Dang F, Sun XK, Liu KB *et al.* A survey on clock synchronization in the Industrial Internet. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 38(1): 2023. DOI: 10.1007/s11390-023-2908-4

A Survey on Clock Synchronization in the Industrial Internet

Fan Dang[1,†] (党 凡), *Member, CCF, ACM, IEEE*, Xi-Kai Sun[2,†] (孙熙凯)
Ke-Bin Liu[1] (刘克彬), *Member, CCF, ACM, IEEE*, Yi-Fan Xu[3] (许逸凡), and
Yun-Hao Liu[1,2,*] (刘云浩), *Fellow, CCF, ACM, IEEE*

[1] *Global Innovation Exchange, Tsinghua University, Beijing 100084, China*
[2] *Department of Automation, Tsinghua University, Beijing 100084, China*
[3] *School of Software, Tsinghua University, Beijing 100084, China*

E-mail: dangfan@tsinghua.edu.cn; sxk18@mails.tsinghua.edu.cn; kebinliu2021@tsinghua.edu.cn; xuyifan20@mails.tsinghua.edu.cn; yunhao@tsinghua.edu.cn

Received October 15, 2022; accepted January 9, 2023.

Abstract Clock synchronization is one of the most fundamental and crucial network communication strategies. With the expansion of the Industrial Internet in numerous industrial applications, a new requirement for the precision, security, complexity, and other features of the clock synchronization mechanism has emerged in various industrial situations. This paper presents a study of standardized clock synchronization protocols and techniques for various types of networks, and a discussion of how these protocols and techniques might be classified. Following that is a description of how certain clock synchronization protocols and technologies, such as PROFINET, Time-Sensitive Networking (TSN), and other well-known industrial networking protocols, can be applied in a number of industrial situations. This study also investigates the possible future development of clock synchronization techniques and technologies.

Keywords clock synchronization, industrial communication, Industrial Internet, wireless sensor network

1 Introduction

The Industrial Internet is a new Industrial form that integrates communication technology and the Industrial economy in a comprehensive manner. It is the infrastructure for the industry's transformation to digitization, networking, and intelligence. By introducing the Internet of Things technology into industry, the Industrial Internet has the capabilities of production scheduling, location tracking, environmental monitoring, and production management, and plays a vital role in enhancing Industrial production efficiency and reshaping the Industrial form. For instance, in the smartphone manufacturing industry, the use of Industrial Internet technology for wireless connection of production equipment and flexibly manufacturing products can reduce the cost of production line adjustments and create more economic value with faster iterations.

Clock synchronization plays a fundamentally crucial role in a variety of Industrial Internet based services, including but not limited to the aforementioned services. Using production scheduling as an example, if the data transmitted from certain device nodes is not time-aligned, it can lead to device behavior conflicts, unnecessary costs, and even device damage. This survey focuses on clock synchronization strategies for Industrial Internet wireless nodes. It should be noted that the clock synchronization mentioned above includes aligning the clock frequency,

Survey

Special Issue in Honor of Professor Kai Hwang's 80th Birthday

This work is supported in part by the National Key Research and Development Program of China under Grant No. 2021YFB 2900100.

[†]Co-Primary Authors (Fan Dang is responsible for designing the framework, conducting the research, and writing for the Applications and Related Work sections. Xi-Kai Sun is in charge of researching and writing the Preliminaries and Protocol sections.)

[*]Corresponding Author

initial offset, and time value[1], which will be introduced in Section 2.

Before discussing the clock synchronization mechanism of the Industrial Internet, we also need to discuss the Industrial Internet's clock synchronization requirements. In contrast to the conventional wired packet-switched network, the Industrial Internet encompasses both wired and wireless networks. In Industrial applications, the requirements for clock synchronization vary depending on the scenario, which is one of the new obstacles encountered in the deployment of the Industrial Internet. This paper categorizes these requirements into the points listed below.

- *Accuracy.* In various Industrial production scenarios, applications have varying requirements for clock synchronization precision. Various measurement and control data in the power grid, for instance, require an accuracy of 100 ms to 1 μs or better[2].
- *Rapidity.* In certain scenarios, nodes periodically synchronize their clocks; in some nodes in the duty cycle mode, clock synchronization must be completed during the duty cycle slot phase, and nodes must remain in the sleep mode to conserve energy[3]. It implies that a complete clock synchronization process must be completed as quickly as possible within a predetermined time limit, or that it must be adaptable to ensure that the synchronization is completed within the time limit while maximizing its accuracy. Rapidity in a distributed network corresponds to the increased convergence speed[4–6].
- *Security.* Due to the diversity of devices and communication protocols, the security of clock synchronization on the Industrial Internet is more challenging[7]. Given the inevitability of potential attacks such as Sybil attacks[8], secure clock synchronization is required in harsh environments by default[9]. Besides, many clock synchronization protocols designed in ideal scenarios are ineffective because they cannot defend against specific attacks[10–12].
- *Robustness.* The meaning of robustness is that a network of nodes has a structurally loose topology[13], resulting in high fault tolerance, i.e., when a node fails, it does not affect the clock synchronization of the entire network, and scalability, i.e., when a new node is added, the network can achieve clock synchronization of all nodes at a low cost. In addition, random packet loss is a common occurrence in wireless networks for nodes that exchange time information in pairs due to link constraints, packet collisions, and other factors. Therefore, when the clock is being synchronized, retransmission or broadcast must be considered to reduce the packet loss rate[14].
- *Low Complexity.* In the Industrial Internet, there are numerous low-cost devices with limited storage, computing, and communication capabilities, and the majority are powered by batteries[15]. Some applications even require a single battery to provide energy for decades[11]. Therefore, the clock synchronization process at the sensor node requires low computational complexity and few data packets, and the possible complex computing process should be transferred to the cloud, server, or other edge devices with more resources in order to accommodate the limited resources[16–20] of nodes and conserve energy.

