IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 18, 2019, accepted August 13, 2019, date of publication August 20, 2019, date of current version August 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2936448

Resilient Consensus-Based Time Synchronization in Asynchronous Sensor Networks

SHULING HUANG[1], NING ZHENG[1,2], YIMING WU[2], AND MING XU[2]
[1]School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China
[2]School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Yiming Wu (yimgwu@hotmail.com)

This work was supported in part by the Cyberspace Security Major Program of the National Key Research and Development Plan of China under Grant 2016YFB0800201, in part by the Natural Science Foundation of China under Grant 61803135, Grant 61702150, and Grant 61572165, in part by the Zhejiang Provincial Basic Public Welfare Research Project under Grant LGG18F020015, in part by the State Key Program of Zhejiang Province Natural Science Foundation of China under Grant LZ15F020003, and in part by the Key Research and Development Plan Project of Zhejiang Province under Grant 2017C01065.

ABSTRACT This paper addresses the time synchronization problem of asynchronous wireless sensor networks (WSNs) under false data injection attacks. First, we extend the fundamental metric of r-robustness with the notion of trusted links and then show that the structural robustness of networks can be significantly improved without adding additional links after setting a small subset of links to be trusted. Then, for a more practical case where the intercommunication topologies between each sensor node are time-varying, we develop a resilient consensus-based distributed algorithm where the sampling period is allowed to be chosen by each sensor independently and it is shown that the time synchronization problem for WSNs can be solved via the proposed controller. Numerical simulations are conducted to illustrate the effectiveness of our proposed method.

INDEX TERMS Resilient consensus, false data injection attacks, time synchronization, asynchronous, time-varying, trusted link.

I. INTRODUCTION

In wireless sensor networks (WSNs), the precise time synchronization has become a fundamental requirement for various applications such as energy saving, trajectory estimation of mobile objects, surveillance coverage optimizing, and environment monitoring [1]–[4]. Many time synchronization control approaches have been developed for WSNs in various scenarios in the past years [5]–[9].

However, most existing works are based on the assumption that the WSNs are deployed in a benign environment. As the small-size, low-cost as well as wireless communication requirement of sensor nodes, the present WSNs are quite vulnerable to cyber attacks. Also, the existing time synchronization mechanisms may become vulnerable or even invalid in the presence of attacks. For example, in a WSN without considering security defense mechanism, exchanging messages through wireless communication links are vulnerable to attackers who aim to inject corrupted messages into the network. Its impact on existing time synchronization protocols is devastating since such corrupted information will be propagated in an epidemic way. To avoid the impact of such malicious attacks, it is therefore important to develop secure time synchronization strategies.

Recently, the time synchronization based on consensus algorithms for WSNs under attacks has been explored in [8], [10]–[14]. A common approach that filters the information received from one's neighbors to ensure robustness has been introduced for the study of resilient consensus problem in recent works [15]–[18], and has been extended a family of algorithms, called the Weighted Mean-Subsequence-Reduced (W-MSR) algorithm [19]. The basic idea of W-MSR is to eliminate the constant number of maximum and minimum values that each node received from its neighbors. He *et al.* [20] proposed a secured maximum consensus-based time synchronization (SMTS) protocol for WSNs to detect and invalidate message manipulation attacks. Dong and Liu [21] considered the Sybil attack, which is one typical attack on such sensor networks, and proposed a robust and secure time-synchronization (RTSP)

The associate editor coordinating the review of this article and approving it for publication was Zheng Yan.

protocol. More recently, the authors in [8] considered the resilient consensus-based distributed time synchronization under both false data attacks and unreliable communication. Similarly, the authors in [22] studied the edge-bound content modification attack on WSNs, and proposed an effective detection mechanism against such attacks. There have also been similar works [22], [23] which consider the quantify communication for consensus network under malicious attacks.

As explained in the aforementioned works [15]–[19], only in a sufficient network connectivity, consensus can be achieved under attacks. An important criterion for evaluating the performance of different network structure to tolerate attacks is called network robustness [24], which is used to characterize the property that encapsulates the notion of sufficient local redundancy of incoming information of each node is needed. Loosely speaking, higher network robustness means better performance to tolerate attacks. Thus, good network robustness is essential for secure consensus-based controller design. As for the problem of improving network robustness, a conventional way is achieved by adding further links between nodes, i.e., by increasing redundancy. However, it may be prohibitively expensive or impossible in practice. More recently, a novel idea for increasing robustness without adding extra links is proposed in [25], which the basic strategy is to make a small subset of nodes trusted, that is, insusceptible to failures. With the help of trusted nodes, our earlier work [26] discussed the consensus problem for the first-order and second-order heterogeneous system. And Mitra *et al.* [27] also used the trusted nodes to address the issue of distributed state estimation of a linear dynamical process in an attack-prone environment.

On the other hand, consensus in asynchronous networks is a more realistic case facing practical problems, since node independently updates its state at times determined by its own clock in real environments [5]–[7]. The resilient consensus problem in asynchronous networks is studied in [18], [28], [29]. However, in these mentioned works, they all assumed that the topology of the underlying graph is fixed during the whole consensus process. But, in practice, it is hard to avoid constraints on communication capabilities, due to limited communication ranges, bandwidth, and physical obstruction of the communication channels. Therefore, system with time-varying topologies is of significance from both theoretic and engineering points of view. For these reasons, in this paper, we focus on the resilient consensus where agents with communication graphs varying as a function of time and update their states in an asynchronous mode.

Motivated by the aforementioned works, we in this paper consider the consensus-based time synchronization for WSNs in the presence of false data injection attacks. Different from most prior works which are mainly devoted to synchronous and fixed networks under attacks. We propose a novel security mechanism into the consensus computation process combined with trusted links to solve the time synchronization problem in the asynchronous and time-varying networks. The major contributions of this work are summarized as follows:

1) Inspired by [25], a concept of r-robustness with trusted links is introduced to measure the network resilience, and we found that the robustness of a network to tolerate malicious attacks can be effectively improved by setting a small subset of links as the trusted links.
2) Compared with [8], [30], [31], the asynchronous and time-varying topologies are taken into consideration in this work, and a novel Trust and Time-interval Weighted based Mean-Subsequence-Reduced (TTW-MSR) algorithm is proposed. We then give the corresponding stability and convergence analysis on the discrete-time case under a directed switching graph.
3) We propose a time synchronization algorithm base on TTW-MSR for WSNs, and evaluate its effectiveness on WSNs by extensive simulations.

The remainder of this paper is organized as follows: Section II gives some knowledge of graph theory and attack models, and provides the problem formulation. In Section III, necessary and sufficient topology conditions are analyzed and the trust-based consensus algorithm is proposed for the asynchronous and time-varying network. In Section IV, we analyze the problem of time synchronization and extend the algorithm proposed in Section III to realize time synchronization. Simulations are provided in Section V followed by some concluding remarks in Section VI.

II. PRELIMINARIES AND PROBLEM FORMULATION

In this section, we introduce several notions and a preliminary result for directed graphs and attack models, and formulate the problems.

A. PRELIMINARIES FROM GRAPH THEORY

A weighted directed graph of time-varying network with n nodes ($n > 1$) is defined as a triple $G[t] = (\mathcal{V}, \mathcal{E}[t], \mathcal{A}[t])$, where $\mathcal{V} = \{1, \ldots, n\}$ is the node set, $\mathcal{E}[t] \subseteq \mathcal{V} \times \mathcal{V}$ is the directed link set and $\mathcal{A}[t] \in \mathbb{R}^{(n\times n)}$ is the adjacency matrix at time t, respectively. The directed link (i, j) is called the incoming link of i, which means this link can transmit messages from node j to node i. And similarly, the directed link (j, i) is called outcoming link of i. For node i, the set of its incoming neighbors at time step t is denoted by $\mathcal{N}_i[t] = \{j : (i, j) \in \mathcal{E}[t]\}$, the number of neighbors is notated by $|\mathcal{N}_i[t]|$, and the set of its incoming links is denoted by $\mathcal{E}_i[t] = \{(i, j) : (i, j) \in \mathcal{E}[t]\}$. The element $w_{ij}[t]$ in $\mathcal{A}[t]$ is defined by $w_{ij}[t] \in [\mu, 1)$ if $(i, j) \in \mathcal{E}[t]$ and $w_{ij}[t] = 0$ otherwise, where $\mu > 0$. Here, self-loop is not considered, i.e., $(i, i) \notin \mathcal{E}[t]$, $\forall i \in \mathcal{V}$. We will use the terms node and agent interchangeably throughout the paper, and the graphs we mentioned in this paper are all directed.

Next, we introduce several concepts of r-robustness. Further details and examples can be found in [19] and [24].

Definition 1: (*r-reachable set*): A nonempty set $\mathcal{S} \subseteq \mathcal{V}$ is said to be r-reachable if there exists at least one node $i \in \mathcal{S}$

such that $|\mathcal{E}_i| \geq r, r \in Z_+$, where $\mathcal{E}_i = \{(i,j) \in \mathcal{E} : j \in \mathcal{V}\backslash\mathcal{S}\}$ denotes the set of i' incoming links from the outside of subset $\mathcal{S}$.

Definition 2: (*r-robustness*): A directed graph is said to be r-robust if for every pair of nonempty, disjoint subsets of $\mathcal{V}$, at least one of the subsets is r-reachable.

