INL/CON-09-15940
PREPRINT

Time Synchronization in Hierarchical TESLA Wireless Sensor Networks

International Symposium on Resilient Control Systems

Jason L. Wright
Milos Manic

August 2009

The INL is a
U.S. Department of Energy
National Laboratory
operated by
Battelle Energy Alliance

Time Synchronization in Hierarchical TESLA Wireless Sensor Networks

Jason L. Wright
Idaho National Laboratory
2525 Freemont Avenue
Idaho Falls, ID 83415, USA
jlwright@ieee.org
jason.wright@inl.gov

Milos Manic
Department of Computer Science
University of Idaho at Idaho Falls
1776 Science Center Dr., Ste. 306
Idaho Falls, ID 83402, USA
misko@ieee.org

***Abstract*—Time synchronization and event time correlation are important in wireless sensor networks. In particular, time is used to create a sequence events or time line to answer questions of cause and effect. Time is also used as a basis for determining the freshness of received packets and the validity of cryptographic certificates. This paper presents secure method of time synchronization and event time correlation for TESLA-based hierarchical wireless sensor networks. The method demonstrates that events in a TESLA network can be accurately timestamped by adding only a few pieces of data to the existing protocol.**

I. Introduction

In a sensor network, time plays a critical role. Correct time allows the creation of a sequence of events from disparate sensors so that a time line can be created. An accurate time line can answer questions not only of order of events, but also of causality. Time is also used for critical radio functions: managing duty cycle and creation of TDMA schedules for efficient bandwidth usage. Time is also used to determine the freshness or liveliness of received packets and as a important factor in establishing the validity of cryptographic certificates.

In this paper, the Timed, Efficient, Streaming, Loss-tolerant Authentication protocol (TESLA)[1] and the hierarchical sensor network based on TESLA described in [2] will be considered. A low-overhead and secure method of time synchronization and time-series event correlation is presented. The method described in this paper integrates tightly with the security and routing protocols already developed for TESLA-based wireless sensor networks and extends them to provide accurate timing information.

The rest of the paper is organized as follows. Section II provides background information on the hierarchical network, TESLA protocol, and a summary of time synchronization protocols for wireless sensor networks. Section III describes a method that builds upon the previous work and provides a method for synchronizing the time on a TESLA network. Section IV describes a method for providing an event time correlation amongst a group of sensors, and Section V provides the conclusion.

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 with the U. S. Department of Energy. The United States Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

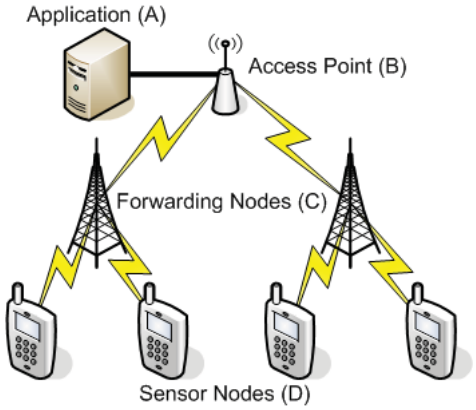

Fig. 1. Hierarchical TESLA network

II. Background

This paper builds upon the TESLA protocol and TESLA hierarchical TESLA certificates work. It adds a time synchronization and event correlation protocol to the existing messaging and security infrastructure without adding a large amount of overhead. This section will provide an overview of the network topology (II-A), TESLA protocol (II-B), TESLA certificates (II-C), and time synchronization protocols (II-D).

A. Hierarchical Network

In [2], a hierarchical network consisting of the application (A), access points (B), forwarding nodes (C), and sensors (D) is shown in Figure 1.

The application (A) is the root of the network. It must correlate events sent by sensors and provide a time line of those events. Access points serve as the root of the wireless network. Forwarding nodes are simplistic devices that listen on an access point network and communicate with sensor nodes.

The application must communicate with sensors to provide configuration information, and sensors must provide time stamped events back to the application. There is no need for direct communication between paths not shown in Figure 1. In

particular, sensors need not communicate directly with other sensors.

Each level in the hierarchical network has different computational resources available. Sensors are assumed to be the most resource constrained. Typically they will be battery powered which imposes limitations on processing power, transmit power, and local storage. Forwarding nodes will have more power resources, and access points still more. Finally, the application is assumed to have substantially more resources than the other network nodes.

One consequence of the limited computational resources at the sensor level means that authentication methods relying on traditional public key cryptography (RSA, Diffie-Hellman, etc.) would be impractical. The modular exponentiation operations are computationally intensive resulting in degraded battery life and slow performance. Several works have analyzed the energy cost of cryptography in wireless networks, including [3], [4]. The hierarchical TESLA network described assumes that public key cryptography is used at the access point and application levels.