In addition, there may be other factors that affect clock synchronization in specific Industrial Internet applications. For example, in certain Industrial settings, problems such as extreme temperature fluctuations are inevitable. Temperature-induced clock drift poses a challenge for conventional clock synchronization techniques in a dynamic Industrial environment[16]. A change of a few degrees in the experiment[21] will cause the clock skew to fluctuate on the order of 10^{-5}, which is extremely unfavorable for high-frequency clocks. In addition, sensor equipment nodes in conventional Industrial networks are typically heterogeneous, and nodes have varying quality of service (QoS) requirements[22]. Therefore, the clock synchronization protocols applied to those Industrial Internet applications must support the clock synchronization of heterogeneous nodes operating at variable temperatures.

The rich application scenarios of the Industrial Internet impose the aforementioned requirements on clock synchronization, but many of these requirements frequently conflict. For instance, greater precision frequently necessitates more processing time and more frequent data interactions, resulting in slower convergence and increased energy consumption. In addition, time synchronization security necessitates an increase in resources, energy consumption, and calculation complexity. It is essential, when designing clock synchronization protocols, to strike a balance between these requirements. This paper will begin with a classification of synchronization mechanisms, then introduce various clock synchronization methods and strategies, analyze their practical application in Industrial communications, and conclude with a discussion of promising future developments.

The rest of the paper is organized as follows. In

Section 2, clock synchronization concepts are introduced, the mathematical form of clock synchronization is simply derived, and the delays that maybe occur in the time synchronization process are analyzed. In Section 3, we provide an overview of time synchronization mechanism suitable for the Industrial Internet, as well as some guidance for this field. In Section 4, we briefly list some practical applications requiring clock synchronization in the Industrial Internet. In Section 5, we summarize potential enhancements directions based on the requirements of the Industrial Internet for clock synchronization. We review the related work in Section 6 and conclude this survey in Section 7.

2 Preliminary

2.1 Clock Synchronization Concept

By counting the periodic output pulses of its crystal oscillator, the network node generates its own local clock. Due to variations in the processing technology and environment of inexpensive crystal oscillators, there are frequently minor variations between crystal oscillators at various nodes. Generally, crystal oscillator output pulse frequency accuracy is measured in parts per million (ppm) or parts per billion (ppb)[23]. For instance, 1 ppm indicates that the frequency of the current node differs from the standard average crystal frequency by 1 Hz per 1 MHz. In addition, the starting point of the counting of various nodes is distinct, meaning that the initial value of each node's clock is distinct. Since the clocks have been used, there has been a clock asynchrony problem in the clocks of nodes due to the difference among cheap crystal oscillators in the counting starting point. After each clock synchronization is completed, the clock will still be out of synchronization due to the difference in crystal oscillator frequency. Therefore, clock synchronization is a continuous process that nodes must execute.

Clock synchronization is the process of unifying the clocks of endpoint devices. After each node shares the same time scale, the causal relationship between events in the physical world can be determined, and each node can be truly integrated into a network as a whole. Clock synchronization, broadly speaking, entails aligning clock frequency, initial offset, and time value, which in this context means the followings.

- *Frequency Synchronization.* The counting frequency of all nodes' clocks is identical, or frequency errors of all clocks are maintained within a predetermined error range.
- *Initial Offset Synchronization.* The crystal oscillator counting process of all nodes begins simultaneously, i.e., the clocks are instantaneously synchronized to the same value, or clock errors are kept within a specified error range.
- *Time Synchronization.* All nodes' clocks are synchronized with the reference clock, and then share the same synchronization period and time scale.

Intuitively, the three synchronization aspects are shown in Fig.1. The combination of frequency synchronization and initial offset synchronization results in time synchronization.