B. ATTACK MODEL

Recently, false data injection attacks have been widely studied for the networked control system [32]–[34]. In this paper, we focus on the case of such a malicious attack. We assume the attacker who has the ability to get control of one or more of the incoming links at node i, and can arbitrarily injects false information into communication channels.

Consider the fact the ability of an attacker may be limited by some factors, such as computational power and energy consumption, it may not be able to corrupt all links. Therefore, it is reasonable to consider the resilience of the network to specific scope of the attacks. A common assumption model in the area of resilient consensus problem is so called F-local model. In this typical model, the scope of the attacks is usually assumed to be bounded by a constant F in the neighborhood of each node. Application this assumption to our link false data injection attacks, we have the following definition.

Definition 3: (*F-local attack model*) A network is called under F-local attack if there are at most F incoming links of each node are attacked at every time step t.

C. PROBLEM FORMULATION

Consider a multi-agent system consisting of n agents, the communication topology is a weighted directed graph $G = \{\mathcal{V}, \mathcal{E}, \mathcal{A}\}$. Each agent has the following dynamics

$$x_i[t+1] = x_i[t] + u_i[t], \quad t \in \mathbb{N}, \tag{1}$$

where $x_i \in \mathbb{R}$ represents the state of agent i, and u_i is the control input to be designed.

The resilient consensus problem has been studied in the control community for the last few years [19], [35]. In this problem, there are two major concerns for the state value of each node in the system: safety condition and agreement condition. We denote by $m[0]$ and $M[0]$ are minimum and maximum initial state values of nodes respectively, and the resilient consensus can be defined as follows.

Definition 4: (*resilient consensus*) [19] The state variable $x_i[t], i \in \mathcal{V}$ of system (1) is said to achieve resilient consensus if the following two conditions are verified.

- Safety condition: there exists a set $S \subseteq [m[0], M[0]]$ that for all nodes $i \in \mathcal{V}$, it holds that $x_i[t] \in S$ for $t \in \mathbb{Z}_+$.
- Agreement condition: $\exists x^* \in \mathbb{R}$ such that $lim_{t\to\infty} x_i[t] = x^*$ for all $i \in \mathcal{V}$.

In the present work, we also consider resilient consensus of systems (1) over asynchronous network. That is to say, not all the nodes in the network updating their states simultaneously. Let $\mathcal{U}[t]$ denote the set of nodes updating their state values at time step t, then it is easy to see that $\mathcal{U}[t] = \mathcal{V}$ for a synchronous network. Note that if there exists node in the network for a long time without updates, the consensus can never be reached. In order to avoid this, another assumption is given as follows.

Assumption 1: Consider an asynchronous network G, each node makes an update at least once during the period $\mathcal{P}$, that is,

$$\bigcup_{l=0}^{\mathcal{P}} \mathcal{U}[t+l] = \mathcal{V}, \quad \mathcal{P} \in \mathbb{Z}^+ \tag{2}$$

It is worthy to point out that Assumption 1 ensures that all the nodes become active and update once during the period $\mathcal{P}$. In this paper, we consider the data received within the past time interval $\mathcal{P}$ from the current update time step t, i.e., $[t-\mathcal{P}, t]$.

Then for a time-varying network, the communication links of the digraph will change over time, and we use $G[t] = (\mathcal{V}, \mathcal{E}[t], \mathcal{A}[t])$ to denote the graph at time step t.

The following assumption is made in this paper.

Assumption 2: The communication topology of each digraph should stay at least $\mathcal{P}$ time steps before it changes.

The meaning of Assumption 2 is that for each period $\mathcal{P}$, it ensures each node receives at least one data from its each neighbor before the current graph switches.

Furthermore, we define the union graph in the interval $[t-K, t]$.

Definition 5: (*K-union graph*) For a fixed time period K, $K \geq \mathcal{P}$, $G^K[t]$ denotes the union of graphs within the time interval $[t-K, t]$. That is, $G^K[t] = \cup_{l=0}^{K} G[t-l]$.

It should be emphasized that in a time-varying network, the property defining an F-local set must hold for all points in time. Then we modify Definition 3 by bringing in time-varying period K.

Definition 6: ((K,F)-*local attack model*) For a fixed period K, we call an attack model as (K,F)-local attack if there are at most F incoming links of each node in $G^K[t]$ are compromised.

As mentioned above, for a time-varying and asynchronous network, a consensus control protocol uses only certain sampling data for the updates of controller. Then, we mainly focus on the following two problems for system (1) under protocol $u_i[t]$: 1) under what topology conditions the consensus can be achieved for asynchronous and time-varying network; and 2) how to design $u_i[t]$ to achieve consensus even if some communication links are compromised by attackers.

III. GRAPHICAL CONDITION AND ALGORITHM DESIGN

In this section, we will first explore a necessary and sufficient graphical condition for resilient consensus protocol, and then give the detail of the algorithm design.

A. GRAPHICAL CONDITION FOR RESILIENT CONSENSUS

As explained in work [19], only in a sufficient network robustness, resilient consensus can be achieved.

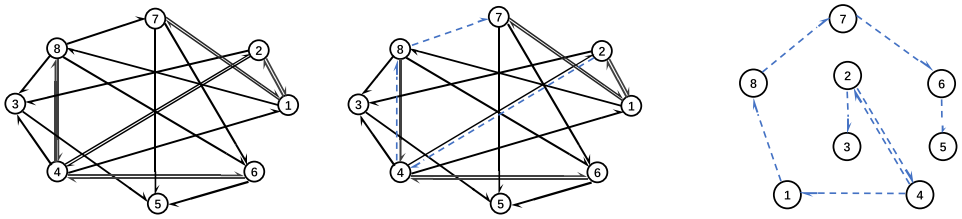


FIGURE 1. (a): A 1-robust with $\mathcal{T}_\varepsilon$ digraph with 8 nodes and 21 communication links. (b): A 3-robust with $\mathcal{T}_\varepsilon$ digraph by setting (2, 4), (4, 8), (8, 7) (blue dashed lines) as trusted links of Figure 1(a). (c): A digraph where the trusted links (blue dashed lines) is a primary link set.

Inspired by work [25], we introduce a new concept, called *trusted link*.

Definition 7: (*trusted link*) A link (i, j) in digraph $\mathcal{G}$ is called a trusted link if it does not suffer the false data injection attack from the attacker. All messages transmitted through (i, j) cannot be modified or failed.

In other words, trusted link (i, j) ensures that the data received by node i equals to the data sent by node j. We use $\mathcal{T}_\varepsilon$ to denote the set of trusted links in the G.

Considering the trusted links may exist in the network, the property of r-robustness mentioned in [19] can no longer be used to judge the network robustness. Thus, we propose a new robustness metric based on the concept of trusted link, termed r-reachable with $\mathcal{T}_\varepsilon$, which is defined as follows.

Definition 8: (*r-reachable with* $\mathcal{T}_\varepsilon$) Given a digraph G and a nonempty subset $S \subseteq \mathcal{V}$, let $\mathcal{T}_\varepsilon$ be a set of trusted links. For node $i \in S$, $\mathcal{E}_i^*$ is a subset of $\mathcal{E}_i$, which satisfies $\mathcal{E}_i^* = \{(i, j) : (i, j) \in \mathcal{E}_i, j \notin S, i \in S\}$. We say S is a r-reachable with $\mathcal{T}_\varepsilon$ set if $\exists i \in S$ such that $|\mathcal{E}_i^*| \geq r$ or $\mathcal{E}_i^* \cap \mathcal{T}_\varepsilon \neq \varnothing$, $r \in Z_+$.

The concept of r-reachable with $\mathcal{T}_\varepsilon$ means there must exist a node in this set which has r incoming links or at least one trusted incoming link from the outside of its own set.

Remark 1: If there exists a node in the set which has at least one trusted incoming link from the outside of its own set, we call this set as an infinity-reachable with $\mathcal{T}_\varepsilon$ set.

Definition 9: (*r-robustness with* $\mathcal{T}_\varepsilon$) We call a digraph G is r-robust with $\mathcal{T}_\varepsilon$ if for any pair of non-empty, disjoint subsets $S_1, S_2 \subseteq \mathcal{V}$, at least one of the subsets is r-reachable with $\mathcal{T}_\varepsilon$.

Next, we give an example to better illustrate the concept of r-robustness with $\mathcal{T}_\varepsilon$. Figure 1(a) is a digraph of 1-robustness with $\mathcal{T}_\varepsilon$, because when we take $S1 = \{1, 2, 7\}$, $S2 = \{4, 6, 8\}$, neither $S1$ nor $S2$ is 2-reachable with $\mathcal{T}_\varepsilon$. But now we can improve the robustness by selecting a set of trusted links. Considering that the selected set should be as small as possible, we select one of the smallest sets that can make the Figure 1(a) reach 3-robustness with $\mathcal{T}_\varepsilon$. As shown in Figure 1(b), we select {(2,4), (4,8), (8,7)} as trusted links. In contrast to the way of selecting trusted links, in this network, the conventional method requires at least six additional links to achieve 3-robustness. Such as adding {(1,4), (2,7), (2,8), (6,1), (6,2), (7,4)}. Figure 1(b) can guarantee resilient consensus even when there is a compromised incoming link around each node, whereas Figure 1(a) cannot. From this example, we can see that the way to set the trusted links has a significant impact on the robustness of the network.

By setting trusted links, the target robustness can be achieved even in the network whose connections among node are not dense. Since robustness can be improved by setting a set of trusted links, how should we find these trusted links? Here we consider the following method.