B. TESLA Protocol

The TESLA protocol as described in [1] and revised in [2] divides time into intervals of equal duration. A time slot n is assigned a key tK_n, and packets generated during this time slot are signed with a MAC using this key. At some point in the future, the node will disclose the seed used to generate this key, s_n, and all nodes on the network can then verify the MAC sent during time slot n. This key disclosure delay is d time slots for $d > 0$.

The protocol uses two publicly known hash functions, F' and F. F' is used to compute keys given a seed value, via Equation 1.

$$F'(s_n) = tK_n \tag{1}$$

The other hash function, F is used to generate the seeds. A node chooses a terminal seed, s_l and computes the sequence $(s_0, s_1, \ldots, s_l)$ via Equation 2.

$$F(s_n) = s_{n-1} \tag{2}$$

$$s_l \xrightarrow{F} s_{l-1} \xrightarrow{F} \cdots \xrightarrow{F} s_1 \xrightarrow{F} s_0 \tag{3}$$

The seeds are used and disclosed in the order of the sequence starting with s_0. The length of the sequence l is chosen to be sufficiently long to increase the complexity of the protocol[1].

A node buffers packets during time slot n, and d time slots later, s_n is revealed. The s_n is verified using Equation 2 and then the MAC of each packet sent during time slot n is verified using the key $F'(s_n) = tK_n$ (Equation 1)[1].

C. TESLA Certificates

A trusted third party (TTP) is described in [2]. This trusted third party is used to generate certificates and keys for each device in the network.

For a sensor D, the initial certificate consists of the identity of the sensor (ID_D), an initial key encrypted with the public key of the trusted third party ($\{iK_D\}_{K_A^+}$), and a timestamp, all of which are signed by the TTP.

$$iCert_{TTP}(D) = (ID_D, \{iK_D\}_{K_A^+}, TS_{TTP}, SIGN_{K_{TTP}^-}(\cdots)) \tag{4}$$

D. Time Synchronization Protocols

Several different approaches can be found in literature for synchronizing the time in wireless sensor networks[5]. The Reference Broadcast Synchronization uses one-way synchronization messages to clock a network[6]. The Timing-sync Protocol for Sensor Networks (TPSN), uses a two-way exchange. Further, TPSN also includes an explicit hierarchy computation[7]. The hierarchy of nodes is implicit in the architecture of this TESLA based network. Flooding Time Synchronization Protocol (FTSP) uses time-stamped broadcast messages, which is similar to the protocol presented in this paper, but it is not integrated with the security and routing protocols already in use in hierarchical TESLA networks[8].

This work, however, builds upon [9], where, all devices are assumed to have a clock with a maximum drift ρ. Rather than synchronizing the time of all devices, timestamps relative to each device are computed and then converted by the receiving device to its own clock. This is similar to the method described here except that in this work the only device that needs to do the conversion is the application. By exploiting the architecture of the TESLA network, event time correlation can be performed with little overhead.

III. Time Synchronization Method

This section describes a method by which the time on a sensor is synchronized with the application (A). It consists of two parts, an initial synchronization and later an authentication. Normal synchronization protocols require authentication first, but the protocol described below synchronizes to an unidentified application first and later confirms its identity.

A. Boot Time Synchronization

For the purposes of this discussion, access points, B, and forwarding nodes, C, will not be considered. For now, it is assumed there is a communication path between sensors, D, and the application, A.

A sensor node wishing to join the network need not know the length of a time slot t or the key disclosure delay, d. The application, A, broadcasts a packet containing the time, the disclosure delay, and the seed from d time slots ago all signed with the key for the current time slot.

$$A \rightarrow all : (time, d, s_{n-d}, MAC_{tK_n}(\cdots)) \tag{5}$$

In the next time slot $(n + 1)$, a similar packet is sent. This packet will be signed by a key tK_{n+1} derived from the next seed from the sequence: s_{n+1}.

$$A \rightarrow all : (time, d, s_{n+1}, MAC_{tK_{n+1}}(\cdots)) \tag{6}$$

By applying Equation 2, $F(s_{n+1}) = s_n$, it can be verified that this packet came from the same A as the previous. No assumption can be made as to the identity of A however.

During time slot $n + d - 1$, the following packet will be sent by A.

$$A \rightarrow all : (time, d, s_{n-1}, MAC_{tK_{n+d-1}}(\cdots)) \quad (7)$$

Finally, during time slot $n+d$, the following packet will be sent by A.

$$A \rightarrow all : (time, d, s_n, MAC_{tK_{n+d}}(\cdots)) \quad (8)$$

This packet is first verified using the relation $F(s_n) = s_{n-1}$, which implies the packet is from the same A as the previous. The node then uses Equation 1 to compute tK_n from s_n and uses the resulting key to verify the MAC the packet in (5).

