Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1954121, 14 pages
https://doi.org/10.1155/2018/1954121

Research Article

Security Vulnerabilities and Countermeasures for Time Synchronization in TSCH Networks

Wei Yang[1], **Yadong Wan**[2], **Jie He**[2] and **Yuanlong Cao**[1]

[1]*School of Software, Jiangxi Normal University, Nanchang, 330022, China*
[2]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, 100083, China*

Correspondence should be addressed to Wei Yang; yw@jxnu.edu.cn

Received 20 August 2018; Revised 15 November 2018; Accepted 27 November 2018; Published 10 December 2018

Guest Editor: Jiageng Chen

Copyright © 2018 Wei Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Time-slotted channel hopping (TSCH), which can enable highly reliable and low-power wireless mesh networks, is the cornerstone of current industrial wireless standards. In a TSCH network, all nodes must maintain high-precision synchronization. If an adversary launches a time-synchronization attack on a TSCH network, the entire network communication system can be paralyzed. Thus, time-synchronization security is a key problem in this network. In this article, time synchronization is divided into single-hop pairwise, clusterwise, and three-level multihop according to the network scope. We deeply analyze their security vulnerabilities due to the TSCH technology itself and its high-precision synchronization requirements and identify the specific attacks; then, we propose corresponding security countermeasures. Finally, we built a test bed using 16 OpenMoteSTM nodes and the OpenWSN software to evaluate the performance of the proposed scheme. The experimental results showed that serious security vulnerabilities exist in time-synchronization protocols, and the proposed countermeasures can successfully defend against the attacks.

1. Introduction

Most industrial applications, e.g., steel mills, chemical industries, and oil refineries, need real-time monitoring and management processes [1, 2]. Traditionally, wired industrial automation and monitoring systems have been deployed to monitor temperature, pressure, or tank-fill levels. But it is difficult and expensive to install communication cables in a factory [3]. With the recent advances in wireless technology, industrial wireless sensor networks (IWSNs) have become a trend, instead of the traditional wired industrial systems. Their advantages are easier deployment and cheaper maintenance. In particular, they can be used in mobile objects and explosive environments.

As industrial wireless applications have critical requirements for reliability, low power, and real-time response rates, IWSNs face many challenges. Research has shown that wireless communication is vulnerable to external interference, path obstruction, and multipath fading. Moreover, nodes waste considerable energy in idle listening states, e.g., IEEE802.15.4-2006 networks [4].

The time-slotted channel-hopping (TSCH) technique can be applied to low-power and highly reliable wireless mesh networks. All nodes in a TSCH network must maintain high-precision synchronization. The nodes radios switch on according to the network schedule, which can avoid idle listening. It also uses a channel-hopping technique to improve the wireless communication reliability. Currently, the TSCH technology is the fundamental for the industrial wireless standards, e.g., ISA100.11a [5], WirelessHART [6], and IEEE802.15.4-2015 [7].

Figure 1 shows a sample timeslot-channel schedule in a TSCH network with a 101-slot super-frame. The horizontal axis represents ASN (absolute slot number) value which indicates how many timeslots have elapsed after the network formation. And the vertical axis represents communication channel. The communication of nodes happens in one timeslot and communication channel according to the network schedule. Some nodes can sleep in a specified timeslot to save energy (e.g., node B and node D sleep in timeslot 1). In addition, the network adopts a channel-hopping technology based on time to improve wireless communication reliability.

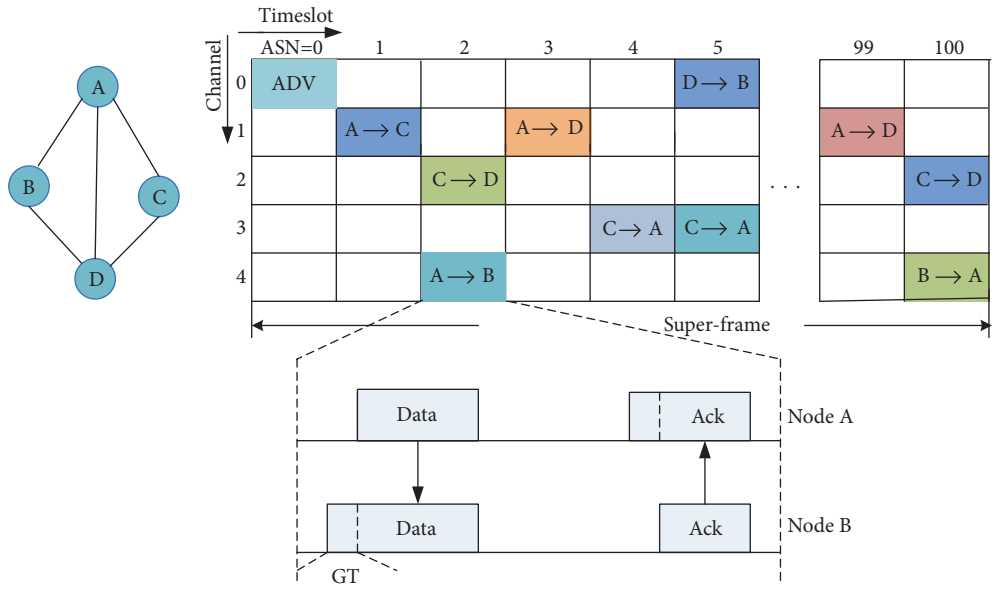


Figure 1: Sample timeslot-channel schedule in a TSCH network.

Overall, high-precision time synchronization is the core feature of TSCH networks.

Time synchronization is fundamental for the TSCH-based wireless networks. However, the time-synchronization protocol in TSCH networks focuses on energy efficiency and clock accuracy while ignoring security issues. If an adversary launches a time-synchronization attack on a TSCH network, the entire network communication system can be paralyzed. There are some secure time-synchronization protocols in WSNs (e.g., SPS [8] and SMTS [9]). But those countermeasures cannot be directly applied to the time-synchronization protocol in TSCH networks. The SPS (secure pairwise synchronization) protocol [8] is designed to defend against the pulse-delay attack in TPSN time synchronization. The SMTS (secured maximum-consensus-based time-synchronization) protocol [9] is designed to defend against message-manipulation attacks in MTS time synchronization. The time-synchronization protocol in TSCH networks is different from the MTS and TPSN protocol. There exist difference security vulnerabilities in the process of time synchronization. And the security countermeasures should also consider the detail of implementation of the time-synchronization protocol. It is necessary to research the secure time synchronization in TSCH-based wireless networks.

In our paper, time synchronization in TSCH networks is divided into single-hop pairwise, clusterwise, and multihop according to the network scope. We analyze in detail their security vulnerabilities, due to the high-precision synchronization requirements and TSCH protocol itself, and identify some specific attacks and then propose corresponding security countermeasures.

The contributions of the paper are threefold. Firstly, we analyze in detail the vulnerability of single-hop pairwise synchronization and propose security countermeasures that include an authentication mechanism and a clock-offset filter (COF) algorithm. The COF algorithm can filter out time-synchronization packets from malicious nodes. Secondly, we analyze in detail the vulnerability of clusterwise synchronization and propose an improved μTESLA scheme that supports immediate authentication. It does not need to wait until it receives the disclosed key before authenticating the packets. Finally, we analyze in detail the vulnerability of multihop synchronization and define an error-accumulation attack. We propose a multipath approach based on trust modeling, which can find a secure path to the root node by establishing a trust model between nodes.

A test bed using 16 OpenMoteSTM nodes and the OpenWSN software was built to validate the effectiveness and feasibility of the security countermeasures.

The paper is organized as follows. Section 2 introduces secure time-synchronization protocols in wireless sensor networks. Section 3 presents time synchronization in TSCH networks and the attack model. Sections 4, 5, and 6 describe secure pairwise, secure clusterwise, and secure multihop time synchronization, respectively. Section 7 presents the experimental evaluation of the secure time synchronization in a network with 16 OpenMoteSTM nodes. Section 8 presents a comparison with other secure time-synchronization protocols. Section 9 concludes the paper.

2. Related Work

Huang et al. [10] showed the seq_num attacks and global time attacks on the flooding time-synchronization protocol (FTSP). Then they proposed a series of countermeasures which include new root-selection and blacklist filter mechanisms to protect against the above attacks. Zhang et al. [11] observed a novel time-synchronization attack (TSA) that can manipulate the timing information in a smart grid. The attack

can reduce the fault-location performance and disable the voltage-instability alarm.

