold. The protocols also make use of the Message Authentication Code (MAC) to ensure the integrity of the time synchronization message updates. Their group-wise synchronization protocol incurs large communication overhead; the total number of messages transmitted in an n-node group is $n + 1$ with the last message containing $n - 1$ MACs, one for each pair of nodes. In [20], nodes use redundant ways for synchronizing their clocks to a common source, so that they can tolerate false or missing synchronization information send by compromised nodes. However, this requires that a sensor has a sufficient number of good neighbors, which may not be guaranteed in practice. In addition, the protocol uses authenticated unicast communication to propagate global synchronization messages, which incurs a large communication overhead. TinySeRSync [21] and ASTS [24] use authenticated MAC-layer timestamping and the μTESLA broadcast authentication protocol to overcome attacks by malicious motes. To function, however, the μTESLA protocol requires that sender and receiver are loosely time synchronized. TinySeRSync satisfies such a precondition by secure single-hop pairwise synchronization while authors of ASTS claim that this loose synchronization is not necessary if a filter is used and less than half of the nodes are captured.

However, the reasoning for this claim is lacking clarity. Song *et al.* [18] focused on dealing with the message-delay attack via outlier- and threshold-based technique. The secure time synchronization scheme for heterogeneous sensor networks [2] consists of two steps. First, all cluster heads are securely synchronized with the base station based on SRS approach and symmetric cryptography; then each cluster member is securely synchronized with its cluster head using public-key based broadcast authentication.The shortcoming of the authentication scheme in step two is that not every cluster member will necessarily be covered by more than one cluster head. Additionally, the propagation delay is estimated by the location information where a secure location service is required. The Attack-tolerant Time Synchronization Protocol (ATSP) [8], as a cooperative intrusion/anomaly detection system, compares a new time announcement with an adaptively-managed profile and identifies time announcements that deviate significantly from the expected normal behavior. ATSP can not defeat the sybil attack. The protocol proposed in [11] uses Pairing and Identity-based cryptography to secure the time synchronization in heterogenous sensor networks. Although a clusterhead only needs one broadcast message to synchronize all its cluster members, the message must piggyback the MACs of all cluster members.

It is important to note that all the proposed solutions focused on securing SRS and RRS approaches. But, none of them have addressed the security issue of the ROS approach. In addition, we observed that the proposed authentication schemes rely on symmetric cryptography using one of two approaches: shared pairwise secret keys or a single shared group-wise key (e.g. μTESLA). While the former approach incurs a heavy storage and communication overhead and lacks scalability, the latter is vulnerable to a single node compromise, and in some cases requires itself prior time synchronization (e.g. μTESLA). All these shortcomings make such schemes unsuitable for broadcast authentication. Public key cryptography (PKC), on the other hand, is desirable for broadcast authentication; a PKC-based scheme realizes immediate message authentication and thus can overcome the delayed message authentication problem present in μTESLA-like schemes. Furthermore, a PKC-based scheme is more resilient to node compromise when compared to symmetric key-based schemes. Although there is a prejudice against its feasibility in WSN, it has recently been reported that PKC is *possible* in current sensor platforms [23]. Among the various PKC approaches, Elliptic Curve Cryptography (ECC) shows the most promise due to its fast computation, small key size, and compact signatures. To achieve the same public key security as a 1024-bit RSA implementation, only 160-bit keys are needed for ECC [6]. Letting us to ask the following question: Is not it time to reconsider the proposed secure time synchronization solutions by integrating PKC-based authentication schemes? Hence, our contribution consists in proposing a secure version of PBS using a PKC-based authentication scheme.

3. Pairwise Broadcast Synchronization Principle

In this work, we focus on achieving instantaneous synchronization between sensor nodes. Hence, we assume that the skew error as well as the drift error are negligible. We assume also that the propagation delay is the same for every pair of nodes.