2.2 Clock Model

Based on the actual counting process of the crystal oscillator, the local clock model $C_{i,t}$ of node i at time t is as follows[24]:

$$C_{i,t} = \int_0^t \gamma_{i,\tau} \mathrm{d}\tau + \theta_{i,0},$$

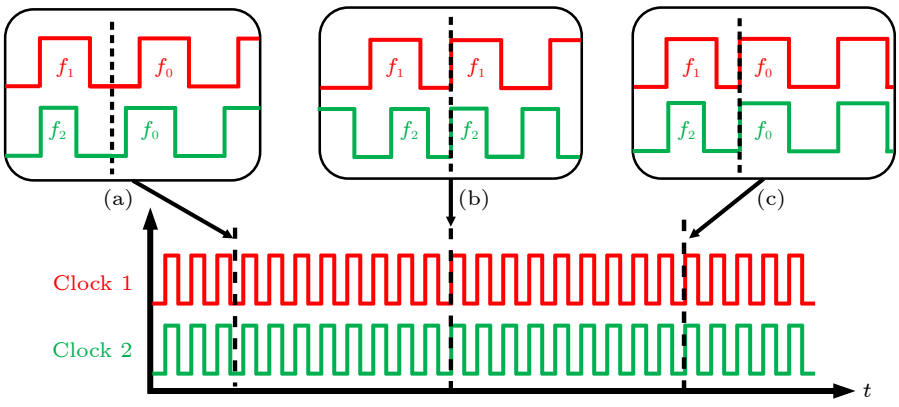


Fig.1. Three synchronization methods. (a) Frequency synchronization. (b) Initial offset synchronization. (c) Time synchronization.

where $\gamma_{i,\tau}$ is the ratio of the real-time clock frequency $f_{i,\tau}$ of node i to the reference clock frequency f_0, i.e.,

$$\gamma_{i,\tau} = \frac{f_{i,\tau}}{f_0}.$$

$\gamma_{i,\tau}$ is a time-related parameter because it may be affected by environmental factors such as temperature at different time points; $\theta_{i,0}$ is the time value of node i at time 0, or the initial clock offset of node i; and time t is the reference clock, or the convergent clock in the distributed network.

As an approximation, the majority of studies assume that the temperature remains essentially constant for a finite period of time[16, 18, 25–28], and then the first-order affine hardware clock model of the local clock $C_{i,t}$ of node i at time t[21] is defined as

$$C_{i,t} = \gamma_i \times t + \theta_{i,0},$$

where γ_i is the clock frequency ratio, which remains substantially stable for a limited time. Therefore we can obtain the clock offset of node i at time t compared with the reference clock as follows:

$$\theta_{i,t} = C_{i,t} - t = (\gamma_i - 1) \times t + \theta_{i,0}.$$

$\delta_i = \gamma_i - 1$ is called the clock skew of node i. For any node $i \in \{1, 2, \ldots, N\}$, the initial offset synchronization is intended to make the initial clock offset $\theta_{i,0}$ of each node essentially identical, the frequency synchronization is intended to set the frequency ratio γ_i of each node converge to 1 or the clock skew δ_i converge to 0, and the time synchronization is intended to make the time $C_{i,t}$ or clock offset $\theta_{i,t}$ of each node basically consistent. Consequently, the fundamental problem in clock synchronization research is how to more accurately estimate clock skew and initial clock offset to compensate, i.e., the clock synchronization process is as follows:

$$C_{i,t} \leftarrow C_{i,t} - \hat{\delta}_i \times t - \hat{\theta}_{i,0}, \tag{1}$$

or

$$C_{i,t} \leftarrow C_{i,t} - \hat{\theta}_{i,t},$$

where $\hat{\delta}_i$ and $\hat{\theta}_{i,0}$ are the estimated values of node i's clock skew and initial offset, respectively.

Furthermore, in some simple protocols, clock synchronization is interpreted as an update of the timestamp[14, 29–31], i.e., a clock synchronization process requires only:

$$C_{i,t} \leftarrow t, \tag{2}$$

where t is the reference time recorded by timestamp. Since there is no clock skew compensation, after a synchronization process is complete, different nodes will become asynchronous again more rapidly, reducing accuracy and necessitating more frequent synchronization.

2.3 Delay Analysis

However, the above clock model, represented by (2), is only a first-order ideal model that disregards the effects of temperature variations and communication delays. For nodes in wireless communication, the data used for clock synchronization is dependent on the data packet transmission between nodes, and thus the uncertainty of data packet delay will result in errors in the estimation of clock parameters. Li *et al.*[32] examined the potential delays in the wireless Industrial Internet packet transmission process as follows.

- *Sending Time* T_s. Sending time is the total amount of time it takes for the sender's message to t from the application layer to the network layer, which is uncertain.
- *Access Time* T_a. Access time is the delay caused by the MAC layer during the channel access procedure of the sender, which is related to protocols.
- *Transmission Time* T_t *and Reception Time* T_{rp}. Transmission/reception time is the time required for the sender/receiver to send/receive data packets at the physical layer. It is a definite time delay determined by the bandwidth of the wireless channel and the size of the data packets.
- *Propagation Time* T_p. Propagation time is the time required for a packet to transmit from the sender to the receiver on a wireless channel.
- *Receive Time* T_{rv}. Receive time is the total amount of time required for the receiver to upload the received data packet from the physical layer to the application layer, which is also uncertain.