Ganeriwal et al. [8] defined a pulse-delay attack on the timing-synchronization protocol for sensor networks (TPSN). This attack causes legitimate nodes to calculate an error clock offset. Then, they proposed a SPS protocol that adopts message-authentication codes and end-to-end delay-estimation methods. The experiment, which is conducted on Mica2 motes, shows that SPS can successfully detect the attack.

He et al. [9] analyzed the vulnerability of the MTS protocol and described message-manipulation attacks. They then proposed a secure MTS protocol using hardware and logical clocks to detect the attacks. The result showed that the protocol can quickly compensate the clock offset. And they also proposed a SATS protocol [12] to protect against message-manipulation attacks in the average-consensus-based time-synchronization (ATS) protocol.

Dong et al. [13] showed Sybil attacks and compromise attacks can destroy a distributed time-synchronization protocol. They proposed RTSP that employs a graph-theoretical technology to detect attacks. But the security mechanism can not be directly used in centralized time-synchronization protocols.

Yang et al. [14] analyzed the vulnerability of the timing-synchronization protocol in IEEE802.15.4e networks. They pointed out the ASN and timeslot-template synchronization attack. Then, they proposed security countermeasures which fully consider the characteristic of IEEE802.15.4e networks to defend against the attacks. However, the paper only focused on secure single-hop synchronization, and only simulations were conducted to verify the effectiveness of the countermeasures.

3. Time Synchronization in TSCH Networks

In this section, we briefly describe time synchronization in TSCH networks. Then, we present a model for time-synchronization attacks.

3.1. Time-Synchronization Process. Every node keeps synchronized in TSCH networks, and the communication happens in timeslots (e.g., 15 ms long). Before a new node joins the network, it should receive the enhanced beacon (EB) packets from neighborhood nodes. The contents of EB packets mainly contain Join Priority (JP) value and ASN value. The new node prefers to choose a neighborhood node with a lower JP value to do ASN synchronization.

After the node successfully joins the network, it needs to maintain synchronization. The timeslots of network nodes should remain aligned. Usually, the nodes are equipped with an inexpensive oscillator to keep time. Because of the differences in temperature or fabrication, there is a clock drift between two oscillators (typically 30 ppm). So the nodes need to do device-to-device synchronization to compensate for the clock offset. Every node can use frame-based or acknowledgment-based synchronization methods to synchronize with the network.

Figure 2(a) illustrates a detailed process of frame-based synchronization between a transmitter and receiver. The receiver should turn on the radio a little earlier than the transmitter. The duration is defined as the guard time (GT). And the receiver needs to record the arriving time of the frame. Based on (1), the receiver can get the value of clock offset. And the receiver can synchronize to the transmitter when it updates the period based on (2). Figure 2(b) illustrates a process of acknowledgment-based synchronization. Here, the transmitter is a child node. It should first send a request frame to the receiver. The receiver needs to calculate the time offset based on (1). And the transmitter gets the time offset and synchronizes to the receiver based on (2).

$$Offset = Arrivetime - TsTxoffset \quad (1)$$

$$CurrentPeriod = NormalPeriod + Offset. \quad (2)$$

3.2. Attack Model. The Dolev-Yao threat model is a typical attack model where the attacker can eavesdrop on, modify, or forge communication messages in the network. Our attack model is based on this model. We consider two types of attack: external and internal.

In the external attack model, an attacker can eavesdrop on or modify messages but cannot obtain the secret key. So it is unable to impersonate a legitimate node. But an internal attacker can get a legal identity because it knows the secret key. The compromise attack is a typical internal attack. The compromised node can not only eavesdrop on network messages, but also forge legitimate network messages. Our paper will consider the impact of the two types of attacks on the time-synchronization protocol.

4. Secure Pairwise Time Synchronization

Two neighbor nodes usually adopt single-hop pairwise synchronization to establish relative clock offsets. In this section, we first analyze its security in depth and then propose a security countermeasure that includes authentication mechanisms and a COF algorithm.

4.1. Vulnerabilities in Pairwise Synchronization. ASN synchronization occurs when new nodes join the network. And the EB packet is an important control packet in the process of ASN synchronization. Figure 3 shows the detailed format of the EB packet. The EB packets mainly contain Join Priority (JP) value, ASN value, and channel-hopping template information for a new node to join the network. The EB packet may be encrypted to defend against eavesdrop attacks. It only adopts a shared key for encryption; otherwise, it will prevent new nodes from joining the network. However, from the ASN synchronization perspective, it should adopt a secret key to hide the *ASN* value. If a new node receives a forged EB packet that contains a false ASN value, it may deduce the wrong channel frequency for all the given pairwise communication according to (3). This is the first security vulnerability in the single-hop pairwise synchronization process.

$$frequency = (ASN + channelOffset)\,\%16. \quad (3)$$

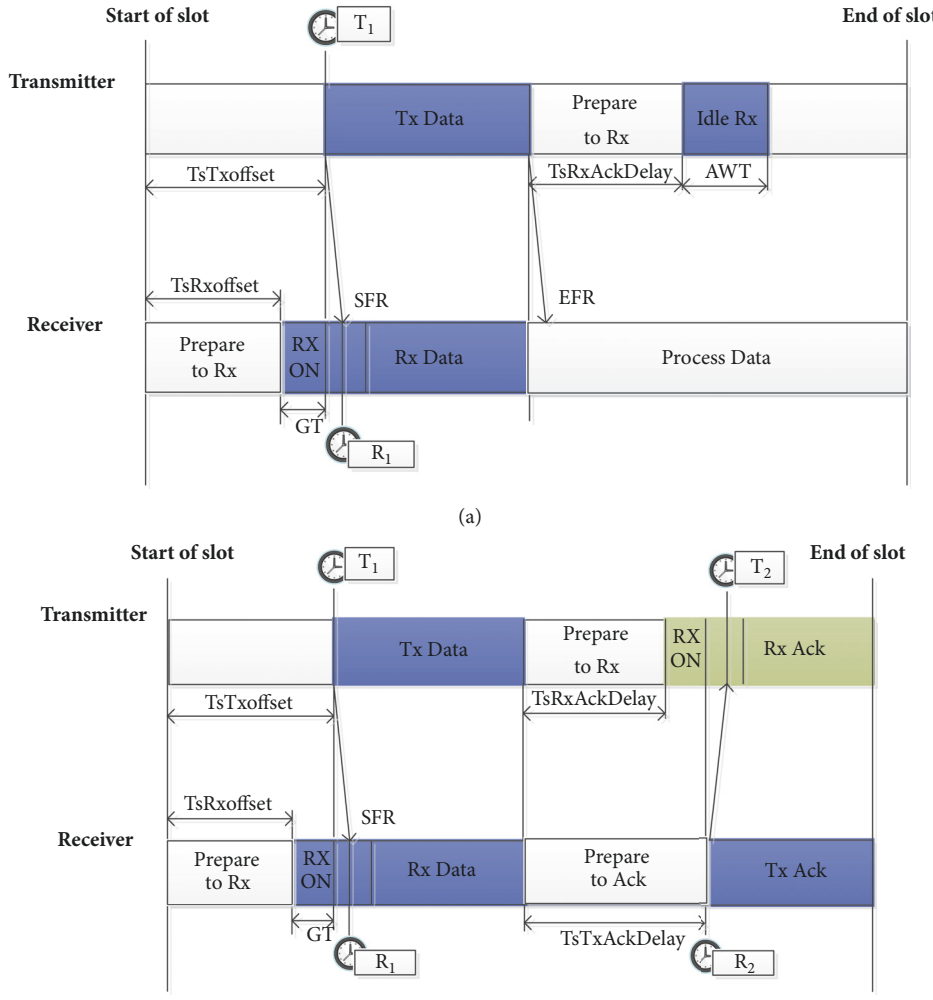


Figure 2: Two methods in device-to-device synchronization. (a) Frame-based synchronization. (b) Acknowledgment-based synchronization.