To better characterize this method, we first define a notion called primary link set.

Definition 10: (*primary link set*) Given a digraph G with n nodes, a link set called primary link set if the following conditions are met.

- All links in the set are trusted links, and the number of links is n which equals to the number of nodes in G.
- Each node has one incoming link in G.
- These links with nodes can form a spanning tree in G.

Figure 1(c) is an example where the links in the graph is a primary link set.

Theorem 1: When there is a primary link set in a digraph G, the robustness of G reaches infinity. That is, the digraph G can reach infinity-robustness with $\mathcal{T}_\varepsilon$.

proof: The proof of Theorem 1 is shown in Appendix A.

Note that if a digraph satisfies infinity-robust with $\mathcal{T}_\varepsilon$, then no matter how many compromised incoming links exist around each node, the consensus can still be achieved. Because when removing all the suspicious data, there still exists data from the trusted link of each node used for updating.

In the following, we present the method. First, we selected a primary link set from the digraph G. Let $\mathcal{T}_\varepsilon$ denote the set of trusted links in G. Then, we set $\mathcal{T}_\varepsilon$ equal to this primary link set. If the target robustness can still achieve after removing the link from set $\mathcal{T}_\varepsilon$, then remove this link, otherwise keep it in the set. If no link can be removed, the remaining links in $\mathcal{T}_\varepsilon$ are the trusted links we are looking for. The disadvantage of this method is that the optimal set cannot be found. But luckily, the target set of trusted links can always be found to reach expected robustness.

We take Figure 1(a) for example. Here we want to find a set of links to reach 3-robustness with $\mathcal{T}_\varepsilon$. First, we select $\mathcal{T}_\varepsilon = \{(7, 1), (4, 2), (4, 3), (8, 4), (7, 5), (8, 6), (8, 7)\}$ which satisfies Definition 10. Then we try to remove (7,1), the topology can still reach 3-robustness with $\mathcal{T}_\varepsilon$. So (7,1) is removed

from $\mathcal{T}_\varepsilon$. We remove other links in turn and check if the target robustness can maintain. Finally, we find after removing (7,1), (4,3), (7,5), (8,6) from $\mathcal{T}_\varepsilon$, 3-robustness with $\mathcal{T}_\varepsilon$ property of this topology can still maintain. Meanwhile removing any remaining links will result in less than 3-robustness with $\mathcal{T}_\varepsilon$. Therefore, the set $\mathcal{T}_\varepsilon = \{(4,2),(8,4),(8,7)\}$ is the set which we are looking for.

As for how to optimize the method to find the set of trusted links can be our future work.

In the next section, we will use this robustness concept to deal with the resilient consensus problem under false data injection attacks in the asynchronous time-varying network.

Considering a time-varying network, the constraint that the graph may not be $(2F+1)$-robust with $\mathcal{T}_\varepsilon$ at each time step. To solve this problem, a strategy that allows the network to achieve resilient consensus is introduced as follows.

We give a solution by granting that the collection of joint communication graphs over a bounded time interval is $(2F+1)$-robust with $\mathcal{T}_\varepsilon$, detailed below.

Definition 11: ((K, r)-*robustness with* $\mathcal{T}_\varepsilon$) For a fixed time period K, we call $G[t]$ is a (K, r)-robust with $\mathcal{T}_\varepsilon$ graph if the union graph $G^K[t]$ is r-robust with $\mathcal{T}_\varepsilon$.

Though $G[t]$ may not be ensured an r-robust with $\mathcal{T}_\varepsilon$ graph at each time step t, Definition 11 provides a new perspective that we only need require the union communication graph over a bounded time interval K be a $(2F+1)$-robust with $\mathcal{T}_\varepsilon$ graph. Thus, the graph is not required to be $(2F+1)$-robust with $\mathcal{T}_\varepsilon$ at every time step.

B. RESILIENT ALGORITHM

In this section, we present a resilient consensus algorithm with trusted links.

We propose a modification of the original MSR algorithm for asynchronous time-varying networks, which we term as Trust and Time-interval Weighted based MSR algorithm (TTW-MSR).

The algorithm considers all data received within the past K time steps from the current time step t. And the messages received from the trusted links will be used for updating. Each node $i \in \mathcal{V}$, updates its state as follows

$$x_i[t+1] = x_i[t] + \sum_{j \in \psi_i[t]} w_{ij}[t - l_{ij}[t]](x_j[t - l_{ij}[t]] - x_i[t]), \tag{3}$$

where $(t - l_{ij}[t])$ denotes the timestamp of latest message received from node j to node i, $l_{ij}[t] \in [0, K]$. $w_{ij}[t - l_{ij}[t]]$ is the (i,j)th entry of the adjacency matrix $\mathcal{A}[t - l_{ij}[t]]$. $\psi_i[t]$ is the set of nodes whose most recent data send to node i will be used for updating of node i. The detail of algorithm shown in Algorithm 1.

In Algorithm 1, the input values, including i, Γ_i, F, K, $\mathcal{T}_\varepsilon$, t, mean that node i has a list Γ_i which stores the received time-stamped messages from its neighbors in the past time period K from current time step t. Specifically, Γ_{ij} denotes the received message from node j. The message is a triple $(j, t_s, x_j[t_s])$, where j is the sending node and t_s is sending time of node j. It is noted that the data in Γ_{ij} is stored only within the recent time interval K and will be discarded when data expires. $\mathcal{T}_\varepsilon$ is the set of trusted links and F denotes the system is under (K, F)-local attack model. First, node i sends its state value to its current out-neighbors. Then it uses $\mathcal{N}_i^K[t]$ to denote its neighbors in time interval $[t-K, t]$. On lines 3-5, $(t - l_{ij}[t])$ is the timestamp of the most recent received message for each neighbor j in the time interval. And then these most recent received messages will be stored in $\chi[t]$. On line 6, a function named *Filter* is performed. In this function, node i first sorts $\chi[t]$ according to state value. Second, compared to its own value $x_i[t]$, if there are less than F values strictly smaller than $x_i[t]$, then node i removes these data. Otherwise, it removes the smallest F values in the sorted list. Similarly, at most F largest data that larger than $x_i[t]$ are removed. The remaining data in the sorted list is denoted as $\mathcal{R}_i[t]$. For that the data transmitted through the trusted link is trustful, we also use these received data for updating. Hence, as shown on line 7, $\psi_i[t]$ is the final neighbor set for the update of node i. Finally, on line 8, node i computes the new state value by rule (3).

Algorithm 1 TTW-MSR

Input: $i, \Gamma_i, F, \mathcal{T}_\varepsilon, K, t$
Output: $x_i[t+1]$

1: send message $(i, t, x_i[t])$ to its out-neighbors
2: $\mathcal{N}_i^K[t] \leftarrow \cup_{l=t-K}^{t} \mathcal{N}_i[l]$
3: $l_{ij}[t] \leftarrow \min\{l \in [0,K] | \Gamma_i[t-l] \neq \emptyset\}, \forall j \in \mathcal{N}_i^K[t]$
4: $\chi[t] \leftarrow \emptyset$
5: $\chi[t] \leftarrow \chi[t] \cup \Gamma_{ij}[t - l_{ij}], \forall j \in \mathcal{N}_i^K[t]$
6: $\mathcal{R}_i[t] = Filter(x_i[t], \chi[t], \mathcal{T}_\varepsilon, F)$
7: $\psi_i[t] = \mathcal{R}_i[t] \cup \{j | (i,j) \in \mathcal{T}_\varepsilon, j \in \mathcal{N}_i^K[t]\}$
8: $x_i[t+1] = x_i[t] + \sum_{j \in \psi_i[t]} w_{ij}[t - l_{ij}[t]](x_j[t - l_{ij}[t]] - x_i[t])$
9: **return** $x_i[t+1]$

Based on the aforementioned discussions, now we are in the position to provide the main result.

Theorem 2: Consider the system (1) with asynchronous update protocol (3) under Assumptions 1 and 2 with a (K, F)-local set of false data injection attacks. The resilient consensus can be reached if the underlying graph satisfies $(K, 2F+1)$-robustness with $\mathcal{T}_\varepsilon$ for each time step t, $t \geq K$.

proof: The proof of Theorem 2 is shown in Appendix B.

Remark 2: It should be note that in Section III-A, we just set $K = t$ when $t < K$. Because in this initial time, the resilient consensus cannot be guaranteed, for the $G^K[t]$ is not a (K, r)-robust with $\mathcal{T}_\varepsilon$ graph, where $t \in [0, K)$. However, the estimates are not affected by malicious outlier value since the F largest value and F smallest values are removed. Therefore, the update process during the initial K time steps will not affect the result of Theorem 2.

IV. APPLICATION TO TIME SYNCHRONIZATION

In this section, we extend TTW-MSR algorithm to time synchronization.

Assuming all nodes in a time-varying network modeled as $G[t]$ exchange data with neighbors every T time. Consider the clock frequencies of nodes are inconsistent, and the actual update period of each node i is T_i, $i = 1, 2, \ldots, n$.

Moreover, the clock frequency of each node is only a slight deviation [6], which means there exists a period $\mathcal{P}$ ensuring all nodes will do update and communication at least once in this time interval. Therefore, we can apply the analysis of resilient consensus in asynchronous time-varying networks discussed in Section III to time synchronization.