At this point, there is a cryptographically strong guarantee that D is receiving packets from the same A, but no assurance the A should be trusted. D will continue monitoring this A and computing clock skew and such until its identity can be verified. D can also measure the length of a time slot by examining the time values sent by A. Thus, to loosely synchronize its time with a node, D needs no prior knowledge of the time, the length of a time slot, or the key disclosure delay. The method for deciding whether the identity of A is valid will be discussed in the next section.

B. Authenticating Time

After a sensor node D has loosely synchronized its time with A, it must then verify the identity of the A with whom it is communicating. To do so, a fairly simple protocol can be used. A nonce is encrypted with iK_D and sent to A along with $iCert_{TTP}(D)$. Only a node with knowledge of K_A^+ will be able to decrypt iK_D from the certificate.

$$D \rightarrow A : (req, \{nonce\}_{iK_D}, iCert_{TTP}(D)) \quad (9)$$

A must then respond with $nonce + 1$, also encrypted with iK_D, and with a MAC computed with tK_n. $nonce+1$ is used to prevent a replay attack.

$$A \rightarrow D : (ok, \{nonce + 1\}_{iK_D}, MAC_{tK_n}(\cdots)) \quad (10)$$

When s_n is disclosed, D will verify the seed is from the correct sequence and also verify the MAC of the packet above. It will then decrypt the received nonce and compare it against the nonce that was sent originally. With this exchange, D will have verified that the station claiming to be A is, in fact, the A desired.

IV. Event Time Correlation

The ability to create a time line of events from disparate sensors is an important feature of a wireless sensor network. This section provides a protocol for performing this calculation with maximum precision by adding timestamps to event reporting messages.

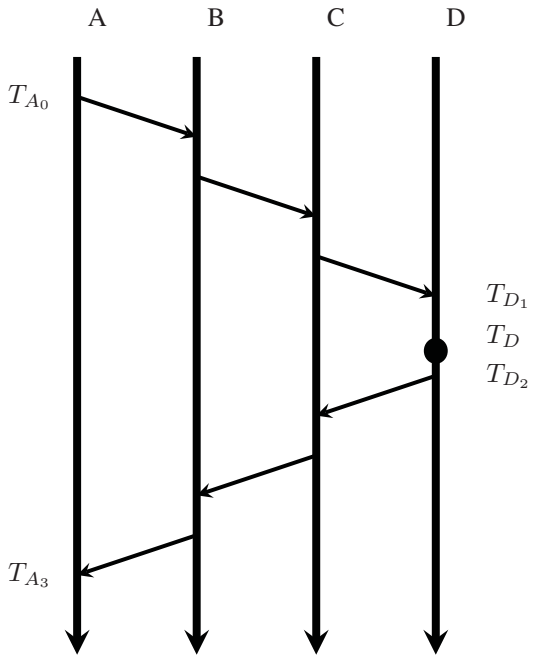

Fig. 2. Synchronizing with access points and forwarding nodes

This section is divided as follows. Section IV-A describes the relative time of each sensor and how it can be converted to an absolute time. Section IV-B describes the time protocol itself, and Section IV-C analyzes the accuracy of the protocol and algorithm.

A. Relative Time Calculation

In this section, we will consider an event E that occurs at time T. A sensor D will detect the event relative to its current clock value T_D. It is assumed that application has a high precision synchronization with the real world (e.g. a GPS clock). The goal is to coordinate the time at D with the time at the application (A) with maximum precision. This is accomplished by adding two timestamps to event reporting messages.

The communications path between the sensor D and the application A will have at least one forwarding node C and an access point B. At each point along the path two types of delay will be added: propagation delay and processing delay. Figure 2 shows the path of a packet going from application to sensor via an access point and a forwarding node.

The path in Figure 2 going from A to D is a message disclosing s_n as described. The format of this message is described in Equation 8. T_{D_1} is the time, relative to the clock of D at which the message arrived. T_D is the time an event happened relative to the clock at D, and T_{D_2} is the time the message containing T_D is transmitted by D back towards A. T_{A_0} and T_{A_3} are the times, relative to the value of the clock at A when the disclosure packet was sent and the event message was received by A, respectively.