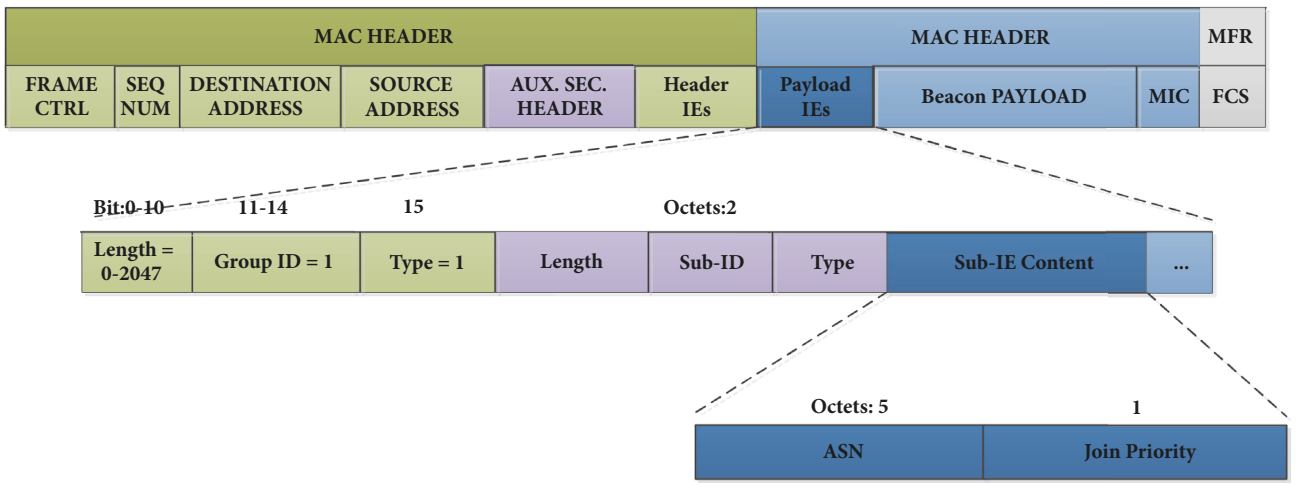


Figure 3: Enhanced beacon packet format.

In the process of device-to-device synchronization, the receiver may deduce the time offset according to (1). However, if an adversary modifies the $TsTxoffset$ value in the timeslot template, the receiver may calculate an incorrect time offset, according to (4). Δt represents the modification size. It is defined as a timeslot-template attack. However, the attacker cannot modify the $TsTxoffset$ value arbitrarily. If the value of Δt is larger than the GT, it may have no effect, as the receiver may turn off the radio in this case. This is the second security vulnerability in the single-hop pairwise synchronization process.

$$offset = timeReceived - TsTxoffset + \Delta t. \quad (4)$$

The pulse-delay attack which was first proposed by Ganeriwal et al. [8] may destroy device-to-device synchronization. To quote from Ganeriwal et al. [8]: "The adversary first sends jamming signals and replays it after a little delay, which will cause the receivers to calculate an incorrect clock offset." Both the frame-based and acknowledgment-based ways may be affected by this attack. This is the third security vulnerability in the single-hop pairwise synchronization process.

4.2. Secure Pairwise Synchronization. In the ASN synchronization process, a new node may receive a forged EB packet that contains a false ASN value. This occurs when joining a network. Fortunately, the IETF 6TiSCH working group recently made some progress in the secure Join protocol [16]. If a new node can securely join a network or choose a legitimate time source, it can successfully do ASN synchronization. Therefore, in our study, we mainly focus on the second and third security vulnerabilities in the pairwise synchronization process.

Generally, all nodes periodically perform device-to-device synchronization with their time parents in TSCH. The offset is usually less than a threshold when the period of synchronization is fixed. Therefore, the criterion of attack success is that the legitimate node receives a clock offset that exceeds the threshold value. Here, we present a mathematical model to describe the device-to-device time-synchronization process (see (5) and (6)). Let $T1$ represent the transmitter-recorded sending time of the synchronization frame and $R1$ represent the receiver-recorded receiving time of the synchronization frame. Similarly, $T2$ and $R2$ represent the relation time. For more detail, refer to Figure 2(b). Using (5) and (6), the time offset and transmission delay are easily calculated.

$$offset = \frac{(R1 - T1) - (T2 - R2)}{2} \quad (5)$$

$$delay = \frac{(R1 - T1) + (T2 - R2)}{2}. \quad (6)$$

First, let us consider the timeslot-template attack where an adversary modifies the value of $TsTxoffset$. Assume that the receiver is a malicious node that is compromised by an adversary. The $TsTxoffset$ value of receiver is modified to $TsTxoffset - \xi$. According to (1), the receiver would calculate a larger offset than the normal case. From the perspective of the mathematical model, the R1 and R2 value increase. Thus, according to (5) and (6), the adversary can conduct the offset increases, as indicated by (7), but the delay does not increase.

$$offset = \frac{(R1 - T1) - (T2 - R2)}{2} + \xi. \quad (7)$$

Second, the pulse-delay attack may destroy pairwise synchronization in TSCH. Assuming the adversary introduces delay Δ. The $T1$, $T2$, and $R2$ value do not change. But the $R1$ value may increase to $R1 + \Delta$. According to (5) and (6), it can conduct the offset increases, as indicated by (8), and the delay increases, as indicated by (9).

$$offset = \frac{(R1 - T1) - (T2 - R2) + \Delta}{2} \quad (8)$$

$$delay = \frac{(R1 - T1) + (T2 - R2) + \Delta}{2}. \quad (9)$$

In order to defend against time-synchronization attacks in the pairwise synchronization in TSCH, a secure algorithm is proposed as shown in Algorithm 1.

Our proposed scheme combines the message-integrity authentication mechanism and COF algorithm. In the process of child node A sending a synchronization request to parent node B, it contains random nonce N_A and MAC (message authentication codes). The random nonce N_A is used to protect against a reply attack. The MAC is an effective way to defend against an external attack which has been proved. However, the adversary may launch an internal attack such as a compromise attack or pulse-delay attack. Here we propose a COF algorithm to protect against the internal attack.

The core idea of the COF algorithm is that the synchronization packet will be filtered out when the clock offset or delay is larger than a threshold value. Therefore, how to accurately estimate the threshold value is a key issue. Let Q represent clock threshold and d^* represent delay threshold. When the period of synchronization T is fixed, the Q value can be estimated based on (10), where Max_drift represent the maximum clock drift between two nodes. The value of Max_drift can refer to the crystal manually. The value of d^* can also be estimated theoretically. Ganeriwal et al. [8] have proved that the transmission delay usually follows a Gaussian distribution. Let d represent the delay in the process of exchanging data packets, which is calculated according to (6). As the delay follows a Gaussian distribution, it can conclude that most of the delay is among $[d_{avg} - 3\sigma, d_{avg} + 3\sigma]$. It can reach 99.7% confidence. So the value of d can be estimated theoretically according to (11).

$$Q \leq T * Max\_drift \quad (10)$$

$$d^* = d_{avg} + 3\sigma. \quad (11)$$

5. Secure Clusterwise Time Synchronization

In this section, we first analyze in detail the vulnerability of clusterwise time synchronization and then propose a lightweight security countermeasure.

1: Node A sends a synchronization request to parent node B:
2: $A \longrightarrow B : A, B, T_1, N_A, MAC(K_{AB}, A, B, T_1, N_A)$;
3: Node B sends back an ACK message to A:
4: $B \longrightarrow A : B, A, T_1, R_1, R_2, N_A, ACK, MAC(K_{AB}, B, A, R_1, R_2, N_A, ACK)$;
5: Node A calculates:
6: $offset = \frac{(R_1 - T_1) - (T_2 - R_2)}{2}$, $delay = \frac{(R_1 - T_1) + (T_2 - R_2)}{2}$;
7: **if** ($offset \leq Q$&&delay <= d*)
8: $CurrentPeriod = NormalPeriod + offset$;
9: **else**
10: Filter out the synchronization packet, $num\_attack + +$;
11: **if** ($num\_attack > N$)
12: Blacklist(B), Send an alarm message to the Root;
13: **end**

Algorithm 1: Secure pairwise time synchronization.

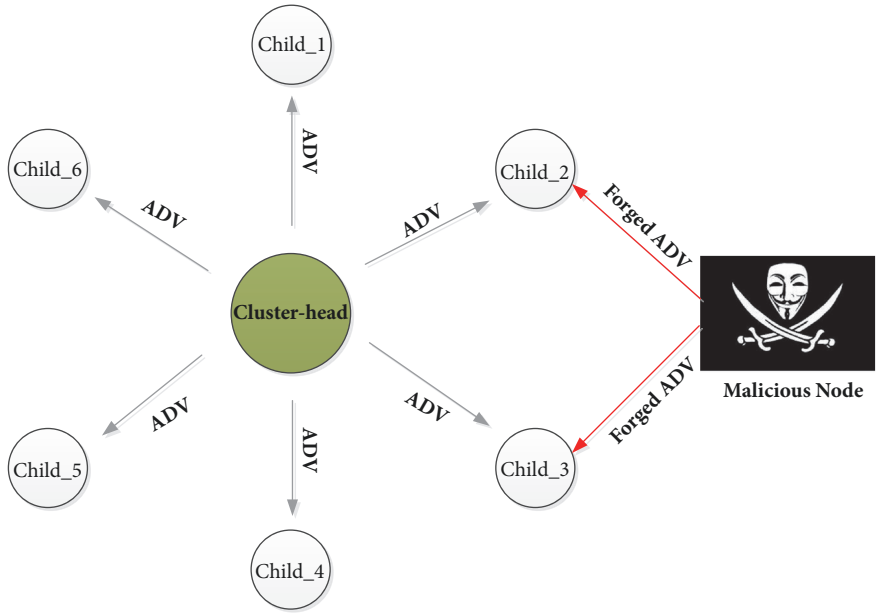


Figure 4: A malicious node can easily forge broadcast time-synchronization packets.