A. CLOCK MODEL

By refereing to [9]–[12], the clock model of each node i can be approximated as a linear model, which is denoted as

$$\mathcal{C}_i[t] = \alpha_i t + \beta_i, \quad i \in \mathcal{V}, \tag{4}$$

where t is absolute time that no sensor node knows the true value of t, $\mathcal{C}_i$ is physical clock, α_i is physical drift which represents the rate of the clock change, and β_i is physical offset. Both α_i and β_i cannot be computed in practice. Slight differences in α_i and β_i of sensors will cause their physical clock updating with a different speed. Considering that other components of the sensor may depend on a continuous running physical clock, the value of the physical clock cannot be changed [10]. That is, the values of α_i and β_i cannot be directly corrected. Therefore, the concept of logical clock [9]–[12] is generally introduced as follows

$$\mathcal{C}_i^*[t] = \alpha_i^*[t]\mathcal{C}_i[t] + \beta_i^*[t], \quad \forall i, j \in \mathcal{V}, \tag{5}$$

where α_i^* is called logical drift, and β_i^* is the logical offset used to correct the values of α_i and β_i of the physical clock model respectively. Initially, $\alpha_i^*[0] = 1$ and $\beta_i^*[0] = 0$. Thus the final clock model is

$$\mathcal{C}_i^*[t] = \alpha_i^*[t]\alpha_i t + \alpha_i^*[t]\beta_i + \beta_i^*[t], \quad \forall i, j \in \mathcal{V}. \tag{6}$$

The goal of time synchronization is to find (α_i^*, β_i^*) for each node which satisfies

$$\lim_{t \to +\infty} \mathcal{C}_i^*[t] - \mathcal{C}_j^*[t] = 0, \quad \forall i, j \in \mathcal{V}. \tag{7}$$

More specifically, another equivalent representation is

$$\begin{cases} \lim\limits_{t \to +\infty} \alpha_i^*[t]\alpha_i = \hat{\alpha} & \forall i \in \mathcal{V}, \\ \lim\limits_{t \to +\infty} \alpha_i^*[t]\beta_i + \beta_i^*[t] = \hat{\beta} & \forall i \in \mathcal{V}, \end{cases} \tag{8}$$

where $\hat{\alpha} \in [\min_{i \in \mathcal{V}}\{\alpha_i\}, \max_{i \in \mathcal{V}}\{\alpha_i\}]$ and $\hat{\beta} \in \mathbb{R}$. For simplicity, we call $\alpha_i^*\alpha_i$ as adjusted drift and $(\alpha_i^*\beta_i + \beta_i^*)$ as adjusted offset.

Note that the actual values of α_i and β_i may change slowly over time, but the time to complete time synchronization is much less than α_i or β_i changes. Therefore, we can solve this problem by restarting the synchronous update algorithm after an appropriate time. And there is an implicit assumption that the process of message exchange is instant and the transmission delay can be ignored. This assumption has been used in most of the time synchronization protocols [9], [13].

B. UPDATE OF CLOCK

To meet the final goal of time synchronization as shown in (7) and (8), nodes need to periodically communicate with neighbors. There are two issues that deserve attention. One is that the nodes update asynchronously. And the other is how to update α_i^* and β_i^*.

For the communication mode, since the update frequencies of the nodes are inconsistent, when the system sets a common update period T, the actual update period of each node is $T_i = T/\alpha_i, i = 1, 2, \ldots, n$. It leads to the asynchronous update mode of the nodes. In other words, the node cannot receive data of all neighbors at the same time. It is worth noting that there exists such a $\mathcal{P} = T/\min_{i \in \mathcal{V}}\{\alpha_i\}$, all nodes will do update at least once during this period. For the existence of such a $\mathcal{P}$ that satisfies Assumption 1, the analysis in Section III is applicable to time synchronization.

For the second issue, when actual update period T_i arrives, node i first broadcasts its own data to current neighbors and then updates the state value immediately. The message denoted as M_i sent by i contains $(i, \mathcal{C}_i[t], \mathcal{C}_i^*[t], \alpha_i^*[t])$, where i is the id of the sending node, $\mathcal{C}_i[t]$ and $\mathcal{C}_i^*[t]$ are the physical clock and logical clock at time t respectively, and $\alpha_i^*[t]$ is the logical drift of node i at time t. When receiving the message M_j from its neighbor j at time t, node i will record this data along with its own $C_i[t]$ in Ω_{ij}. If the old data of neighbor j exists in Ω_{ij}, then old data will be replaced. Node $i \in \mathcal{V}$ updates its logical drift $\alpha_i^*[t]$ at time t according to following rule

$$\alpha_i^*[t^+] = \alpha_i^*[t] + \sum_{j \in \psi_i[t]} w_{ij}[t_1](\alpha_{ij}[t]\alpha_j^*[t_1] - \alpha_i^*[t]), \tag{9}$$

where $t^+ \in (t, t + T_i]$ denotes the timestamp before the next update of node i, $w_{ij}[t_1]$ is the (i, j)th entry of the adjacency matrix $\mathcal{A}[t_1]$, and $\psi_i[t]$ is the set of neighbor nodes which not be discarded according to the filtering rule. The parameter $\alpha_{ij}[t]$ is a relative physical drift ratio of $\frac{\alpha_j}{\alpha_i}$, which calculated from $\alpha_{ij}[t] = \frac{\mathcal{C}_j[t_1] - \mathcal{C}_j[t_2]}{\mathcal{C}_i[t_1] - \mathcal{C}_i[t_2]}$, t_1 and t_2 is absolute time instant of two messages when node i received the data of node j with $t_1 > t_2$.

And similarly the logical offset $\beta_i^*[t]$ at time t updated according to following rule

$$\beta_i^*[t^+] = \beta_i^*[t] + \sum_{j \in \psi_i[t]} w_{ij}[t_1](\mathcal{C}_j^*[t_1] - (\alpha_i^*[t]\mathcal{C}_i[t_1] + \beta_i^*[t])). \tag{10}$$

As can be seen from the update process (9), in order to calculate the value of the relative physical drift ratio α_{ij}, one need two messages from the same neighbor. Thus, each node will have two sets to store received messages. One denotes as Ω_i, which is used to store the most recently received data of neighbors. Specifically, Ω_{ij} denotes the data in Ω_i received from j. These data may be modified and needed to be filtered. After each update process, the data in Ω_i, which used to update α_i^* and β_i^* will be stored in the set Υ_i. Similarly, Υ_{ij} denotes the data received from node j. Thus, Υ_i stores the

Algorithm 2 TLTS

Input: $\mathcal{T}_\varepsilon, T, T^*, F, t = 0$

1: initialize $\alpha_i^*[0] = 1, \beta_i^*[0] = 0, \Omega_i = \emptyset, \Upsilon_i = \emptyset, \forall i \in \mathcal{V}$
2: **for** $i \in \mathcal{V}$ **do**
3: $\quad T_i \leftarrow T/\alpha_i$
4: **while** $t < T^*$ **do**
5: $\quad$ **if** $\mathcal{C}_i[t]/T_i \in \mathbb{N}^+$ **then**
6: $\quad\quad$ broadcast state message $\mathcal{M}_i[t]$ to its neighbors
7: $\quad\quad$ call *timeUpdate*(i)
8: $\quad$ **if** node i receives the message from node j at time t **then**
9: $\quad\quad \Omega_{ij} \leftarrow \mathcal{M}_j[t] \cup \mathcal{C}_i[t]$

latest data from Ω_i which participated in the update of node i. It is obvious that the data in the Ω_i is newer than Υ_i. At the same time, we noticed that since the historical data is required at least twice, the first data received from each neighbor is directly stored in the Υ_i. The update process of the node will not start until two messages of each neighbor are received. Therefore, this update rule cannot detect whether the first data sent by each neighbor node has been modified. This is also the problem existing in many time synchronization algorithms.

C. ALGORITHM FOR TIME SYNCHRONIZATION

The algorithm named Trusted Links based Time Synchronization (TLTS) algorithm to achieve time synchronization under (K, F)-local model is shown as follows, where Algorithm 2 gives the main framework of TLTS and Algorithm 3 shows the detail of state updating.

Algorithm 2 introduces the framework of TLTS. In this algorithm, $\mathcal{T}_\varepsilon$ is the set of trusted links of the digraph G, T is the common update period of each node, T^* is the total running duration of this algorithm which is set enough to achieve time synchronization, and F is the scope of attacks of the (K, F)-local attack model. t indicates the current absolute time. On lines 2-3, the real update period of each node will be computed. On lines 5-7, the node i sends data to neighbors and performs the update only when its actual update period T_i arrives. And the received message from node j along with its own physical clock will be stored in the Ω_{ij}.