The completed delay in the data packet exchange process is shown in Fig.2[32]. Therefore the total delay T_d can be calculated as

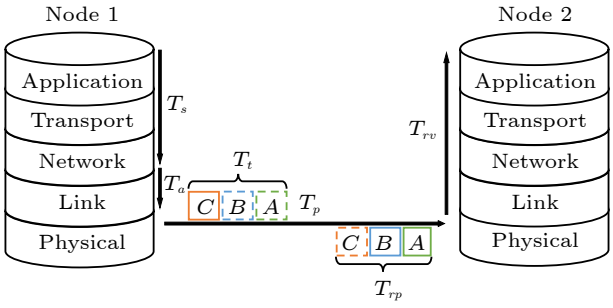


Fig.2. Delays in data packet exchange process[32].

$$T_d = T_s + T_a + T_t + T_p + T_{rv}.$$

It should be noted that the clock synchronization in the majority of Industrial sensing nodes utilizes MAC layer timestamps, and thus the influence of T_s, T_a and T_{rv} is minor. Moreover, T_p is typically much shorter than T_t/T_{rp} within the factory's limited distance. Consequently, transmission/reception time is the primary cause of delay in the Industrial network. After accounting for delays, the clock model can be formulated as follows:

$$C_{i,t} = \gamma_i \times t + \theta_{i,0} + T_d.$$

Therefore, the first-order model-based clock synchronization can be updated as follows:

$$C_{i,t} \leftarrow C_{i,t} - \hat{\delta}_i \times t - \hat{\theta}_{i,0} - \hat{T}_d,$$

or

$$C_{i,t} \leftarrow C_{i,t} - \hat{\theta}_{i,t} - \hat{T}_d,$$

where $\hat{T}_d$ is an estimated value of the delay from the sender node to the receiver node. Similarly, (2) can be updated as

$$C_{i,t} \leftarrow t + \widehat{T}_d. \tag{3}$$

Note that due to the existence of a delay, t in the symbol $C_{i,t}$ only represents the time t at which clock synchronization begins, and the local clock of node i actually changes after time i.

3 Clock Synchronization Mechanism

Considering the variety of clock synchronization protocols, we classify them at multiple levels, as depicted in Fig.3. Later in this section, the relationship between levels will be discussed.

First, we categorize the clock synchronization protocols into two groups based on the reference clock source: external clock source synchronization and internal message exchange synchronization. External clock source synchronization refers to providing an accurate clock source from the outside, typically using (2) or (3) to synchronize the clocks of nodes. This category has less computation and has a global clock; however, it requires an accurate clock source to provide a reference clock for each independent node, such as Global Navigation Satellite System (GNSS) time servers and timekeeping radio station synchronization (e.g., WWVB in the US)[18]. To support the previously mentioned clock source, the node must also load the corresponding signal receiver equipment, which is relatively expensive. Internal message exchange synchronization signifies that all nodes in the Industrial Internet network are synchronized to the same clock via the message exchange between nodes in accordance with a particular topology structure. This category does not require each node to exchange clock information with the reference clock source, and in some cases does not even rely on an external clock source[17, 26, 31, 33], resulting in low hardware costs.

For external clock source synchronization, limited by expensive GPS receivers, additional relay modules, and poor GPS indoor signals, the traditional protocols based on GPS time server-client or timekeeping radio station synchronization cannot be widely used in the Industrial Internet, although these protocols even have accuracy up to nanosecond-level[34]. Some researchers attempt to obtain time information from