5.1. Vulnerabilities in Clusterwise Synchronization. To improve the energy efficiency, a cluster of nodes in TSCH networks may sleep in the given timeslots and wake simultaneously. Many other WSN applications (e.g., fire monitoring, speed estimating, and movement detection) also need consistent distributed coordination and sensing. Thus, a secure and low-power clusterwise synchronization protocol is critical.

In TSCH networks, a node has two methods for maintaining clusterwise time synchronization: KA-based and ADV-based. In the process of KA-based synchronization, every child node must periodically send a KA packet to the cluster-head to maintain synchronization. ADV-based synchronization is different because the cluster-head needs to periodically broadcast a time-synchronization packet to the child node. Vilajosana et al. [17] observe that the ADV-based synchronization scheme is more practical and energy efficient. Thus, our paper mainly focuses on the security vulnerability of ADV-based synchronization.

The ADV-based synchronization process has no security mechanisms and, in particular, no broadcast-authentication mechanism. A malicious node can easily forge broadcast time-synchronization packets to disturb the clusterwise synchronization.

Figure 4 shows the scenario of a malicious node launching an attack on ADV-based clusterwise synchronization. In the normal case, the cluster-head node should periodically broadcast an ADV packet. All the child nodes receive the ADV packet to do synchronization. However, the malicious node launches a time-synchronization attack on child_2 and child_3 by forging an ADV packet with a strong power. Child_2 and child_3 will calculate a false time offset and may lose synchronization with the cluster-head.

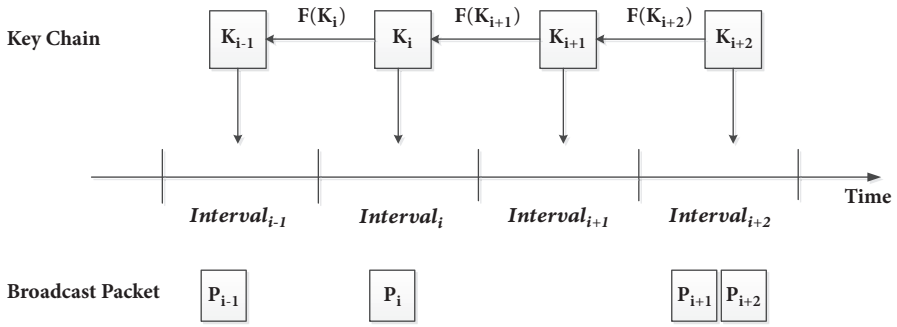


Figure 5: μTESLA broadcast-authentication mechanism.

In TSCH networks, all communications are based on high-precision synchronization. A false time offset can effectively reduce the performance of a cluster network. Thus, a secure clusterwise synchronization protocol should be designed for the TSCH network.

5.2. Secure Clusterwise Synchronization. In TSCH networks, the ADV-based synchronization scheme is more energy efficient than the KA-based synchronization scheme at making a cluster of nodes' synchronization. However, it has a serious security vulnerability. As it lacks a broadcast-authentication mechanism, a malicious node can easily impersonate a cluster to send false ADV packets. Therefore, it is necessary to adopt a broadcast-authentication mechanism to secure the network against this vulnerability.

Generally, digital signatures [18] and μTESLA [19] are typical mechanisms for authenticating broadcast packets in WSNs. Many studies have shown that digital signatures are computationally expensive, which cannot be used in resource-constrained WSNs. The μTESLA scheme is based on symmetric cryptography, which introduces low computation and communication overheads. However, the original μTESLA scheme exhibits several shortcomings.

One critical shortcoming is that the receivers must wait to authenticate packets until the cluster-head discloses the key. The longer delay means the receivers need more memory to buffer the synchronization packets. It also increases the risk of denial-of-service (DoS) attacks. We propose an improved μTESLA scheme in which receivers do not need to delay or buffer synchronization packets. In the following, we first review μTESLA and then propose an improved μTESLA scheme that allows receivers to authenticate most packets immediately upon arrival.

5.2.1. Overview of μTESLA. The core idea of μTESLA is that every broadcast packet attaches a MAC which is hard to forge. The child nodes first buffer the incoming packets until the cluster-head discloses the secret key. Only the cluster-head owns the secret key. A malicious node cannot forge a legitimate MAC unless it obtains the secret key K. Figure 5 illustrates the original μTESLA broadcast-authentication mechanisms. In μTESLA, the time axis is divided into many equal-length time intervals. Every interval is assigned a different secret key. When one or more broadcast packets are generated in a time interval, it uses the assigned secret key for authentication.

Usually, a hash function is adopted to generate a one-way key chain. According to (12), all the other keys can be calculated when the last key K_n is selected. Depending on the mathematical properties of the hash function, it is easy to authenticate the current disclosed key by previous keys, but it is hard to forge the later keys.

$$Key_i = Hash\left(Key_i + 1\right), \quad (i = 0, 1 \ldots n). \tag{12}$$

5.2.2. Improved μTESLA Scheme. The shortcoming of the original μTESLA scheme is that the receivers must wait until they receive the disclosed key before they authenticate the packets. Buffering packets may incur serious DoS attacks and might not be practical for secure time-synchronization applicants. We propose an improved μTESLA scheme that supports immediate authentication. Its main idea is that the cluster-head needs to buffer some packets during an interval and send the current packet with the hash value of a later packet. The receivers can then immediately authenticate current packets according to the hash value of the earlier packet.

Figure 6 shows the improved μTESLA scheme for secure clusterwise time synchronization in TSCH networks. The time is composed of a repeated super-frame. The super-frame is divided into many equal-length timeslots (15 ms). Packet P_j contains M_j, H(M_{j+1}), MAC (K_{i-1}, D_j), and K_{i-1}. Packet P_{j+1} contains M_{j+1}, H(M_{j+2}), MAC (K_i, D_{j+1}), and K_i. When the receivers receive packet P_{j+1}, they can immediately authenticate the current packet using H(M_{j+1}) of the earlier packet.

6. Secure Multihop Time Synchronization

In this section, we first briefly present the multihop time synchronization in TSCH and then perform an in-depth security analysis. Finally, we propose a secure multihop synchronization method that adopts a multipath scheme based on trust modeling.

6.1. Multihop Synchronization in TSCH Networks. The synchronization protocol introduced in the above sections aims to provide a clock offset between neighborhoods of nodes. Multihop TSCH networks usually consist of hundreds of

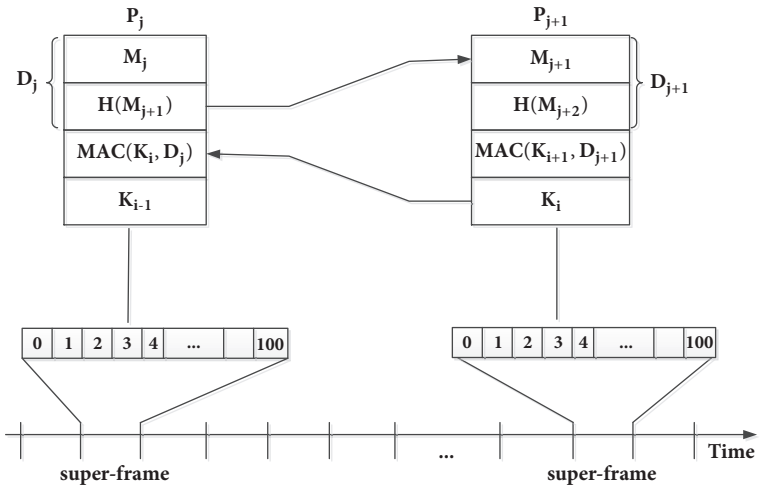


Figure 6: Improved μTESLA for secure clusterwise time synchronization.

nodes, and the distant child nodes must synchronize with the network root. In the multihop time-synchronization process, building a hierarchical network topology is a critical problem. The IETF 6TiSCH working group recommends the RPL (routing protocol for low power and lossy networks) [20] (see below) as a routing protocol to build the time-synchronization tree. The nodes far away from the root can synchronize step by step along the path of the time-synchronization tree.