Algorithm 3 is the detail of Algorithm 2 describing how to update the state value. The process is divided into three steps. The first step is to compute the relative logical drift $\hat{\alpha}_{ij}^*[t]$ with reference to each neighbor j, the second step is to detect the suspicious data, and then update of α_i^* and β_i^*, $i = 1, 2, \ldots, n$, will be performed in the last step. We use t_1, t_2 to represent the absolute receiving time instant of data storing in the Ω_{ij} and Υ_{ij} respectively. On lines 4-5, $\hat{\alpha}_{ij}^*[t]$ denotes the new logical drift of node i with reference to data received from link (i, j). Lines 6-13 detect the reliability of the data in the Ω_i. The method of removing the extremums can ensure the reliability of $\hat{\alpha}_{ij}^*[t]$. Since the data transmitted in the trusted links are always trustful, we will not remove these data. On line 14, ψ_i contains the remaining nodes in

Algorithm 3 TimeUpdate(i)

Input: i
Output: $\alpha_i^*[t^+], \beta_i^*[t^+]$

1: initialize $\xi = \emptyset, \mathcal{V}_{max} = \emptyset, \mathcal{V}_{min} = \emptyset$.
2: **if** Ω_{ij} not empty and Υ_{ij} not empty, $\forall j \in \mathcal{N}_i$ **then**
3: $\quad \xi \leftarrow \xi \cup j$
4: **for** $j \in \xi$ **do**
5: $\quad \hat{\alpha}_{ij}^*[t] \leftarrow \frac{\alpha_j^*[t_1] \times (\mathcal{C}_j[t_1] - \mathcal{C}_j[t_2])}{\mathcal{C}_i[t_1] - \mathcal{C}_i[t_2]}$
6: $\quad$ **if** $\hat{\alpha}_{ij}^*[t] > \alpha_i^*[t]$ **then**
7: $\quad\quad \mathcal{V}_{max} \leftarrow \mathcal{V}_{max} \cup j$
8: $\quad$ **if** $\hat{\alpha}_{ij}^*[t] < \alpha_i^*[t]$ **then**
9: $\quad\quad \mathcal{V}_{min} \leftarrow \mathcal{V}_{min} \cup j$
10: **if** $|\mathcal{V}_{max}| > F$ $(|\mathcal{V}_{min}| > F)$ **then**
11: $\quad F$ nodes counting from the largest (smallest) $\hat{\alpha}_{ij}^*[t]$ in $\mathcal{V}_{max}$ $(\mathcal{V}_{min})$ are removed in ξ
12: **else**
13: $\quad$ all nodes in $\mathcal{V}_{max}$ $(\mathcal{V}_{min})$ are removed in ξ
14: $\psi_i[t] \leftarrow \xi \cup \{j | (i, j) \in \mathcal{T}_\varepsilon, j \in \mathcal{N}_i^K[t]\}$
15: **for** $j \in \psi_i[t]$ **do**
16: $\quad \Upsilon_{ij} \leftarrow \Omega_{ij}$
17: $\quad \Omega_{ij} \leftarrow \emptyset$
18: $\alpha_i^*[t^+] \leftarrow \alpha_i^*[t] + \sum_{j \in \psi_i[t]} w_{ij}[t_1](\hat{\alpha}_{ij}^*[t] - \alpha_i^*[t])$
19: $\beta_i^*[t^+] \leftarrow \beta_i^*[t] + \sum_{j \in \psi_i[t]} w_{ij}[t_1](\mathcal{C}_j^*[t_1] - (\alpha_i^*[t]\mathcal{C}_i^*[t_1] + \beta_i^*[t]))$
20: **return** $\alpha_i^*[t^+], \beta_i^*[t^+]$

Ω_i and neighbor nodes that transmit data on the trusted links. On lines 15-17, since the suspicious data in the ξ has been removed, the message received from remaining nodes can directly overwrite the data of the corresponding node in the Υ_i. Then, we clear the data in the Ω_i ready to receive the new data. The calculation of the next-time logical drift $\alpha_i^*(t^+)$, $t^+ \in (t, t + T_i]$. $\beta_i(t^+)$ is performed as shown on lines 18-19, where t represents the current absolute time.

The TLTS algorithm can effectively achieve the time synchronization under (K, F)-local model if the union communication digraph is $(K, 2F + 1)$-robust with $\mathcal{T}_\varepsilon$ and satisfies the Assumption 1 and 2 as discussed in Theorem 2.

V. SIMULATION EXPERIMENT

In this section, we conduct simulations to illustrate the influence of setting the trusted links on the time synchronization under the false data injection attacks. The results can confirm the efficiency of our proposed algorithm TLTS.

The asynchronous time-varying communication digraph of our experiments is shown in Figure 2. There are eight nodes and data exchanges among the nodes through the directed links. The blue dashed directed line represents the trusted link, the red dotted directed line represents the compromised link, and the black solid directed line represents the normal link. Figure 2 indicates that the graph changes periodically over time. A total of three different communication

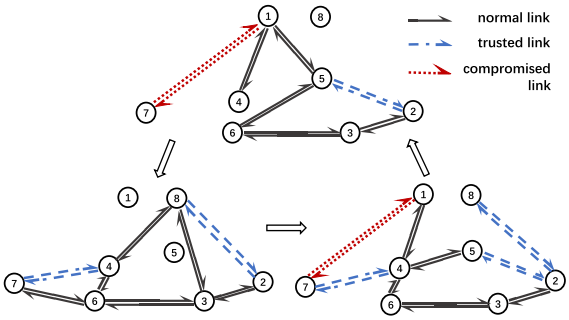


FIGURE 2. The time-varying network with three subgraphs changed periodically where red dotted directed line denotes the compromised link, blue dashed directed line denotes the trusted link, and black solid directed line denotes the normal link.

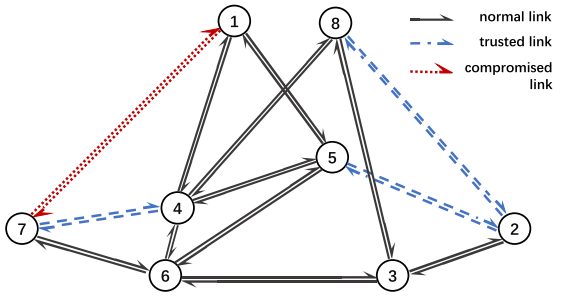


FIGURE 3. The union of time-varying network as shown in Figure 2 where red dotted directed line denotes the compromised link, blue dashed directed line denotes the trusted link, and black solid directed line denotes the normal link.

subgraphs were generated. At the same time, we verified that each subgraph does not satisfy 3-robustness with $\mathcal{T}_\varepsilon$. Therefore, there is no guarantee the achievement of clock synchronization under the false data injection attacks. The union digraph is shown in Figure 3. Similarly, we verified the robustness of this graph. We found that when we use the traditional r-robust metric to measure this graph, it is just 1-robust, but when taking the trusted links into account, the graph can reach 3-robustness with $\mathcal{T}_\varepsilon$.

The aim of our experiments is to achieve the (7) and (8). Therefore, for simplicity, we call $\alpha_i^*\alpha_i$ as adjusted drift, $(\alpha_i^*\beta_i + \beta_i^*)$ as adjusted offset and $\max_{i,j\in\mathcal{V}}\{\mathcal{C}_i^* - \mathcal{C}_j^*\}$ as the maximum error in the result figures. Some common parameters are set as follows.

The common update period of nodes is set as $T = 0.1s$. For better observation of the experiment, the sensor clock parameters α_i and β_i are randomly selected from the interval [0.5, 1.5] and [0, 100], respectively. The initial states are set as $\alpha_i^*[0] = 1$ and $\beta_i^*[0] = 0$, for $i \in \mathcal{V}$. Thus there is such a period of $\mathcal{P} = T/\min_{i\in\mathcal{V}}\{\alpha_i\} = 0.1/0.5 = 0.2\ s$, all nodes will do update and communication at least once. The scope of attack is assumed bounded by 1, i.e., $F = 1$. We set links (1,7), (7,1) as compromised links, and set links (2,8), (8,2), (2,5), (5,2), (4,7), (7,4) as trusted links. If the stay time of each subgraph in Figure 2 is set greater than $0.2s$, then there exists a K, the nodes in Figure 3 will receive data from all neighbors and do update at least once. We conducted three experiments as shown in experiment (1), (2), and (3). Once the values of α_i, β_i are determined, they will keep consistent in experiments.

1) *System without trusted links*: In this experiment, we set the stay time of each subgraph in Figure 2 as $0.25s$ to meet the Assumption 2. But we do not consider the role of the trusted links in the algorithm (TLTS), that is, the algorithm assumes the trusted links in Figure 2 and Figure 3 are the normal links, and the experimental results are shown in Figure 4(a), 4(b), 4(c), and 4(d). The result shows the adjusted drift $(\alpha_i^*\alpha_i)$, adjusted drift $(\alpha_i^*\beta_i + \beta_i^*)$ and logical time $(\mathcal{C}_i^*)$ are divided into two parts, respectively. And the maximum error $(\max_{i,j\in\mathcal{V}}\{\mathcal{C}_i^* - \mathcal{C}_j\})$ is constantly increasing. These figures show the failure of the clock synchronization.
2) *System with trusted links*: In this experiment, we also set the stay time of each subgraph in Figure 2 as $0.25s$ to meet the Assumption 2. But this time we consider the role of trusted links. Each node updates and communicates with neighbors following the TLTS rule. The experimental results are shown in Figure 4(e), 4(f), 4(g), and 4(h). Adjusted drift $(\alpha_i^*\alpha_i)$, adjusted drift $(\alpha_i^*\beta_i+\beta_i^*)$ and logical time $\mathcal{C}_i^*$ reach consensus respectively. And the maximum error $(\max_{i,j\in\mathcal{V}}\{\mathcal{C}_i^* - \mathcal{C}_j\})$ of system tends to 0. It is note that the time synchronization is achieved.
3) *When stay time of each subgraph is not enough*: In this experiment, each node update follows the TLTS rule which takes trusted links into account, but we assume that the time interval before each subgraph switching is less than $0.2s(\mathcal{P} = 0.2s)$. The experimental results are shown in Figure 4(i), 4(j), 4(k), and 4(l). The Adjusted drift $(\alpha_i^*\alpha_i)$, adjusted drift $(\alpha_i^*\beta_i + \beta_i^*)$ and logical time $(\mathcal{C}_i^*)$ are not reach consensus, respectively. And the maximum error $(\max_{i,j\in\mathcal{V}}\{\mathcal{C}_i^* - \mathcal{C}_j\})$ of system is constantly increasing which give the information of failure to achieve clock synchronization.