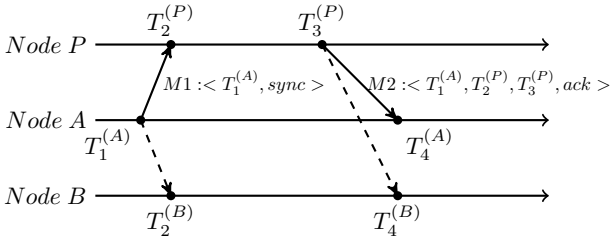

Figure 1. Clock synchronization model of PBS

The principle of PBS (See Figure 1) is that while two nodes (P and A) synchronize their clocks by SRS, other nodes (e.g. B) lying nearby can synchronize themselves by ROS. Here, $T_1^{(A)}$ and $T_4^{(A)}$ represent times measured in node A's local clock, $T_2^{(P)}$ and $T_3^{(P)}$ represent times measured in node P's local clock. Similarly, $T_2^{(B)}$ and $T_4^{(B)}$ represent times measured in node B's local clock. At time $T_1^{(A)}$, node A sends a synchronization packet $M1$ to node P with the value of $T_1^{(A)}$ attached. Nodes P and B receive this packet at $T_2^{(P)}$ and $T_2^{(B)}$, respectively. These values $T_2^{(P)}$ and $T_2^{(B)}$ can be represented by:

$$T_2^{(P)} = T_1^{(A)} + \delta_{AP} + d_{AP} \quad (1)$$

$$T_2^{(B)} = T_1^{(A)} + \delta_{AB} + d_{AB} \quad (2)$$

where δ_{AP} and d_{AP} are, respectively, the clock offset and the propagation delay between nodes A and P. δ_{AB} and d_{AB} are respectively the clock offset and the propagation delay between nodes A and B. At time $T_3^{(P)}$, node P sends back an acknowledgement packet $M2$. This packet contains the values of $T_1^{(A)}$, $T_2^{(P)}$, and $T_3^{(P)}$. Nodes A and B receive $M2$ at $T_4^{(A)}$ and $T_4^{(B)}$, respectively. These values $T_4^{(A)}$ and $T_4^{(B)}$ can be represented by:

$$T_4^{(A)} = T_3^{(P)} + \delta_{PA} + d_{PA} \quad (3)$$

$$T_4^{(B)} = T_3^{(P)} + \delta_{PB} + d_{PB} \quad (4)$$

where δ_{PA} and δ_{PB} are the clock offsets between nodes P and A, and nodes P and B, respectively. d_{PA} and d_{PB} are the propagation delays between nodes P and A, and nodes P and B, respectively.

From equation 1 and 3, we find:

$$d_{AP} = T_2^{(P)} - T_1^{(A)} - \delta_{AP} \quad (5)$$

and

$$d_{PA} = T_4^{(P)} - T_3^{(A)} - \delta_{PA} \quad (6)$$

where $\delta_{AP} = -\delta_{PA}$. Assuming that $d = d_{AP} = d_{AP}$, node A can now calculate the clock offset δ_{AP} and the propagation delay d from equation 5 and 6:

$$\delta_{AP} = \frac{(T_2^{(P)} - T_1^{(A)}) - (T_4^{(A)} - T_3^{(P)})}{2}$$

$$d = \frac{(T_2^{(P)} - T_1^{(A)}) + (T_4^{(A)} - T_3^{(P)})}{2}$$

and, subsequently, synchronizes its clock to node P's clock: $C_A = C_A + \delta_{AP}$, where C_A is the node A's local clock.

By overhearing the timing message exchanges of nodes P and A, node B can be also synchronized to the node P by applying a similar method as in RBS and with no extra timing messages. Subtracting 2 from 1 leads to:

$$T_2^{(P)} - T_2^{(B)} = \delta_{BP} + d_{AP} - d_{AB} \quad (7)$$

Assuming that $d_{AP} = d_{AB}$, equation 7 leads to calculate the clock offset between nodes B and P, δ_{BP}, as:

$$\delta_{BP} = (T_2^{(P)} - T_2^{(B)})$$

and, subsequently, node B can synchronize its clock to node P's clock: $C_B = C_B + \delta_{BP}$, where C_B is the node B's local clock.

4. Attacks Against PBS

In hostile environments, an adversary may disrupt the time synchronization process by either *external* attacks such as sybil attack, message manipulation attack, replay attack, and delay attack, or *internal* attacks, like wormhole attack, and misleading attack. We now demonstrate how these attacks can break the PBS protocol, i.e, we describe how each of the previous attack happens with PBS using the illustrative example given in Figure 1.

Sybil attack An attacker E can pretend to be P, and thus can exchange wrong timing information (i.e., $T_2^{(P)}$) with B. As a result, the time synchronization process between P and B can be disrupted.

Message manipulation attack In this attack, an attacker may drop, modify, or even forge the exchanged timing messages $M1$ or $M2$.