Here, we briefly review the RPL; for more detail, refer to [20]. The RPL is an IPv6 routing protocol for low-power and lossy networks (LLNs) where both the routers and sensor nodes are resource-constrained. It is a distance-vector routing protocol. The RPL organizes the network topology as a Destination-Oriented Directed Acyclic Graph (DODAG) using Objective Functions (OFs). IETF 6TiSCH suggests reusing the DODAG structure: i.e., a node's routing parent is also its clock source. It not only prevents synchronization loops, but also reduces the energy cost to construct the synchronization tree.

6.2. Vulnerabilities in Multihop Synchronization. In multihop TSCH networks, the RPL plays a key role in constructing the time-synchronization tree. The distant child nodes synchronize along the edges of the tree. Having the procedure go smoothly relies on the assumption that none of the side nodes are malicious. Even a single malicious node along the multihop path can affect the time-synchronization performance. Unfortunately, the current RPL is not resilient against all of the attacks, especially the compromise attack. The adversary can easily control or instruct the compromised nodes to damage the time-synchronization services in an Internet-of-Things (IoT) environment. This may cause an error-accumulation phenomenon in the multihop time synchronization.

If a compromised node brings an incorrect offset to its immediate child node, the incorrect offset will propagate along the time-synchronization tree. Because the error-accumulation phenomenon exists in multihop networks, this attack may seriously damage the performance of TSCH networks, which require high-precision synchronization. In this paper, we define this attack as an error-accumulation attack.

Figure 7 illustrates the impact of an error-accumulation attack on multihop synchronization in a TSCH network. The network consists of five nodes. The root node is the reference clock source for the network. Node A1 is the immediate child node of the root. And node A2 is the immediate child node of A1, etc. As the phenomenon of clock drift exists, nodes need to synchronize periodically. Right before synchronizing, a clock offset exists between the node and its clock source (e.g., 200 s). Assume the GT is set to 1,000 s. This means the node may lose synchronization if the clock offset is bigger than 1,000 s.

We assume that, at some point, each node will synchronize just before its clock-source neighbor does. The phenomenon of the top half of Figure 7(a) may occur; i.e., the clock offset is 200 s between each node and its clock source. Let us further assume that the nodes synchronize from left to right; i.e., A1 synchronizes with the root, and then A2 synchronizes with A1. After the node A3 synchronization, the clock offset is about 800 s between A4 and A3. Because the clock offset is lower than the GT, it can successfully synchronize.

In the malicious setting, node A1 is a compromised node, as illustrated in Figure 7(b). The clock offset between compromised node A1 and the root may be 600 s if A1 waits a long time before synchronizing. The clock offset for each pair of legitimate neighbor nodes is 200 s. If the nodes synchronize from left to right, the last node, A4, may lose synchronization. After the node A3 synchronization, the clock offset between nodes A3 and A4 reaches 1,200 s (larger than the GT). This example fully shows the vulnerability of multihop synchronization in TSCH networks. The security of multihop synchronization must be enhanced for TSCH networks to be resilient against attacks.

6.3. Secure Multihop Synchronization. The error-accumulation attack seriously threatens multihop synchronization in TSCH networks. To effectively defend against such attacks,

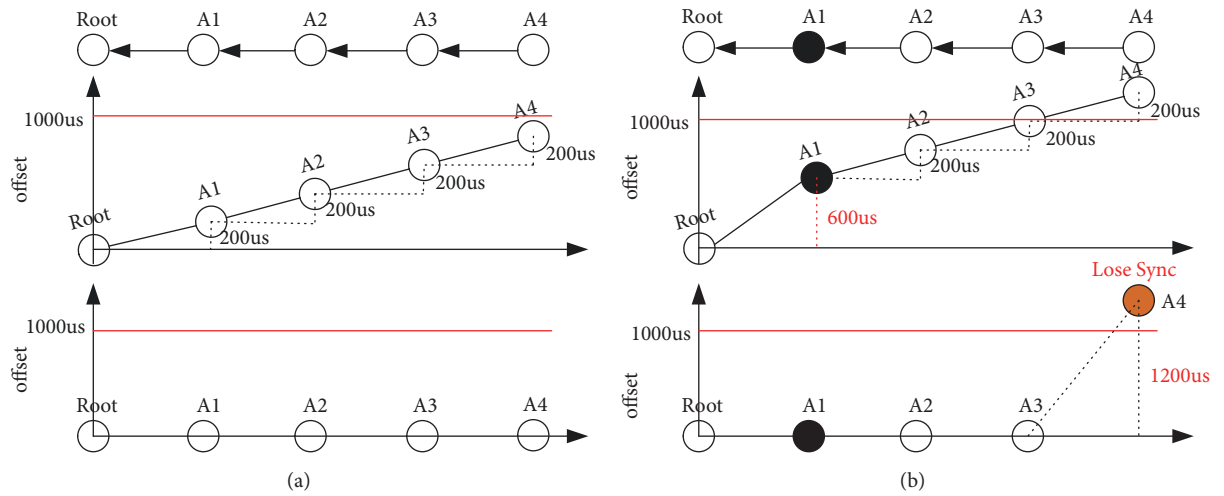


Figure 7: Error-accumulation attack on multihop synchronization in TSCH. (a) In the nonmalicious setting. (b) Under the malicious setting.

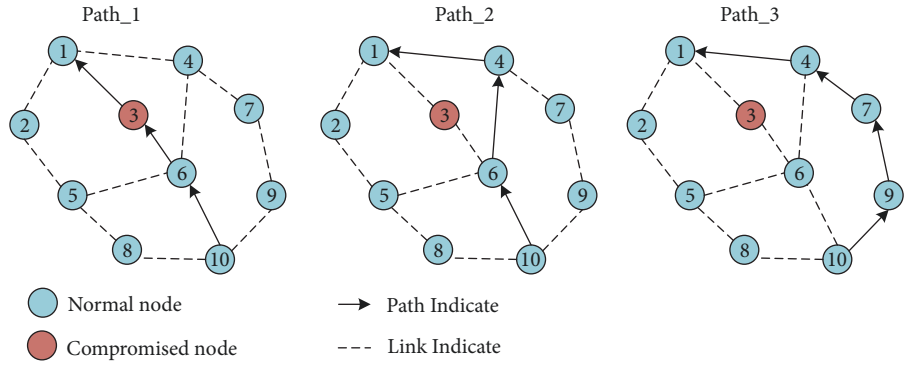


Figure 8: Diagram of multipath-based time synchronization.

we propose a multipath time-synchronization scheme based on trust modeling. The scheme can ensure that the nodes can find a secure path to the root nodes by establishing a trust model. Figure 8 shows node 10 finding a secure path to node 1, bypassing compromised node 3. If node 10 chooses Path_1 for synchronization, compromised node 3 can easily damage the multihop synchronization. If node 10 chooses Path_2 or Path_3 for synchronization, it will be successful. Thus, choosing a secure path from multihop paths is a challenge. Our paper proposes a scheme based on a trust model to solve this problem.

Here, we adopt an entropy-based trust model proposed by Sun et al. [21]. They proposed four axioms and derived a method for calculating a trust value, as follows:

$$P\{A : X, action\} = \frac{1 + \sum_{j=1}^{I} \beta^{t_c - t_j} k_j}{2 + \sum_{j=1}^{I} \beta^{t_c - t_j} N_j}, \tag{13}$$

where t_j represents the observation time, t_c represents the current time, k_j and N_j represent action times, and $0 \le \beta \le 1$ is the remembering factor.

The trust model quantifies the trust relationship between nodes. The trust values can be used to determine whether a node is legitimate or malicious. When a node has multiple neighbor nodes, it should select the node which owns the maximum trust value as the forward node. This way, it can bypass malicious nodes. Let $numTx\_A$ represent the total number of packets sent by node A, $numTx\_K$ represent the number of packets forwarded through node K, and $numTx\_K\_ACK$ represent the number of packets successfully forwarded through node K.