The offset and delay of T_D relative to A can be computed as described in [10] using the following equations:

$$offset = [(T_{D_1} - T_{A_0}) + (T_{D_2} - T_{A_3})]/2, \quad (11)$$
$$delay = (T_{A_3} - T_{A_0}) - (T_{D_2} - T_{D_1}). \quad (12)$$

The time at which the event happened can be computed by:

$$T_A = T_D - offset + T_D - T_{D_1}. \quad (13)$$

The sensor need only set its current clock value to the authenticated value from the last s_n disclosure packet T_{A_0}. T_D is then a positive offset ahead of T_{D_1} and the offset/delay equations can be simplified to:

$$offset = (T_{D_2} - T_{A_3})/2, \quad (14)$$
$$delay = T_{A_3} - T_{D_2}. \quad (15)$$

B. Time Protocol

As shown in Section III-A, the application will send out a disclosure packet:

$$A \rightarrow all : (time, d, s_n, MAC_{tK_{n+d}}(\cdots)). \quad (16)$$

It will be forwarded by all access points and forwarding nodes. It will finally arrive at the destination sensors. The sensors will update their current time to $time$ and take note of s_n.

When an event occurs, the sensor will use a key $K_{A,D}$ to sign a message and send it back to A:

$$D \rightarrow A : (T_{D_2}, s_n, T_{D_2} - T_D, MAC_{K_{A,D}}(\cdots)) \quad (17)$$

Forwarding nodes and access points simply forward this message back to the application which notes its time of arrival and verifies the MAC on the packet.

Using Equation 14, the actual time the event happened (T_E) can be computed as shown in Equation 18.

$$T_E = T_A - (T_{D_2} - T_{A_3})/2 - (T_{D_2} - T_D) \quad (18)$$

C. Analysis

Referring to Figure 2, a particular disclosure message sent from the application (A) to a sensor (D) must be transmitted three times. At each point there is some delay added (transmit/receive delay, propagation delay, etc). Also, each vertical line represents a separate clock, free-running clock domain. The application, access points, forwarding nodes, and sensors each have their own local clock and they are always drifting relative to each other.

For example, a 4MHz oscillator as might be found on a wireless sensor has a total frequency tolerance of $\pm 25ppm$ This results in a maximum drift of $\rho = 0.025ms/sec$.

If the key disclosure delay for the network is chosen to be five seconds, then the maximum drift contributes $0.125ms$ of inaccuracy. This drift bounds the maximum accuracy of event correlation because the delay is accounted for by the time stamps at the application (T_{A_0}, T_{A_3}) and the sensor (T_{D_1}, T_D, T_{D_2}) using Equation 18.

V. Conclusion

The approach presented in this paper is based on TESLA protocol and TESLA certificates. It adds a method for determining the absolute time of events based upon the already present key disclosure protocol.

It has been demonstrated that events in a TESLA network can be accurately timestamped by adding only a few pieces of data. Specifically, additional timing information is added to key disclosure messages and additional timestamps are added to event reports. Instead of relying on pairwise synchronization, the key disclosure messages already required by the TESLA protocol are reused to provide a relative synchronization. An application can use this to provide accurate time lines of events from disparate sensors. The drift in the sensors becomes the predominate factor in the accuracy of the computed timestamps, but this is bounded by the frequent key disclosures required by the TESLA protocol.

References

[1] A. Perrig, R. Canetti, D. Tygar, and D. Song, "The TESLA broadcast authentication protocol," *Cryptobytes*, vol. 5, no. 2, Summer/Fall 2002, RSA Laboratories.

[2] M. Bohge and W. Trappe, "An authentication framework for hierarchical ad hoc sensor networks," in *WiSe '03: Proceedings of the 2nd ACM workshop on Wireless security*. New York, NY, USA: ACM, 2003, pp. 79–87.

[3] A. Hodjat and I. Verbauwhede, "The energy cost of secrets in ad-hoc networks," in *Proceedings of IEEE CAS Workshop on Wireless Communication and Networking*, September 2002.

[4] J. Großschädl, A. Szekely, and S. Tillich, "The energy cost of cryptographic key establishment in wireless sensor networks," in *ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security*. New York, NY, USA: ACM, 2007, pp. 380–382.

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

[6] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, 2002.

[7] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2003, pp. 138–149.

[8] M. Maróti, B. Kusy, G. Simon, and Ákos Lédeczi, "The flooding time synchronization protocol," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2004, pp. 39–49.

[9] K. Römer, "Time synchronization in ad hoc networks," in *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*. New York, NY, USA: ACM, 2001, pp. 173–182.

[10] S. Chen, A. Dunkels, F. Österlind, T. Voigt, and M. Johansson, "Time synchronization for predictable and secure data collection in wireless sensor networks," in *Proceedings of The Sixth Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2007)*, Corfu, Greece, 2007.