Comparing experiments (1) and (2), we can see the importance of improving network resilience to attacks by setting trusted links. Experiment (1) not take trusted links into consideration. Thus, the union graph is only 1-robust, which are vulnerable to (K, F)-local attack model $(F = 1)$. And when the trusted links are considered in experiment (2), the union graph can reach 3-robustness with $\mathcal{T}_\varepsilon$, which guarantee the achievement of time synchronization. Experiments (2) and (3) both consider the role of the trusted links, the difference is the stay time of each subgraph. The stay time of (3) does not satisfy the Assumption 2. Therefore, some nodes may not do the update and communicate with neighbors before switching of the subgraph. Thus result in the lack of enough data for updating. For these reasons, the failure of clock synchronization is caused. These experiments illustrate the efficiency of our proposed algorithm TLTS.

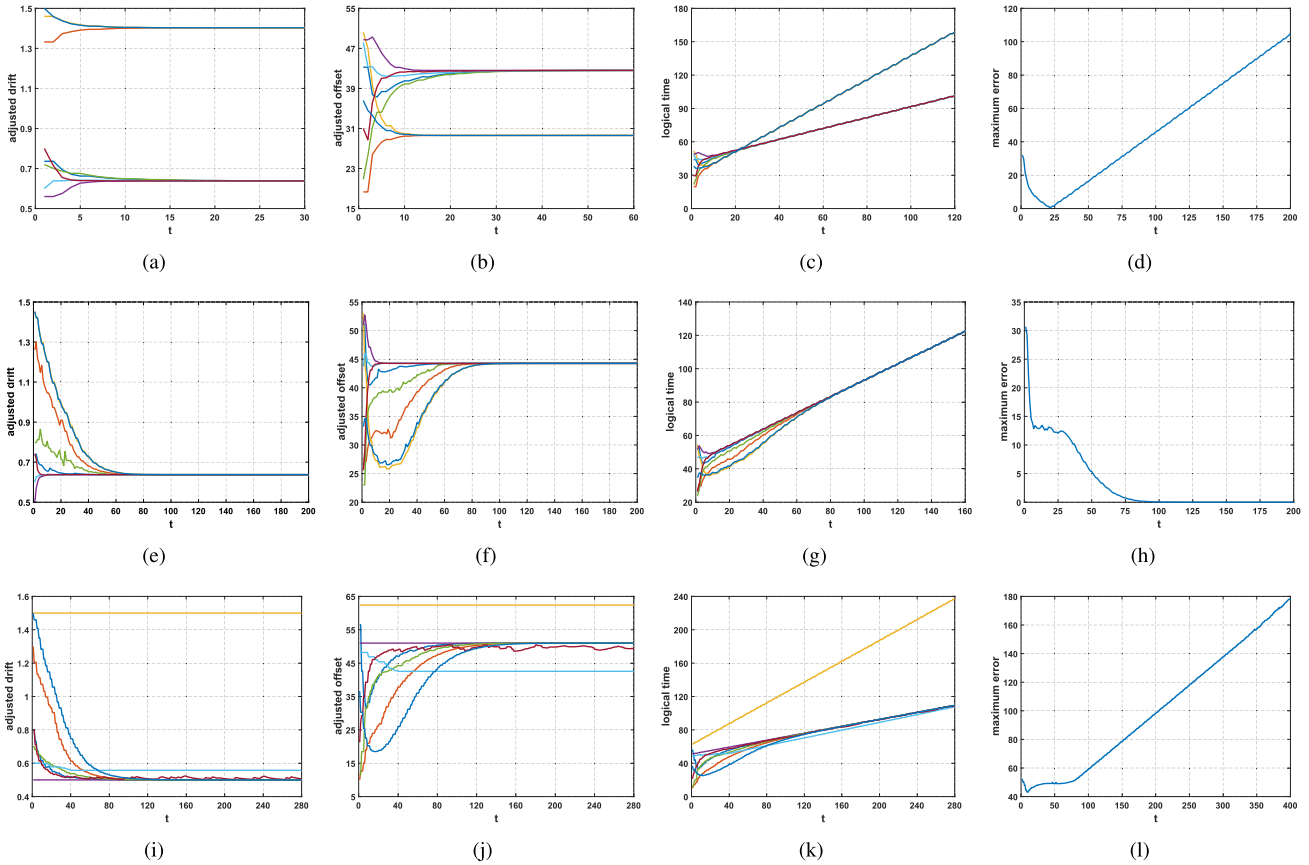


FIGURE 4. (a)(b)(c)(d): Experiment results of adjusted drift ($\alpha_i^*\alpha_i$), adjusted offset ($\alpha_i^*\beta_i + \beta_i^*$), logical time ($C_i^*$), and the maximum error ($\max_{i,j\in\mathcal{V}}\{C_i^* - C_j^*\}$) of Figure 2, where the stay time of each subgraph satisfying the Algorithm 2 but the algorithm does not consider the role of the trusted links. (e)(f)(g)(h): Experiment results of adjusted drift ($\alpha_i^*\alpha_i$), adjusted offset ($\alpha_i^*\beta_i + \beta_i^*$), logical time ($C_i^*$), and the maximum error ($\max_{i,j\in\mathcal{V}}\{C_i^* - C_j^*\}$) of Figure 2, where the stay time of each subgraphs satisfying the Algorithm 2 and each node uses the TLTS algorithm to update state which considers the role of the trusted links. (i)(j)(k)(l): Experiment results of adjusted drift ($\alpha_i^*\alpha_i$), adjusted offset ($\alpha_i^*\beta_i + \beta_i^*$), logical time ($C_i^*$), and the maximum error ($\max_{i,j\in\mathcal{V}}\{C_i^* - C_j^*\}$) of Figure 2, where each node uses the TLTS algorithm to update state which considers the role of the trusted links but the stay time of each subgraph not satisfying the Algorithm 2.

VI. CONCLUSION

In this paper, we have presented a resilient consensus-based time synchronization algorithm—TTW-MSR protocol for WSNs under false data injection attacks. First, a novel approach to improve the resilience of network is proposed to achieve the desired structural robustness. We proved that for the false data injection into the communication links, the algorithm can still achieve convergence under the specific network robustness. The proposed algorithm is fully distributed, asynchronous, and valid to time-varying network topologies. Then we extend this algorithm to time synchronization for WSNs. Extensive simulations demonstrate the effectiveness of the proposed algorithm.

APPENDIX A
PROOF OF THEOREM 1

We prove this by contradiction. Let G be a digraph with vertex set $\mathcal{V}$ and P_G be a primary set of G. We assume that the robustness of G cannot reach infinity. That is, according to Definition 9, $\exists S_1, S_2 \in \mathcal{V}$, neither S_1 nor S_2 is infinity-reachable with $\mathcal{T}_\varepsilon$. It means all nodes in S_1 and S_2 do not has a trusted incoming link from outside of its own set according to Definition 8 and Remark 1. However, each node has a trusted incoming link according to Definition 10 which results that all trusted links cannot form a spanning tree. It contradicts to the Definition 10 where when a primary link set existing in a digraph the trusted links with all nodes can form a spanning tree.

APPENDIX B
PROOF OF THEOREM 2

First, we prove the safety condition in Definition 4. Let $M[K,t]$ and $m[K,t]$ to be the maximum and minimum state values of the nodes in the interval $[t-K,t]$, respectively.

$$m[K,t] = \min_{i\in\mathcal{V},l\in[0,K]} x_i[t-l],$$
$$M[K,t] = \max_{i\in\mathcal{V},l\in[0,K]} x_i[t-l].$$

We then show for an arbitrary time t, $m[K,t]$ and $M[K,t]$ are monotonically non-decreasing and monotonically non-increasing, respectively. Consider node $i \in \mathcal{V}$ whose state

value $x_i[t]$ satisfies $m[K,t] \le x_i[t] \le M[K,t]$, after the filtering step, it will use the remaining values and trusted values $\psi_i[t]$ for updating. Note that each data in $\psi_i[t]$ is among the interval $[m[K,t], M[K,t]]$, the value for each node i at next time step is lower bounded by

$$\begin{aligned} x_i[t+1] &= w_{ii}[t]m[K,t] + \sum_{j\in\psi_i[t]} w_{ij}[t-l_{ij}]x_j[t-l_{ij}[t]] \\ &\ge w_{ii}[t]m[K,t] + \sum_{j\in\psi_i[t]} w_{ij}[t-l_{ij}]m[K,t] \\ &= m[K,t], \end{aligned}$$

where $w_{ii} = 1 - \sum_{j\in\psi_i[t]} w_{ij}[t-l_{ij}]$. And similarly, we can get that $x_i[t+1] \le M[K,t]$. Consequently, we conclude for each node i, $m[K,t] \le x_i[t+1] \le M[K,t]$. Therefore, for an arbitrary time step t and $t^* \in \mathbb{Z}_+$, $m[K,t] \le m[K,t+1] \le m[K,t+2] \le \cdots \le m[K,t+t^*-1] \le m[K,t+t^*] \le M[K,t+t^*] \le M[K,t+t^*-1] \le \cdots \le M[K,t+2] \le M[K,t+1] \le M[K,t]$. Thus we proved $m[K,t]$ and $M[K,t]$ have non-decreasing monotone and non-increasing monotone respectively.