In our paper, the source of the nodes trust value is mainly considered from two aspects. One is the packet-forwarding rate $rate1 = numTx\_K/numTx\_A$; it reflects the cooperative relationship of the nodes. The other is the packet-forwarding success rate $rate2 = numTx\_K\_ACK/numTx\_K$. The trust value of a node is $T^d = \theta_1 T^w + \theta_2 T^r$, where $\theta_1, \theta_2 \in [0, 1]$ and $\theta 1 + \theta 2 = 1$. T^w can be calculated according to (14), and T^r can be calculated according to (15).

$$\begin{aligned} T^w &= P\{A : K, rate1\} \\ &= \frac{1 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K}{2 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_A} \end{aligned} \tag{14}$$

$$\begin{aligned} T^r &= P\{A : K, rate2\} \\ &= \frac{1 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K\_ACK}{2 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K}. \end{aligned} \tag{15}$$

Node A observes node K at frequent intervals. $G_A(t_j)$ represents the observation time of node A, where $j = 1, 2\cdots, I$. $G_A(t_c)$ represents the current time. β is the remembering factor. The entropy-based trust model can bypass malicious nodes and find a secure path between a distant node and the root node. However, the value of β needs to be adjusted according to practical network applications.

7. Experiment Evaluation

In this section, we perform an evaluation through real experiments on a network of 16 OpenMoteSTM nodes running OpenWSN [22]. In the following, we first introduce an OpenMoteSTM node, the OpenWSN software, and the experimental setup. Then, we show the performance evaluation result of the proposed scheme.

7.1. OpenMoteSTM Nodes and OpenWSN Software. To verify the effectiveness of the proposed scheme, a low-power wireless sensor node called OpenMoteSTM was designed. Figure 9 shows the detailed structure of an OpenMoteSTM node. It incorporates a high-performance ARM-based 32-bit microcontroller and a short range radio. The microcontroller (STM32F103) can operate at 72 MHz, with 64 kB of embedded SRAM, and 512 kB of flash. It has a variety of peripherals, e.g., GPIOs, UART, SPI, ADC, and a timer.

The radio (AT86RF231) operates in 2.4 GHz and fully supports the IEEE802.15.4-2006 standard. The OpenMoteSTM node is also equipped with a 32.768 kHz crystal to drive the microcontroller timers. Because of the difference in manufacturing and working temperatures, the crystal has a typical drift of up to 30 ppm.

OpenWSN [22] is open-source software developed by UCB which support the IEEE802.15.4e TSCH protocol. The original OpenWSN implements time synchronization in TSCH but has no security mechanisms. Currently, OpenWSN can support a variety of hardware platforms, e.g., Guidance and Inertial Navigation Assistant (GINA), TelosB, and OpenMoteSTM nodes. It also offers the OpenVisualizer software, which can show the internal state (neighbor table, scheduling table, error reports, etc.) of each node in the network.

7.2. Experimental Setup. Figure 10 shows the test-bed scenario. The OpenMoteSTM nodes, which hang from the testing shelf, are all connected to the USB hub by USB lines. It not only provides power to every node, but also can view the state of the nodes through the OpenVisualizer software.

OpenWSN implements a time-synchronization protocol in TSCH. It adopts a hard-coded resource-scheduling algorithm [23], where a slotframe consists of 11 timeslots. The first timeslot is allocated for EB packets. And there are five shared timeslots allocated for data packets. The period for sending EB packets is set to 10 s. However, the original OpenWSN does not adopt any security mechanisms. We added code to implement the proposed security scheme on the OpenWSN platform.

7.3. Performance Evaluation. Here we adopt the following metrics for performance evaluation: synchronization error, energy consumption, and synchronization rate. The synchronization error reflects the synchronization precision in the process of synchronization. The energy consumption is a key metric in resource-constrained WSNs, which directly determines the feasibility of the scheme. The synchronization rate can reflect the percentage of nodes successfully synchronizing to the networks. Next, we will carefully analyze the performance of the secure pairwise and multihop synchronization.

7.3.1. Secure Pairwise Time Synchronization. We adopted a pair of OpenMoteSTM nodes to evaluate the performance of secure pairwise time synchronization. We introduce node A and node B. Node A needs to synchronize with node B at a period of 5 s. The GT is set to 1 ms. Assuming the adversary intermittently launches a pulse-delay attack, the synchronization error may vary with time. In real deployment, the attack parameter Δ is set to 0.8 ms.

Figure 11 shows that the comparison of synchronization error values in both the nonmalicious and malicious settings. In the nonmalicious setting, the value on the vertical axis is almost below 0.1 ms, in both the original and our proposed scheme. This illustrates that our proposed scheme does not affect the synchronization error. And we used a green wave line to show the fluctuation of the synchronization error. The synchronization error may increase with time because of the clock drift of each pair of nodes. After synchronizing, child node A compensates its clock according to (2). Thus, the synchronization error is reduced to a very small value.

In the malicious setting (where Δt is set to 0.8 ms), the synchronization error is very different between the original and our proposed scheme. Figure 11 shows the maximum value of the synchronization error almost reaches 0.9 ms in the original scheme. However, it only rises slightly in our proposed scheme because our proposed scheme can successfully detect the delay attack using the threshold-filter mechanism. As we know, the period of synchronization in our experiment is 5 s. According to (10), threshold Q is calculated to be 0.3 ms when Max_drift sets 60 ppm (a typical value). In our proposed scheme, when node A receives a synchronization packet from the malicious node, whose clock offset is larger than threshold Q, it will ignore the packet and increase the number of attacks. In the real experiment, the attack parameter Δt is 0.8 ms, larger than the threshold Q. Therefore, the synchronization packet will be ignored, and the synchronization error may go up a little with the clock drift of the pair of nodes. The experimental results not only show the impact of the attack, but also validate the effectiveness of our proposed scheme.

Energy consumption is a key metric in resource-constrained WSNs, which directly determines the feasibility of our scheme. The secure pairwise synchronization proposed in our paper adopts encryption and authentication mechanisms to protect against attacks. In the ASN synchronization process, an EB packet may be encrypted using a shared key to defend against eavesdrop attacks. The device-to-device

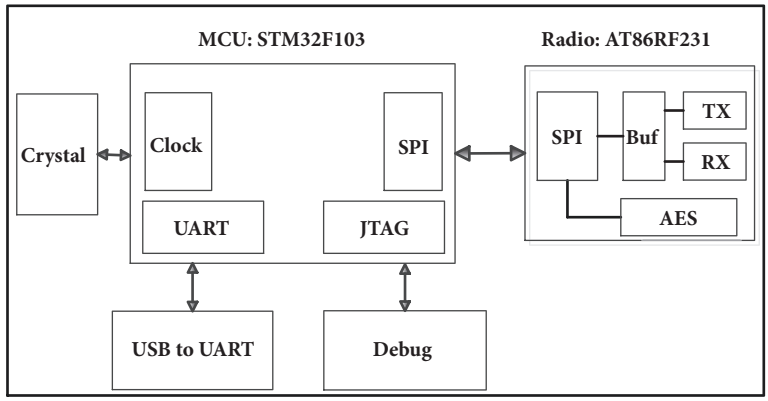


Figure 9: Structure of an OpenMoteSTM node.

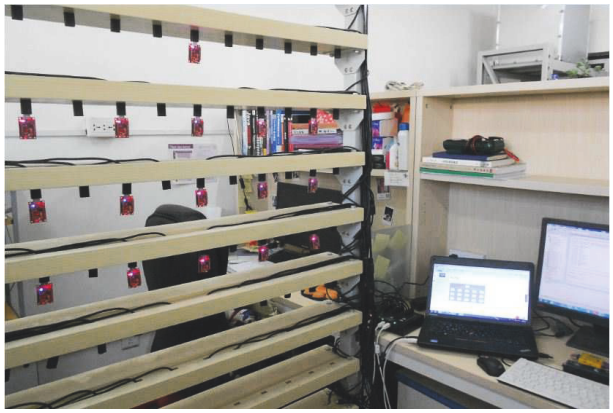


Figure 10: Test-bed implementation of time synchronization in TSCH networks.

Table 1: Energy consumption in different security modes.

Security Mode	Energy
No Security	156 μJ
Encryption	171 μJ
Authentication	192 μJ
Encryption + Authentication	207 μJ

synchronization process needs message-integrity authentication to defend against attacks. OpenMoteSTM nodes adopt an AT86RF231 radio, which supports hardware-accelerated encryption and authentication. The AT86RF231 radio uses the Electronic Codebook (ECB) mode to encrypt messages and the Cipher Block Chaining (CBC) mode to generate the MAC.