Replay attack An attacker E can replay an old timing message $M2$ as a legitimate message for the attacker to its neighbor nodes, and hence, the timing synchronization process can be disrupted.

Delay and wormhole attacks An attacker can intentionally delay the time messages $M1$ to introduce a synchronization error Δ (i.e., $T_2^{(B)*} = T_2^{(B)} + \Delta$ or $T_2^{(P)*} = T_2^{(P)} + \Delta$). So, node B calculates a wrong time difference δ_{err}, and it is given by:

$$\delta_{err} = (T_2^{(P)*} - T_2^{(B)}) = (T_2^{(P)} - T_2^{(B)}) + \Delta = \delta + \Delta$$

$$\delta_{err} = (T_2^{(P)} - T_2^{(B)*}) = (T_2^{(P)} - T_2^{(B)}) - \Delta = \delta - \Delta$$

Misleading attack The node P can send incorrect timestamps for the reception time message $M1$ (i.e., $T_2^{(P)*} = T_2^{(P)} + \Delta$ or $T_2^{(P)*} = T_2^{(P)} - \Delta$). Hence, $Node\ B$ calculates incorrect time difference. Similarly to delay attack, δ_{err} can be calculated as:

$$\delta_{err} = (T_2^{(P)} + \Delta - T_2^{(B)}) = (T_2^{(P)} - T_2^{(B)}) + \Delta = \delta + \Delta$$

$$\delta_{err} = (T_2^{(P)} - \Delta - T_2^{(B)*}) = (T_2^{(P)} - T_2^{(B)}) - \Delta = \delta - \Delta$$

5. Secure Pairwise Broadcast Synchronization

To address attacks on PBS, we propose a Secure Pairwise Broadcast Synchronization Protocol (SPBS) relying on:

PKC-based authentication scheme Observing that symmetric-key-based broadcast authentication schemes such as μTESLA are insufficient for WSNs, we resort to PKC for more effective solution. The PKC allows us to strengthen the security and to reduce the communication overhead.

Nonce techniques the nonce techniques verify the freshness of a message by issuing pseudo-random numbers for ensuring that old communications could not be reused in replay attacks.

Threshold-based approach If the calculated end-to-end delay exceeds a threshold value, the synchronization is aborted. As suggested in [3] and according to the experimental data in [4], the end-to-end delay follows

a Gaussian distribution. Thus, with a 99.97% confidence, the true delay will fall in the interval $[d_{avg} - 3\sigma, d_{avg} + 3\sigma]$, where d_{avg} is the average delay and σ is the variance. Hence, the delay upper-bound can be set to $d_{max} = d_{avg} + 3\sigma$ while the delay lower-bound can be set to $d_{min} = d_{avg} - 3\sigma$.

5.1. Notations

We adopt the following notations throughout the description of the protocol.

- ID_A, ID_P denote the sensor identifiers of nodes A and P, respectively.
- N_A is a nonce generated by A (a nonce is an unpredictable bit string, usually used to achieve freshness).
- SK and PK denote respectively the secret key and the public key.
- $H < M >$ denotes the computation of a hash function on message M.
- $SIG_{SK}(M)$ is the signature of M signed by SK.
- $A(T) \rightarrow (T')*$ denotes A broadcasts, at A's local clock T, a message to its neighbors. The message is received at the neighbors' local clock T'.

5.2. Protocol Detail

1. Node A first broadcasts a synchronization message to node P, containing its identifier (ID_A), a challenge nonce (N_A), the value of the timestamp $T_1^{(A)}$, and a signature SIG_{SK_A} over the hash of ID_A, N_A, and $T_1^{(A)}$.

$$M1 = (ID_A, N_A, T_1^{(A)}, sync)$$
$$A(T_1^{(A)}) \rightarrow (T_2^{(P)}, T_2^{(B)})* : (M1, SIG_{SK_A}(H < M1 >))$$

2. On receiving the synchronization message, node P records its arrival time $T_2^{(P)}$ and attempts to authenticate it using the A's public key PK_A. If the authentication fails, the message is dropped. Otherwise, node P replies by broadcasting an acknowledgement message to node A at $T_3^{(P)}$. The message contains the P's identifier (ID_P), the nonce N_A, the timestamps $T_1^{(A)}$, $T_2^{(P)}$, $T_3^{(P)}$, and a signature over the hash of the aforementioned fields.