Next, we prove the agreement condition in Definition 4. From the convergence of $m[K,t]$ and $M[K,t]$, there is a finite time t^* that they both reach their final value, denoted by m^* and M^* respectively. Formally, $\lim_{t\to t^*} m[K,t] = m^*$ and $\lim_{t\to t^*} M[K,t] = M^*$. Next we need to prove that for a finite time t^*, $m^* = M^*$. We conduct a proof by contradiction to prove that this conclusion must be established.

Before starting to prove, we give a conclusion to show the max change of state values of each node among a period K. That is, for any $0 \le l \le K$, we have

$$|x_i[t] - x_i[t-l]| \le (1-\mu^T)(M[K,t-K] - m[K,t-K]). \tag{11}$$

Now we show the detail of the above conclusion. Consider the lower-bounded weight we denoted as μ, which satisfies $\mu \in (0, 1/2)$. Then we have

$$\begin{aligned} x_i[t+1] &= w_{ii}[t]x_i[t] + \sum_{j\in\psi_i[t]} w_{ij}[t-l_{ij}]x_j[t-l_{ij}] \\ &\le \mu x_i[t] + (1-\mu)M[K,t], \end{aligned}$$

after $l \le K$ steps, $x_i[t+l] \le \mu^l x_i[t] + (1-\mu^l)M[K,t]$. Then,

$$\begin{aligned} |x_i[t+l] - x_i[t]| &\le \mu^l x_i[t] + (1-\mu^l)M[K,t] - x_i[t] \\ &= (1-\mu^l)(M[K,t] - x_i[K,t]) \\ &\le (1-\mu^l)(M[K,t] - m[K,t]). \end{aligned}$$

Hence, we have $|x_i[t] - x_i[t-l]| \le (1-\mu^T)(M[K,t-K] - m[K,t-K])$, $\forall l \le K$.

Now, we continue to prove the Theorem 2. On the contrary, suppose that $m^* \ne M^*$. Without loss of generality, suppose that $m^* < M^*$. Then, $\exists \gamma_0 \in \mathbb{R}^+$, such that $m^* + \gamma_0 < M^* - \gamma_0$. Moreover, for any time step t and $\gamma_c \in \mathbb{R}^+$, we define

$$S_m[K,t,\gamma_c] = \{j \in \mathcal{V} | x_j[t-l] < m^* + \gamma_c, \exists 0 \le l \le K\}, \tag{12}$$

$$S_M[K,t,\gamma_c] = \{j \in \mathcal{V} | x_j[t-l] > M^* - \gamma_c, \exists 0 \le l \le K\}. \tag{13}$$

The $S_m[K,t,\gamma_c]$ is the set of nodes in which each node has a state value smaller than $m^* + \gamma_c$ at least once in the past K time steps from time t. Similarly, $S_M[K,t,\gamma_c]$ is the set of nodes in which each node has a state value larger than $M^* - \gamma_c$ at least once in the time interval $[t-K, t]$.

Now, assuming $|\mathcal{V}|$ is the total number of nodes, we set $\gamma < \frac{\mu^{K^2|\mathcal{V}|}}{1-\mu^{K^2|\mathcal{V}|}}\gamma_0$. Obviously, $\gamma_0 > \gamma > 0$. Note that there exists t_γ such that $m[K,t] > m^* - \gamma$ and $M[K,t] < M^* + \gamma$, $\forall t \ge t_\gamma$.

To analyse with the property of robustness, the two selected sets need to be nonempty and disjoint. Next, we show $S_m[K,t_\gamma,\gamma_0]$ and $S_M[K,t_\gamma,\gamma_0]$ satisfy these two constraints.

Consider that $S_m[K,t_\gamma,\gamma_0]$ and $S_M[K,t_\gamma,\gamma_0]$ are both nonempty, for there must exists node i and j whose value equals to $m[t]$ and $M[t]$, respectively. In order to guarantee these two sets $S_m[K,t_\gamma,\gamma_0]$ and $S_M[K,t_\gamma,\gamma_0]$ are disjoint, we need to let maximum state value of nodes in the set $S_m[K,t_\gamma,\gamma_0]$ in time interval $[t-K,t]$ smaller than $M^* - \gamma_0$. For this goal, we set $\gamma_0 < \frac{\mu^T}{2}(M^* - m^*)$. Next we will show if $\gamma_0 < \frac{\mu^T}{2}(M^* - m^*)$, $S_m[K,t_\gamma,\gamma_0]$ and $S_M[K,t_\gamma,\gamma_0]$ must be disjoint.

Now, we deduce the conclusion of $\gamma_0 < \frac{\mu^T}{2}(M^* - m^*)$. According to (11), the difference between the maximum and minimum state value in past K time steps satisfies the formula $(M^* - \gamma_0) - (m^* + \gamma_0) \ge (1-\mu^T)(M[K,t-K] - m[K,t-K])$, then we compute this formula and the upper limit of γ_0 is shown as follows

$$\begin{aligned} \gamma_0 &\le (M^* - m^* - (1-\mu^T)(M[K,t-K] - m[K,t-K]))/2 \\ &\le (M^* - m^* - (1-\mu^T)(M^* - m^*))/2 \\ &\le \frac{\mu^T}{2}(M^* - m^*). \end{aligned}$$

Moreover, $\frac{\mu^T}{2}(M^* - m^*) < \frac{1}{2}(M^* - m^*)$, such a γ_0 must exists. Now, $S_m[K,t_\gamma,\gamma_0]$ and $S_M[K,t_\gamma,\gamma_0]$ are disjoint by the definition of γ_0.

Note that the network $G^K[t]$ is $(K, 2F+1)$-robust with $\mathcal{T}_\varepsilon$, for $t \ge K$, and there are no more than F compromised incoming links of each node in the time interval $[t-K,t]$ ((K,F)-local model), there exists a node in $S_m[K,t_\gamma,\gamma_0] \cup S_M[K,t_\gamma,\gamma_0]$ who has at least $(2F+1)$ incoming links or one trusted incoming link outside of its own set. Assume that $i \in S_m[K,t_\gamma,\gamma_0]$ is such a node.

The messages transmitted through the non-compromised links to node i were not modified during transmission. Hence, there exists a time step $(t+t')$, $1 \le t' \le K$, $t' \in \mathbb{Z}_+$, after filtering step, at least one of the remaining value larger than $m^* + \gamma_0$ that will be used for computing $x_i[t_\gamma + t']$.

Note the smallest value that node i will use at time-step t_γ is $m[K,t_\gamma]$, placing the largest possible weight on $m[K,t_\gamma]$,

the $x_i[t_\gamma + t']$ is lower bounded by

$$\begin{aligned} x_i[t_\gamma + t'] &\geq (1-\mu)m[K, t_\gamma] + \mu(m^* + \gamma_0) \\ &\geq (1-\mu)(m^* - \gamma) + \mu(m^* + \gamma_0) \\ &= m^* + \mu\gamma_0 - \gamma(1-\mu) \\ &= m^* + \gamma_{t'}. \end{aligned}$$

Here, $\gamma_{t'} = \mu\gamma_0 - \gamma(1-\mu)$, we get $\gamma_{t'} < \gamma_0$. Similarly, if $i \in S_M[K, t_\gamma, \gamma_0]$, then we can also get $x_i[t_\gamma + t'] \geq M^* - \gamma_{t'}$.

Then, for any time step $t_\gamma + j$, the above analysis can be repeated as long as $S_m[K, t_\gamma + j, \gamma_j]$ and $S_M[K, t_\gamma + j, \gamma_j]$ both contain nodes.

After K' steps, $K' \in [K, K^2]$, $x_i[t_\gamma + K'] \geq m^* + \gamma_{k'}$, where $\gamma_{k'} = \mu^{k'}\gamma_0 - \gamma(1-\mu^{k'})$, $\gamma_{k'} < \gamma_{k'-1} < \ldots < \gamma_1 < \gamma_0$. It means at least one node in $S_m[K, t_\gamma, \gamma_0]$ increases at least to $m^* + \gamma_{k'}$, or one of the nodes in $S_M[K, t_\gamma, \gamma_0]$ decreases at most to $M^* - \gamma_{k'}$. It must be that either the number of nodes in $S_m[K, t_\gamma + K', \gamma_{k'}]$ is strictly lesser than the nodes in $S_m[K, t_\gamma, \gamma_0]$ or the number of nodes in $S_M[K, t_\gamma + K', \gamma_{k'}]$ is strictly lesser than the nodes in $S_M[K, t_\gamma, \gamma_0]$.

Since the number of nodes is finite, there exists a time step $\tilde{t} = t_\gamma + K^2|\mathcal{V}|$ such that $S_m[K, t_\gamma + \tilde{t}, \gamma_{\tilde{t}}] = \varnothing$ or $S_M[K, t_\gamma + \tilde{t}, \gamma_{\tilde{t}}] = \varnothing$. It implies that the minimum value of nodes is lower bounded by $m^* + \gamma_{\tilde{t}}$, or the maximum value of nodes is upper bounded by $M^* - \gamma_{\tilde{t}}$, respectively.