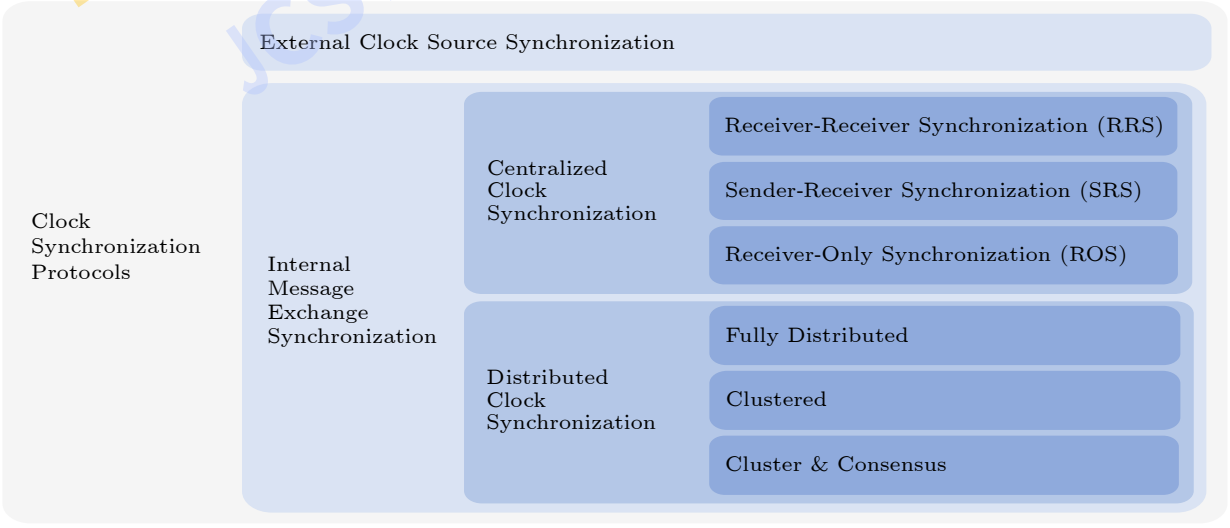


Fig.3. Multilevel classification of clock synchronization protocols.

other signals in order to create a more precise global clock for all nodes. The protocol AirSync[18] uses widely available aircraft as the sender and the aircraft signal ADS-B to perform time synchronization among nodes, achieving the indoor nodes' synchronization without requiring communication between nodes. Unfortunately, ADS-B does not contain timestamp information; instead, AirSync estimates the time value through aircraft movement and can achieve sub-millisecond accuracy. A node can obtain a reference clock using free tamper-proof timestamp data provided by public blockchains, such as Ethereum, according to [5]. A node updates its clock by passively receiving the blockchain block header, extracting the timestamp, and estimating the elapsed time between consecutive blocks as a delay, achieving only second-level accuracy. Since no communication between nodes and the blockchain consensus mechanism guarantees block security, this synchronization scheme is secure, decentralized, and resistant to external attacks.

More schemes fall into the category of internal message exchange synchronization. This paper divides nodes into two categories based on their topological relationship: centralized clock synchronization and distributed clock synchronization. Simple topologies for both classes are shown in Fig.4. These two types of protocols will be discussed in detail in Subsection 3.1 and Subsection 3.2, respectively.

3.1 Centralized Clock Synchronization

Centralized clock synchronization relies on specific source nodes and hierarchical topology[26], and nodes form a spanning tree topology by communicating with adjacent-layer nodes. In this spanning tree topology, the root node is typically one or more nodes that can communicate with external time sources to synchronize the network's clock. This type of clock synchronization involves the transmission of reference clock information along the tree to descendants. Thus, the clock synchronization of each layer's nodes is achieved. Protocols based on centralized clock synchronization typically have a fast convergence speed and small synchronization errors, but rely on a strong tree topology structure, have poor dynamic scalability, and require complex algorithms to handle issues such as internal node failures and the addition of new nodes. Such protocols usually have high computational complexity and poor robustness[27]. In addition, security must be considered due to the need for a specific source node to provide a reference clock, which increases the likelihood of single point failure and Sybil attack[26]. When the root node of a spanning tree with a unique root fails, the clocks of the nodes in the network cannot be synchronized.

We will discuss centralized time synchronization protocols in terms of how parent and child nodes in a spanning tree communicate.

3.1.1 *Receiver-Receiver Synchronization (RRS)*

In the RRS model, if node C is to synchronize the clock of node A, nodes A and C must first receive the reference information from sender node B, and then receiver A must broadcast its own timestamp information, as depicted in Fig.5. After receiving this information, receiver C can synchronize its clock with that of A by adhering to certain rules. Using the as-

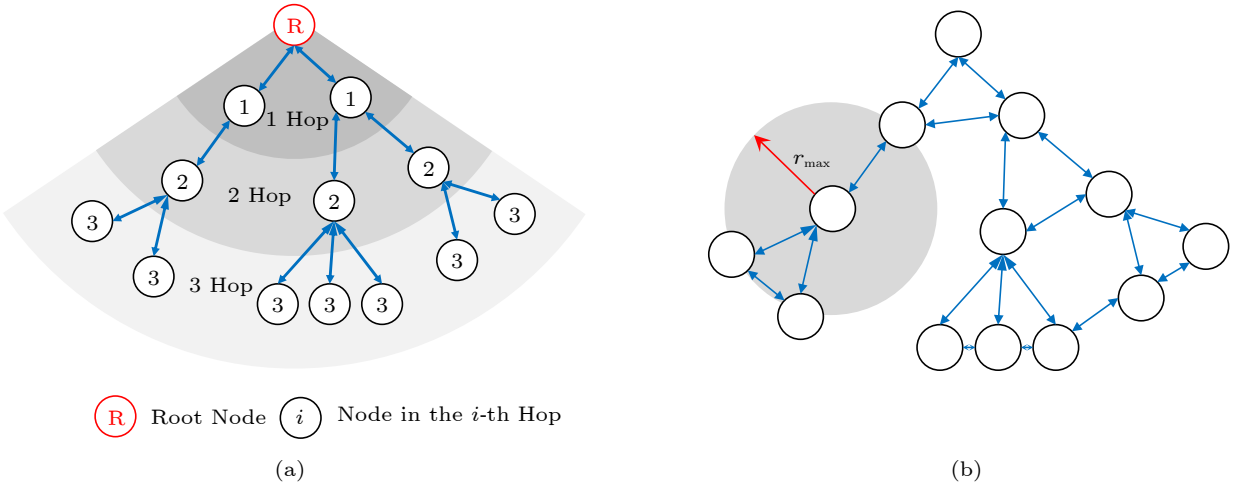


Fig.4. Two simple topologies. (a) Topology of centralized clock synchronization. (b) Topology of distributed clock synchronization. $r_{\max}$ means the maximum communication distance of a node.