The typical operating voltage U of a node is 3.3 V. The operating current can be measured by a current probe. The energy consumption can be calculated according to

$$E = U * I * T. \quad (16)$$

Table 1 illustrates the energy consumption in different security modes when the node transmits a 16-byte message. The energy consumption in the encryption mode only increases 9% compared to the no-security mode. However, it

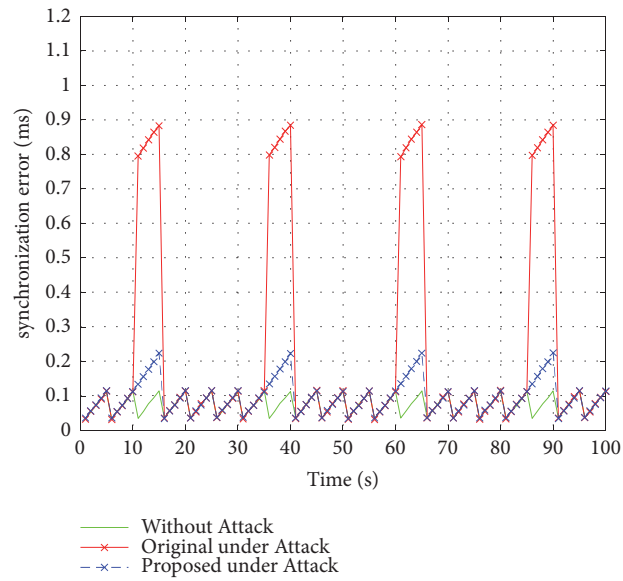


Figure 11: The comparison of synchronization error values under the different scenarios.

may increase 23% in the authentication mode. Generally, the device-to-device synchronization period is about 16 s. Thus, the energy consumption may not increase significantly from the network life cycle.

7.3.2. Secure Multihop Time Synchronization. We implemented a test bed as illustrated in Figure 10 to evaluate the effectiveness of secure multihop time synchronization. The real experiment uses 16 OpenMoteSTM nodes to build a three-hop-deep wireless network. The multihop network adopts the RPL to build the time-synchronization tree.

In the nonmalicious setting, all nodes are legitimate. The nodes far away from the root can synchronize step by step along the path of the time-synchronization tree. The pairs of neighborhood nodes use acknowledgment-based synchronization. The period of synchronization is set to 10 s and the GT is set to 1 ms. In the malicious setting, a compromised node launches an error-accumulation attack.

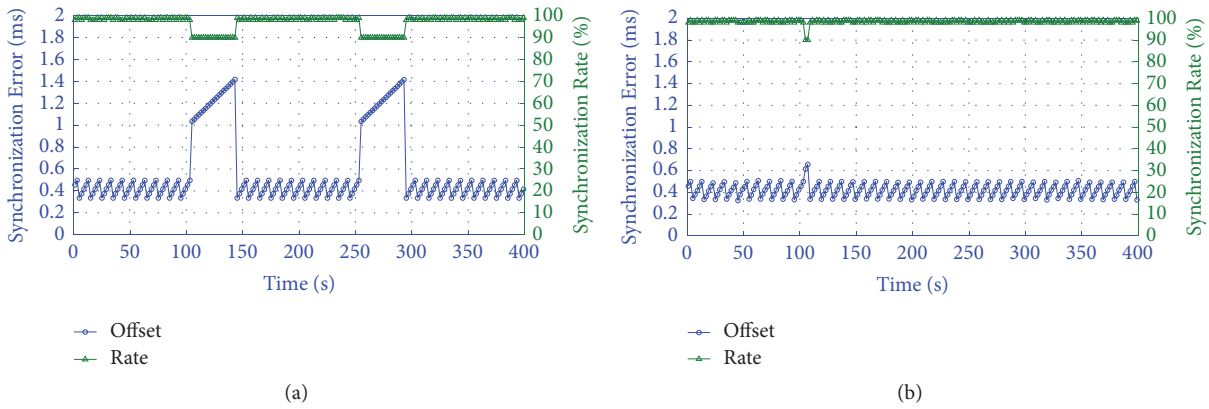


Figure 12: The synchronization error and synchronization rate vary with time in malicious settings. (a) Original scheme. (b) Our proposed scheme.

In real deployment, the compromised node intermittently launches attacks.

Figure 12 illustrates that the synchronization error and synchronization rate vary with time in malicious settings. The experiments were run 30 times and the average value was calculated. Figure 12(a) indicates the fluctuation of the synchronization error of the nodes and the synchronization rate in the original scheme. In the 0-100 s interval, the average synchronization error of the network stayed in a relatively stable state and the value was less than 0.55 ms. The synchronization rate almost reaches 99%. It indicates all nodes synchronize to the network. In the 100-150 s interval, the malicious node launches an error-accumulation attack. The average synchronization error of the network increases rapidly. And the synchronization rate drops to about 90% as some nodes have lost synchronization. In the 150-200 s interval, the synchronization error and synchronization rate return to the normal level as the malicious node stops attacking. Figure 12(b) indicates the fluctuation of the synchronization error of the nodes and the synchronization rate in our proposed scheme. In the 0-100 s, the average synchronization error and synchronization rate matched those for the original scheme. It illustrates that our proposed scheme does not affect the synchronization error. In the 100-150 s interval, the malicious node launches an error-accumulation attack. Compared to the original scheme, the impact of attacks on the synchronization error and network synchronization rate is much smaller. And in the 150-400 s interval, the synchronization error and network synchronization rate stay in the normal level, because our proposed scheme can ensure that the nodes can bypass the malicious node and find a secure path to the root nodes.

8. Comparison with Related Protocols

Time-synchronization protocols in WSNs usually divide into two categories: centralized synchronization and distributed synchronization. And secure time-synchronization protocols can also be divided into two categories. Table 2 shows the comparison of different secure time-synchronization protocols. He et al. [12] analyzed the vulnerability of the ATS protocol and described message-manipulation attacks. They then proposed a SATS (secure ATS) protocol using hardware and logical clocks to detect the attacks. The result showed that the SATS protocol can successfully defend against message-manipulation attacks. Ganeriwal et al. [8] pointed out the possible attacks on TPSN, such as a forge and modify attack, pulse-delay attack, and compromise attack. These attacks can mislead legitimate nodes to calculate an error clock offset. They then proposed a SPS protocol, which adopted message-authentication and end-to-end delay-estimation methods, to defend against the above attacks. The experiments showed that SPS can successfully protect against the attacks. Maximum-consensus-based time synchronization (MTS) is a typical distributed synchronization in WSNs. An adversary may easily destroy the MTS protocol by launching message-manipulation attacks. The noise-resilient MTS (NMTS) protocol [15], which adopted skew estimation, clock skew, and offset compensation mechanisms, can successfully defend against the attacks. However, the above secure protocols cannot be directly applied to the time-synchronization protocol in TSCH networks. The time-synchronization protocol in TSCH networks is different from the ATS, MTS, and TPSN protocol. There exist difference security vulnerabilities in the process of time synchronization. To the best of our knowledge, the current article is the first to provide an in-depth security analysis of the time synchronization in TSCH networks. We conducted an in-depth security analysis of the single-hop pairwise, clusterwise, and multihop time synchronization. The adversary may easily destroy the time-synchronization protocol by launching the reply attack, pulse-delay attack, timeslot-template attack, and error-accumulation attack. Security countermeasures, which include authentication mechanisms, the COF algorithm, improved μTESLA, and the multipath approach based on trust modeling, were proposed to protect against those attacks. The experimental results validated the effectiveness and feasibility of the security countermeasures. Moreover, the energy cost of our proposed protocol is very low.

9. Conclusions

The current time-synchronization protocol in TSCH networks has security vulnerabilities and is easily exploited by adversaries. We developed a suite of protocols for secure

Table 2: Comparison of different secure time synchronization protocols.

Secure Protocols	SATS [12]	SPS [8]	NMTS [15]	Our Proposed
Time Synchronization	ATS (Distributed Synchronization)	TPSN (Centralized Synchronization)	MTS (Distributed Synchronization)	TSCH (Centralized Synchronization)
Vulnerabilities	Reply Attack Pulse-Delay Attack Message Manipulation Attack	Forge/Modify Attack Pulse-Delay Attack Compromise Attack	Pulse-Delay Attack Message Manipulation Attack	Reply Attack Pulse-Delay Attack Timeslot-template Attack Error-accumulation Attack
Countermeasures	Logical Clock Checking Hardware Clock Checking	Encryption Authentication μTESLA	Maximum Consensus Skew Estimation Offset Compensation	Authentication COF Algorithm Improved μTESLA Multipath Approach
Energy Cost	High	Medium	High	Low

pairwise, secure clusterwise, and secure multihop time synchronization.