$$M2 = (ID_P, N_A, T_1^{(A)}, T_2^{(P)}, T_3^{(P)}, ack)$$
$$P(T_3^{(P)}) \rightarrow (T_4^{(A)}, T_4^{(B)})* : (M2, SIG_{SK_P}(H < M2 >))$$

3. On receiving the acknowledgement message, node A records its arrival time $T_4^{(B)}$ and then checks its authenticity using the P's public key PK_P. If the check is positive, node A verifies if the calculated end-to-end delay falls in the interval $[d^*_{min}, d^*_{max}]$. If so, node A proceeds to the calculation of its clock offset relative to P.

$$d_1 = \frac{(T_2^{(P)} - T_1^{(A)}) + (T_4^{(A)} - T_3^{(P)})}{2};$$
if $(d^*_{min} \leq d_1 \leq d^*_{max})$
then $\delta_{AP} = \frac{(T_2^{(P)} - T_1^{(A)}) - (T_4^{(A)} - T_3^{(P)})}{2}$;
else *abort*;

4. By only overhearing the timing message exchanges of nodes P and A, node B can be also securely synchronized to node P. On receiving the synchronization and acknowledgement messages, node B records their respective arrival time $T_2^{(B)}$ and $T_4^{(B)}$ and then checks their authenticity using the A's public key PK_A and the P's public key PK_P, respectively. If the check is positive, node B verifies if the calculated end-to-end delay falls in the interval $[d^*_{min}, d^*_{max}]$. If so, node B proceeds to the calculation of its clock offset relative to P.

$$d_2 = (T_4^{(B)} - T_3^{(P)})$$
if $(d^*_{min} \leq d_2 \leq d^*_{max})$
then $\delta_{BP} = (T_2^{(P)} - T_2^{(B)})$;
else *abort*;

5.3. Security Analysis

In this section, we discuss how SPBS can cope with attacks cited in Section 4:

Message *integrity* and *authenticity* are achieved through the use of a public key-based signature scheme such as the elliptic curve digital signature algorithm (ECDSA), where the sender node signs the hash of a timing message with its secret key (SK) and broadcasts the signature along with the message. The receivers verify the correctness of the signature by using the sender's public key (PK). This prevents attackers from successfully modifying any values in the timing messages. Furthermore, the attacker cannot impersonate nodes A and P as she does not know their secret keys SK_A and SK_P, respectively. Thus, the protocol is protected against the *Sybil* attack.

Replay attacks are prevented by using a random nonce, N_A, during the synchronization handshake. In addition, the use of the same random nonce in both synchronization and response messages means that node P can't send the acknowledgement message until it has received the corresponding synchronization message sent by A, thus preventing the node P from jumping the gun.

An interesting observation, made by [4], is that by performing a *delay* attack, a *wormhole* attack, or a *misleading* attack, the attacker also changes the computed end-to-end delay. We exploited this observation in SPBS to defeat these attacks. SPBS prevents these attacks by comparing the computed message end-to-end delays $d1$ and $d2$ with a threshold of propagation delay. Our protocol uses two thresholds d^*_{max} and d^*_{min} representing, respectively, the minimum and maximum propagation delay between neighbor nodes. d^*_{max} and d^*_{min} can be estimated before network deployed with high accuracy [4]. If the computed delay is greater than the maximal expected delay or smaller than the minimal expected delay, the offset calculation is aborted. Note that we have added no extra overhead on the functionality of the ROS approach.

6. Concluding Remarks

This paper presented a Secure Pairwise Broadcast Synchronization Protocol (SPBS) which ensure the security of the ROS approach. SPBS reduces communication overhead due to use of PKC, and is highly robust against different attacks. Mathematical security analysis and performance evaluation via computer simulations form the basis of our current work in this progressive research work. A future work consists in extending SPBS to the case of secure group-wise time synchronization based on scheme proposed in [12].

References

[1] F. Cristian. Probabilistic clock synchronization. *Distributed Computing*, 3:146–158, 1989.

[2] X. Du, M. Guizani, Y. Xiao, and H. Chen. Secure and efficient time synchronization in heterogeneous sensor network. *IEEE Trans. on Vehicular Technology*, 57(4):2387–2394, Jul 2008.

[3] J. Elson and D. EstrinBirman. Fine-grained network time synchronization using reference broadcast. *In Proc. of the 5th Symp. on Oper. Syst. Design and Implementation (OSDI)*, pages 147–163, Dec. 2002.