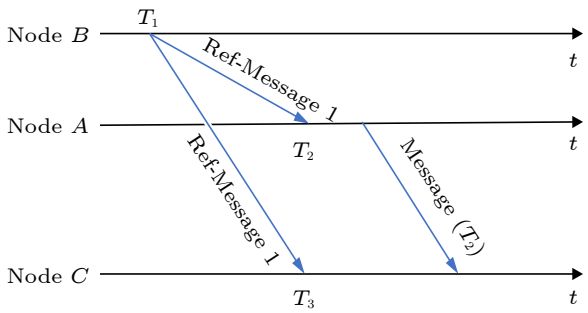


Fig.5. Data exchange sequence diagram of RRS communication model.

sumption that nodes A and C receive reference messages simultaneously, the clock offset between nodes A and C is $T_2 - T_3$. By comparing the timestamps between nodes A and C as opposed to nodes B and C, the first-step communication delay uncertainty is reduced[35].

Reference Broadcast Synchronization (RBS) is a classic RRS synchronization scheme, originally proposed in [10] for the clock synchronization of wireless sensor networks (WSNs). By exchanging messages in the form of RRS, all receivers will store the beacon information from other nodes, allowing each node to calculate the time difference between itself and other receivers. When the RRS process is repeated numerous times, linear regression can be utilized to determine the clock skew and initial offset[29]. RBS's precision can reach tens of microseconds[36]. In general, RBS is restricted to single-hop networks, but according to [37] RBS can achieve multi-hop synchronization by converting broadcast frequencies. The energy consumption is high because each node must continuously broadcast its own timestamp information and receive information from other receivers. In terms of security, there is a single point of failure due to the need for reference information from the sender[22].

To reduce communication costs, the energy-efficient Coefficient Exchange Synchronization Protocol (CESP) was proposed[38], which is based on the concept of RBS synchronization. CESP does not require MAC layer timestamps to conserve energy. Receivers first process a large number of time beacon reference packets from the sender, and then exchange "synchronization coefficients" (a time parameter defined in [38]) only once, thereby significantly reducing the number of packet exchanges per receiver node. This scheme focuses on synchronization between paired receivers, allowing it to be applied to spanning-tree multi-hop networks, but the single-point failure issue still exists.

3.1.2 *Sender-Receiver Synchronization (SRS)*

In the SRS model, node A is the sender of the reference timestamp, i.e., the master node with the reference clock, while node B is the receiver of the reference timestamp, i.e., the slave node. Through the two-way time information exchange, the slave node B estimates the clock offset in order to synchronize its own clock with that of the master node A. In some work[9, 34, 39], the master node may initiate the time synchronization process, or it may be necessary to exchange information multiple times, e.g., PTP, but it still requires direct message exchange between the master and slave nodes, still being regarded as the SRS mode. By synchronizing each pair of master-slave nodes in the spanning tree according to the SRS mode, the entire network's clock is synchronized. Due to the instability of the link between the two exchanges, however, an asymmetrical delay may occur. As the tree grows deeper, errors and asymmetric delays will gradually accumulate due to asynchronous synchronization.

The classic Timing-sync Protocol for Sensor Networks (TPSN) completes time synchronization in two stages: in the hierarchical discovery stage, each node is organized into a spanning tree structure; and then in the clock synchronization stage, starting from the root node, each pair of parent-child nodes is synchronized in turn according to the SRS mode shown in Fig.6, which can provide second-level accuracy. According to the SRS model shown in Fig.6, when a pair of master-slave nodes are synchronized using TPSN, the clock offset of the slave node is estimated as $\frac{(T_2-T_1)-(T_4-T_3)}{2}$, and the delay of the slave node is estimated as $\frac{(T_2-T_1)+(T_4-T_3)}{2}$. The internal node acts as the client of the parent node, synchronizing its own clock to be the same as the parent node; at the same time, it acts as the server of the child node, providing its

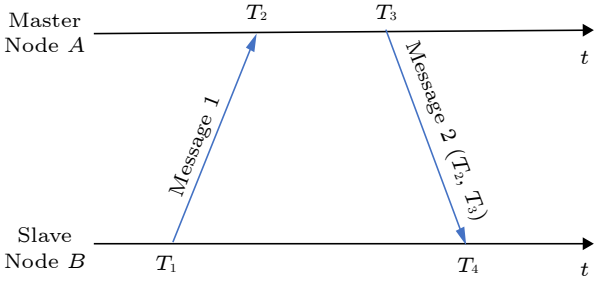


Fig.6. Data exchange sequence diagram of SRS communication model.

own synchronized clock as the reference clock of its child node[29]. To further reduce cost, based on TPSN, [39] proposes Speculative Precision Time Protocol (SPTP), which uses timestamps carried by packages used when paired nodes perform other display interaction functions rather than specially transmitting data packets for time synchronization. This method achieves sub-microsecond accuracy on the test bench. However, this method associates the frequency of time synchronization with the network traffic. When the network traffic is low, the synchronization accuracy may decrease.