We conducted an in-depth security analysis of the single-hop pairwise time synchronization and proposed a security countermeasure that includes authentication mechanisms and a COF algorithm. The COF algorithm produced a filter based on a typical clock model to filter out time-synchronization packets from malicious nodes; it was used to defend against timeslot-template attacks. The experimental results showed that the synchronization error was very low, even in the malicious settings.

In clusterwise time synchronization, an adversary can easily forge broadcast time-synchronization packets to disturb normal synchronization. We proposed an improved μTESLA scheme that supports immediate authentication. The receivers did not need to wait a long time to receive the disclosed key and authenticate the packets. We provided an in-depth security analysis of multihop time synchronization and described an error-accumulation attack. We proposed a multipath approach based on trust modeling, which could find a secure path to the root node by establishing a trust model between nodes. Finally, a test bed using 16 OpenMoteSTM nodes and the OpenWSN software was built. The experimental results validated the effectiveness and feasibility of the security countermeasures.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 61741125, the Natural Science Foundation of Jiangxi Province under Grant No. 20171BAB212014, and the National High Technology Research and Development Program under Grant No. 2014AA041801-2.

References

[1] Z. Sheng, C. Mahapatra, C. Zhu, and V. C. Leung, "Recent advances in industrial wireless sensor networks toward efficient management in IoT," *IEEE Access*, vol. 3, pp. 622–637, 2015.

[2] Q.-P. Chi, H.-R. Yan, C. Zhang, Z.-B. Pang, and L. D. Xu, "A reconfigurable smart sensor interface for industrial WSN in IoT environment," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1417–1425, 2014.

[3] G. Han, L. Liu, J. Jiang, L. Shu, and G. Hancke, "Analysis of energy-efficient connected target coverage algorithms for industrial wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 135–143, 2017.

[4] IEEE Standard 802.15.4-2006, "IEEE Standard for Information Technology, Local and Metropolitan Area Networks, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)," 2006.

[5] International Society of Automation, "ISA-100.11a-2011: Wireless systems for industrial automation: process control and related applications," 2011.

[6] HART Communication Foundation, "WirelessHART Specification 75: TDMA Data-Link Layer," *IEEE Transactions on Vehicular Technology*, 2008.

[7] IEEE Standard 802.15.4-2015, "IEEE 802.15.4-2015: IEEE Approved Draft Standard for Low-Rate Wireless Personal Area Networks (WPANs)," 2015.

[8] S. Ganeriwal, C. Pöpper, S. Čapkun, and M. B. Srivastava, "Secure Time Synchronization in Sensor Networks," *ACM Transactions on Information and System Security*, vol. 11, no. 4, pp. 23–57, 2008.

[9] J. He, J. Chen, P. Cheng et al., "Secure time synchronization in wireless sensor networks: A maximum consensus-based approach," *IEEE Transactions on Parallel & Distributed Systems*, vol. 25, no. 4, pp. 1055–1065, 2014.

[10] D.-J. Huang, W.-C. Teng, and K.-T. Yang, "Secured flooding time synchronization protocol with moderator," *International Journal of Communication Systems*, vol. 26, no. 9, pp. 1092–1115, 2013.

[11] Z. Zhang, S. Gong, A. D. Dimitrovski, and H. Li, "Time synchronization attack in smart grid: Impact and analysis," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 87–98, 2013.

[12] J. He, P. Cheng, L. Shi, and J. Chen, "SATS: secure average-consensus-based time synchronization in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 61, no. 24, pp. 6387–6400, 2013.

[13] W. Dong and X. Liu, "Robust and secure time-synchronization against Sybil attacks for sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1482–1491, 2015.

[14] W. Yang, Y. Wan, and Q. Wang, "Enhanced secure time synchronisation protocol for IEEE802.15.4e-based industrial Internet of Things," *IET Information Security*, vol. 11, no. 6, pp. 369–376, 2017.

[15] J. He, X. Duan, P. Cheng, L. Shi, and L. Cai, "Distributed time synchronization under bounded noise in wireless sensor networks," in *Proceedings of the 2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, pp. 6883–6888, Los Angeles, CA, USA, December 2014.

[16] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: Deterministic IP-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.

[17] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister, "A realistic energy consumption model for TSCH networks," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 482–489, 2014.

[18] K.-A. Shim, Y.-R. Lee, and C.-M. Park, "EIBAS: an efficient identity-based broadcast authentication scheme in wireless sensor networks," *Ad Hoc Networks*, vol. 11, no. 1, pp. 182–189, 2013.

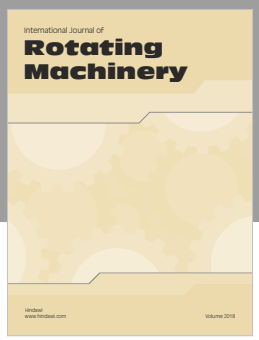
[19] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4554–4564, 2009.

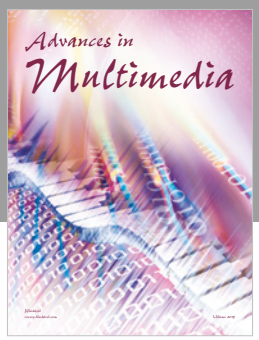
[20] T. Winter, P. Thubert, A. Brandt et al., "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC Editor RFC6550, 2012.

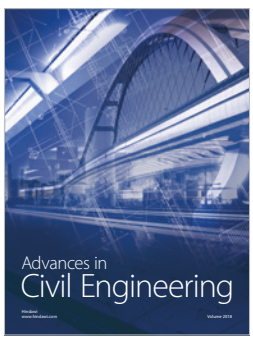
[21] Y. L. Sun, W. Yu, and Z. Han, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 305–315, 2006.

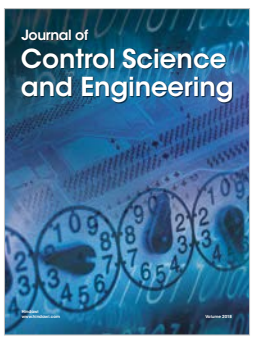
[22] T. Watteyne, X. Vilajosana, B. Kerkez et al., "OpenWSN: a standards-based low-power wireless development environment," *European Transactions on Telecommunications*, vol. 23, no. 5, pp. 480–493, 2012.

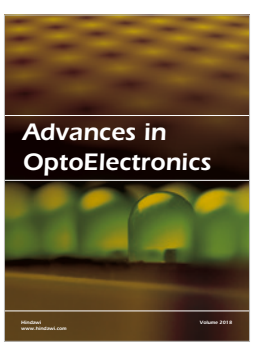
[23] T. Zhang, T. Gong, C. Gu et al., "Distributed Dynamic Packet Scheduling for Handling Disturbances in Real-Time Wireless Networks," in *Proceedings of the 2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 261–272, Pittsburg, PA, USA, April 2017.

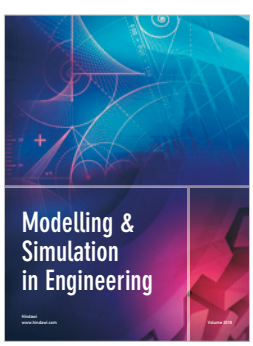


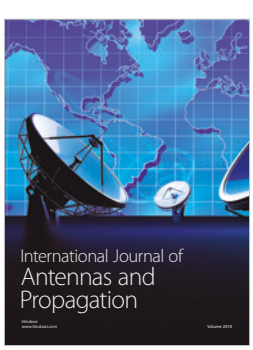


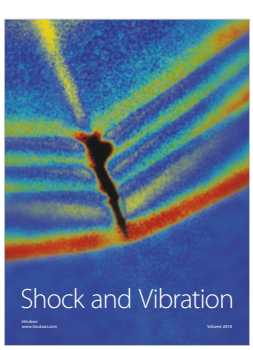


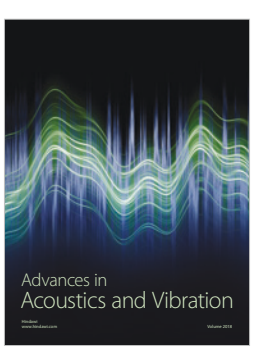












Hindawi

Submit your manuscripts at
www.hindawi.com