If $\gamma_{\tilde{t}} > 0$, then $S_m[K, t_\gamma + \tilde{t}, \gamma_{\tilde{t}}] = \varnothing$ implies a contradiction to the fact that smallest value converges monotonically to m^*, and $S_M[K, t_\gamma + \tilde{t}, \gamma_{\tilde{t}}] = \varnothing$ contradicts to the fact that the largest value converges monotonically to M^*. Next, we show that $\gamma_{\tilde{t}} > 0$. Recall $\gamma < \frac{\mu^{k^2|\mathcal{V}|}}{1-\mu^{k^2|\mathcal{V}|}}\gamma_0$, we get

$$\begin{aligned} \gamma_{\tilde{t}} &> \mu^{\tilde{t}}\gamma_0 - \gamma(1-\mu^{\tilde{t}}) \\ &> \mu^{\tilde{t}}\gamma_0 - \frac{\mu^{k^2|\mathcal{V}|}}{1-\mu^{k^2|\mathcal{V}|}}(1-\mu^{\tilde{t}}) \\ &= \mu^{k^2|\mathcal{V}|}\gamma_0 - \frac{\mu^{k^2|\mathcal{V}|}}{1-\mu^{k^2|\mathcal{V}|}}\gamma_0(1-\mu^{k^2|\mathcal{V}|}) \\ &= 0. \end{aligned}$$

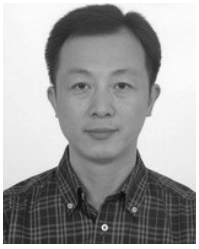
It results in a contradiction. Therefore, we have $m^* = M^*$.

REFERENCES

[1] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: A survey," *IEEE Netw.*, vol. 18, no. 4, pp. 45–50, Jul. 2004.
[2] Z. Zhong and T. He, "MSP: Multi-sequence positioning of wireless sensor nodes," in *Proc. ACM 5th Int. Conf. Embedded Netw. Sensor Syst.*, 2007, pp. 15–28.
[3] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon, "Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2009, pp. 277–288.
[4] S. Zhu, C. Chen, X. Ma, B. Yang, and X. Guan, "Consensus based estimation over relay assisted sensor networks for situation monitoring," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 2, pp. 278–291, Mar. 2015.
[5] D. Hauden, E. Bigler, A. d'Almeida, S. Ballandras, and P. Leblois, "Frequency instabilities in saw resonators and oscillators," in *Proc. IEEE Ultrason. Symp.*, Oct. 1989, pp. 53–57.
[6] Z. Zhong, P. Chen, and T. He, "On-demand time synchronization with predictable accuracy," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2480–2488.
[7] X. Hu, T. Park, and K. G. Shin, "Attack-tolerant time-synchronization in wireless sensor networks," in *Proc. IEEE INFOCOM 27th Conf. Comput. Commun.*, Apr. 2008, pp. 41–45.
[8] Y. Kikuya, S. M. Dibaji, and H. Ishii, "Fault-tolerant clock synchronization over unreliable channels in wireless sensor networks," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 4, pp. 1551–1562, Dec. 2017.
[9] J. He, P. Cheng, L. Shi, J. Chen, and Y. Sun, "Time synchronization in WSNs: A maximum-value-based consensus approach," *IEEE Trans. Autom. Control*, vol. 59, no. 3, pp. 660–675, Mar. 2014.
[10] L. Schenato and F. Fiorentin, "Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks," *Automatica*, vol. 47, no. 9, pp. 1878–1886, Sep. 2011.
[11] S. Bolognani, R. Carli, E. Lovisari, and S. Zampieri, "A randomized linear algorithm for clock synchronization in multi-agent systems," *IEEE Trans. Autom. Control*, vol. 61, no. 7, pp. 1711–1726, Jul. 2016.
[12] P. Sommer and R. Wattenhofer, "Gradient clock synchronization in wireless sensor networks," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2009, pp. 37–48.
[13] L. Schenato and G. Gamba, "A distributed consensus protocol for clock synchronization in wireless sensor network," in *Proc. 46th IEEE Conf. Decis. Control*, Dec. 2007, pp. 2289–2294.
[14] H. Li, J. He, P. Cheng, and J. Chen, "Consensus-based time synchronization in sensor networks: An experimental study," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 207–212.
[15] Y. Wu, X. He, and S. Liu, "Resilient consensus for multi-agent systems with quantized communication," in *Proc. IEEE Amer. Control Conf. (ACC)*, Jul. 2016, pp. 5136–5140.
[16] S. M. Dibaji and H. Ishii, "Consensus of second-order multi-agent systems in the presence of locally bounded faults," *Syst. Control Lett.*, vol. 79, pp. 23–29, May 2015.
[17] N. H. Vaidya, L. Tseng, and G. Liang, "Iterative approximate byzantine consensus in arbitrary directed graphs," in *Proc. ACM Symp. Principles Distrib. Comput.*, 2012, pp. 365–374.
[18] A. Haseltalab and M. Akar, "Approximate byzantine consensus in faulty asynchronous networks," in *Proc. IEEE Amer. Control Conf. (ACC)*, Jul. 2015, pp. 1591–1596.
[19] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram, "Resilient asymptotic consensus in robust networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 4, pp. 766–781, Apr. 2013.
[20] J. He, J. Chen, P. Cheng, and X. Cao, "Secure time synchronization in wirelesssensor networks: A maximumconsensus-based approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 4, pp. 1055–1065, Apr. 2014.
[21] W. Dong and X. Liu, "Robust and secure time-synchronization against Sybil attacks for sensor networks," *IEEE Trans. Ind. Informat.*, vol. 11, no. 6, pp. 1482–1491, Dec. 2015.
[22] Y. Dong, N. Gupta, and N. Chopra, "Content modification attacks on consensus seeking multi-agent system with double-integrator dynamics," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 26, no. 11, 2016, Art. no. 116305.
[23] Y. Wang, "Privacy-preserving average consensus via state decomposition," *IEEE Trans. Autom. Control*, to be published.
[24] H. Zhang and S. Sundaram, "Robustness of information diffusion algorithms to locally bounded adversaries," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2012, pp. 5855–5861.
[25] W. Abbas, A. Laszka, Y. Vorobeychik, and X. Koutsoukos, "Improving network connectivity using trusted nodes and edges," in *Proc. IEEE Amer. Control Conf. (ACC)*, May 2017, pp. 328–333.
[26] J. Huang, Y. Wu, L. Chang, M. Tao, and X. He, "Resilient consensus with switching networks and heterogeneous agents," *Neurocomputing*, vol. 341, pp. 70–79, May 2019.
[27] A. Mitra, W. Abbas, and S. Sundaram, "On the impact of trusted nodes in resilient distributed state estimation of LTI systems," in *Proc. IEEE Conf. Decis. Control (CDC)*, Dec. 2018, pp. 4547–4552.
[28] S. M. Dibaji and H. Ishii, "Resilient consensus of second-order agent networks: Asynchronous update rules with delays," *Automatica*, vol. 81, pp. 123–132, Jul. 2017.
[29] D. M. Senejohnny, S. Sundaram, C. De Persis, and P. Tesi, "Resilience against misbehaving nodes in asynchronous networks," *Automatica*, vol. 104, pp. 26–33, Jun. 2019.
[30] J. He, P. Cheng, L. Shi, and J. Chen, "SATS: Secure average-consensus-based time synchronization in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 61, no. 24, pp. 6387–6400, Dec. 2013.

[31] C. Zhao, J. He, P. Cheng, and J. Chen, "Secure consensus against message manipulation attacks in synchronous networks," in *Proc. World Congr.*, vol. 19, 2014, pp. 1182–1187.
[32] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Proc. ACM 1st Int. Conf. High Confidence Netw. Syst.*, 2012, pp. 55–64.
[33] S. Sundaram, S. Revzen, and G. Pappas, "A control-theoretic approach to disseminating values and overcoming malicious links in wireless networks," *Automatica*, vol. 48, no. 11, pp. 2894–2901, 2012.
[34] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Trans. Autom. Control*, vol. 58, no. 11, pp. 2715–2729, Nov. 2013.
[35] Y. Wu, X. He, S. Liu, and L. Xie, "Consensus of discrete-time multi-agent systems with adversaries and time delays," *Int. J. Gen. Syst.*, vol. 43, nos. 3–4, pp. 402–411, 2014.

SHULING HUANG received the B.E. degree in network engineering from Zhejiang Gongshang University, Hangzhou, China, in 2017. Since 2017, she has been with the Laboratory of Internet and Network Security, Hangzhou Dianzi University, Hangzhou. Her current research interests include multi-agent systems and wireless sensor networks.



NING ZHENG received the M.S. degree from Zhejiang University, Hangzhou, China, in 1990. He is currently a Full Professor with Hangzhou Dianzi University, Hangzhou, China. His current research interests include multi-agent networks and information security.

YIMING WU received the B.E. degree in automation and the Ph.D. degree in control science and engineering from the Zhejiang University of Technology, China, in 2010 and 2016, respectively. From 2012 to 2014, he was a Research Assistant with Nanyang Technological University, Singapore. Since July 2016, he has been with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include multi-agent systems, resilient consensus, and secure control systems.

MING XU received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2004. He is currently a Full Professor with Hangzhou Dianzi University, Hangzhou. His research interests include network security and digital forensics.