Simple Network Time Protocol (SNTP) is widely utilized in embedded IoT systems[11] and can be unicast or broadcast according to the SRS mode illustrated in Fig.6. Although SNTP can achieve time synchronization at a low cost, it can only provide submillisecond accuracy under low latency and is unsuitable for large networks[11]. CoSiNeT[30] combines SNTP with the data packet exchange method SPoT[40] in order to ensure synchronization accuracy under conditions such as poor network conditions and significant delays. Compared with SNTP, the performance is improved by 76% without losing synchronization accuracy. Idrees *et al.*[34] compared several well-known clock synchronization protocols and concluded that the Precision Time Synchronization Protocol (PTP) may be the most suitable answer for robust clock synchronization in the Industrial networks, which can achieve sub-microsecond precision. The Best Master Clock (BMC) algorithm is used to construct the minimum spanning tree in PTP, followed by the exchange method shown in Fig.7. It is executed between each master and slave node to exchange clock information.

Modified-BMC designed in [41] optimizes the spanning tree construction process in PTP and introduces the out-degree of nodes as the basis for constructing spanning trees so as to generate a shallower tree to minimize the accumulation of errors and transmission delay. In general, the protocols mentioned above first construct a spanning tree using predefined spanning-tree building algorithms to associate all nodes.

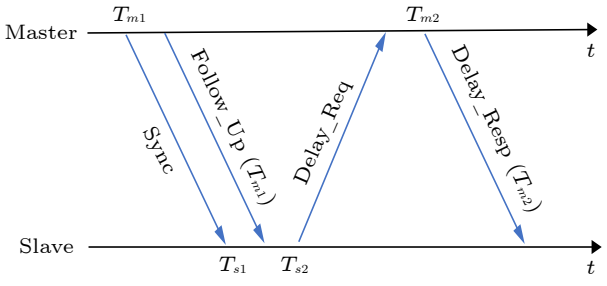


Fig.7. Data exchange sequence diagram of PTP. In PTP, the clock offset is estimated as $\frac{(T_{s1}-T_{m1})-(T_{m2}-T_{s2})}{2}$, and the delay is estimated as $\frac{(T_{s1}-T_{m1})+(T_{m2}-T_{s2})}{2}$.

As opposed to forming spanning-tree structures, flooding methods associate all nodes by forming ad-hoc structures, without the cost of building spanning trees[42]. The traditional Flooding Time Synchronization Protocol (FTSP) is a one-way message propagation protocol in which nodes organize themselves into an ad-hoc network and the root of the network is dynamically (re)elected[42]. Periodically, the reference node broadcasts its time information, and its flooding process is depicted in Fig.8. Each slave node must use linear regression to estimate the clock offset and clock skew between its own clock and the upper-level master node's synchronized clock and then correct its own clock according to certain rules, e.g., (1). Since message propagation is unidirectional from the root node in FTSP, it is impossible to estimate the delays of each hop, such as propagation delay and transmission delay. To estimate the delay in FTSP, the RSP protocol was proposed[43], which employs two packets with MAC layer timestamps to calculate the clock offset and delay. In [14], the Glossy was proposed to reduce the packet loss rate and the synchronization time. It permits nodes to transmit identical packets concurrently and takes advantage of the capture effect to increase flooding efficiency. The Glossy is able to complete flood packets in milliseconds with an accuracy of less than 1 microsecond. To reduce the synchronization time, the Rapid-flooding Multiple one-way broadcast Time Synchronization (RMTS) proto-

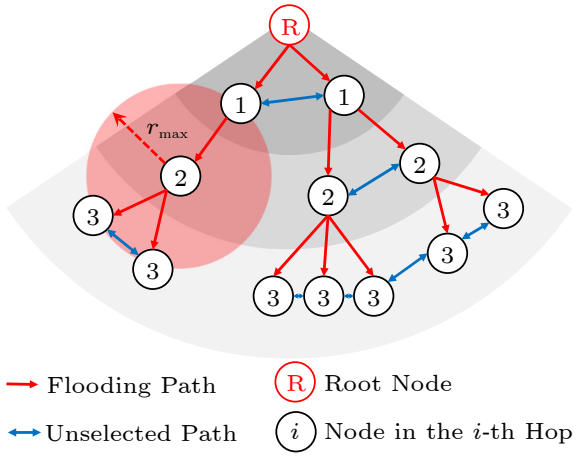


Fig.8. Ad-hoc structure formed by FTSP. $r_{\max}$ means the maximum communication distance of a node.

col[13] employs a multiple unidirectional broadcast model, and the receiver only receives the first arriving packet. In addition, RMTS uses maximum likelihood estimation to estimate clock skew and offset in order to minimize errors; in order to reduce hop-by-hop errors, RMTS mandates that all nodes share clock parameters. Experiments indicate that even in flooded networks with more than 24 hops, RMTS can achieve precise synchronization by the third synchronization period.