[4] S. Ganeriwal, S. Capkun, and M. B. Srivastava. Secure time synchronization in sensor networks. *ACM Trans. on Information and Syst. Security*, 11(4):1–35, 2008.

[5] S. Ganeriwawal, R. Kumar, and S. M. Timing-sync protocol for sensor networks. *In Proc. of the 1st ACM Conf. on Embedded Networked Sensor Systems (SenSys)*, pages 138–149, Nov. 2003.

[6] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and rsa on 8-bit cpus. *In Proc. of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004)*, pages 119–132, Aug. 2004.

[7] R. Gusell and S. Zatti. The accuracy of clock synchronization achieved by tempo in berkeley unix 4.3 bsd. *IEEE Trans. on Software Engineering*, 15:847–853, 1989.

[8] X. Hu, T. P. Kang, and G. Shin. Attack-tolerant time-synchronization in wireless sensor networks. *In Proc. of IEEE INFOCOM 2008*, pages 41–45, mai 2008.

[9] M. Maroti, B. Kusy, G. Simon, and A. Ledezi. The flooding synchronization protocol. *In Proc. of the 2nd ACM Conf. on Embedded Networked Sensor Systems (SenSys)*, Nov. 2004.

[10] D. L. Mills. Internet time synchronization: the network time protocol. *IEEE Trans. on Commun.*, 39(10):1482–1493, Oct. 1991.

[11] S. M. Mizanur Rahman and K. El-Khatib. Secure time synchronization for wireless sensor networks based on bilinear pairing functions. *IEEE Trans. on Parallel and Distributed Systems*, 99, April 2010.

[12] K. L. Noh, E. Serpedin, and K. Qaraqe. Extension of pairwise broadcast clock synchronization for multicluster sensor networks. *EURASIP Journal on Advances in Signal Processing*, 2008(Article 71):10, 2008.

[13] K. L. Noh, E. Serpedin, and K. Qaraqe. A new approach for time synchronization in wireless sensor networks: Pairwise broadcast synchronization. *IEEE Trans. Wireless Commun.*, 9:3318–3322, 2008.

[14] A. Perrig, J. Stankovic, and D. Wagner. Security in wireless sensor networks. *Commun. ACM*, 47(6):53–57, 2004.

[15] S. Rahamatkar, A. Agarwal, and N. Kumar. Analysis and comparative study of clock synchronization schemes in wireless sensor networks. *Int. J. Comp. Sc. and Engg*, 2(3):523–528, April 2010.

[16] K. Rmer, P. Blum, and L. Meier. Time synchronization and calibration in wireless sensor networks. *Handbook of Sensor Networks: Algorithms and Architectures*, pages 199–237, 2005.

[17] P. Sommer and R. Wattenhofer. Gradient clock synchronization in wireless sensor networks. *In Proc. of the 2009 Int. Conf. on Information Processing in Sensor Networks (IPSN'09)*, pages 37–48, 2009.

[18] H. Song, S. Zhu, and G. Cao. Attack-resilient time synchronization for wireless sensor networks. *Ad Hoc Networks Journal*, 5(1):112–125, 2007.

[19] W. Su and I. F. Akyildiz. Time-diffusion synchronization protocol for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 13(2):384–397, April 2005.

[20] K. Sun, P. Ning, and C. Wang. Secure and resilient clock synchronization in wireless sensor networks. *IEEE Journal on Selected Areas in Commun.*, 24(2), Feb. 2006.

[21] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. Tinysersync: Secure and resilient time synchronization in wireless sensor networks. *In Proc. of the 13th ACM Conf. on Computer and Commun. Security (CCS'06)*, pages 36–42, Nov. 2006.

[22] B. Sundararaman, U. Buy, and A. D. Kshemkalyani. Clock synchronization for wireless sensor networks: A survey. *Ad Hoc Networks*, 3:281–323, 2005.

[23] R. Wang, W. Du, X. Liu, and P. Ning. Shortpk: A short-term public key scheme for broadcast autentication in sensor networks. *ACM Trans. on Sensor Networks*, 6(1):9:1–9:29, 2009.

[24] Y. Xianglan, Q. Wangdong, and F. Fei. Asts: An agile secure time synchronization protocol for wireless sensor networks. *In Proc. of the 3rd Int. Conf. on Wireless Commun., Networking and Mobile Computing (WiCom'07)*, pages 2808–2811, 2007.