In the preceding flooding protocols, the sensor node must perform linear regression or maximum likelihood estimation, resulting in a high level of computational complexity. To reduce the computational complexity at sensor nodes, the BATS (Beaconless Asymmetric energy-efficient Time Synchronization) scheme was proposed in [20]. The BATS scheme presupposes that the performance of the flooding network is asymmetric, i.e., the root node has powerful computing resources and a sufficient energy supply, whereas the sensor nodes have limited memory and energy. The BATS scheme only requires sensor nodes to broadcast their timestamp information unidirectionally to the root node; the root node then estimates the clock skew and offset of each sensor node. This scheme can be adapted to multi-hop flooding networks. Through this asymmetric utilization of resources, it is possible to reduce sensor node energy consumption by up to 95% compared with FTSP and achieve microsecond-level precision. A BATS-like scheme, EE-ASCFR, was proposed in [44]. According to the asymmetry of node resources, each sensor node listens to the time information broadcast from the root node to estimate the clock skew and uses a bidirectional message exchange process opposite to Fig.6 to estimate the clock offset and delay, while the head node handles the complex calculation process. Sub-microsecond precision can eventually be achieved.

3.1.3 Receiver-Only Synchronization (ROS)

ROS means that the receiver only needs to listen for messages from two-way communication between specific nodes to achieve time synchronization as opposed to broadcasting its own time information. There is no node performing the function of the sender. Let us consider the scenario depicted in Fig.9, in which nodes A and P are super nodes with sufficient computing and communication resources and can provide time information to all nodes within the communication range. Between node A and node P,

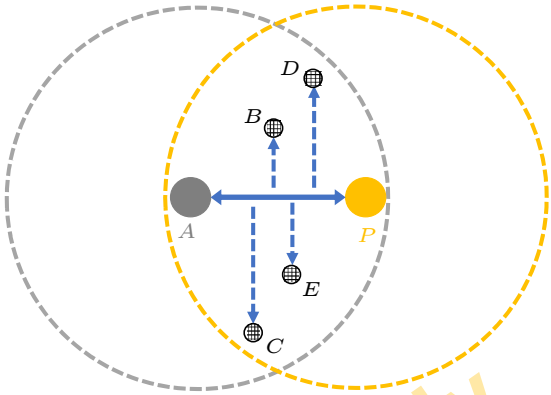


Fig.9. Scenario where the ROS model is applicable.

there are frequent two-way message exchanges and pair-wise synchronization. For all sensor nodes (e.g., node B) within the public communication range of nodes A and P, it is sufficient for them to monitor the message interaction between nodes A and P in order to synchronize their own clocks with those of the two[45].

The Pairwise Broadcast Synchronization (PBS) protocol was proposed in [45], employing the ROS mode, which can drastically decrease the cost of all sensor nodes within the communication range. To achieve clock synchronization over a greater distance, it is sufficient to extend the protocol to construct a spanning tree comprised of super nodes, and the other nodes only need to monitor the two-way message exchanges between a pair of nearby parent-child super nodes. The spanning tree's super nodes can be synchronized in the RRS or SRS mode. The STETS (Spanning Tree-based Energy-efficient Time Synchronization) scheme[46], for instance, is a combination of the SRS mode and the ROS mode. After selecting some backbone nodes (BNs), a spanning tree is constructed, using the SRS mode, and other passive nodes (PNs) receive two-way clock synchronization information between nearby paired BNs using the ROS mode in order to update their own clocks. However, because a PN does not broadcast any information, some remote PNs may be unable to locate paired BNs, and these PNs become isolated.

3.1.4 Security

During the construction of a spanning tree, some nodes may be overlooked[47]. In order to be able to add isolated nodes to the spanning tree without rebuilding it, the R-Sync scheme was proposed[47]. This scheme updates the network by using two types of timers: one for two-way message exchange between

the master and slave nodes, and the other for pulling isolated nodes to the synchronization network, which results in longer synchronization cycles.

Due to the direct communication method between nodes in the spanning tree based network, the security problem of an upper-layer node may affect a lower-layer node. There are inevitably illegal malicious nodes (MNs) in complex Industrial environments, which can affect the clock accuracy of the entire network by broadcasting false timestamps. Under the condition of unavoidable malicious nodes, the Secure Time Synchronization Protocol (STSP)[9] uses both the parent and grandparent nodes as reference nodes to determine whether the parent node is an MN. For the protocol to successfully detect the MN, the grandfather node must be an authorized node. In [48], it was proposed to protect data packets with timestamps using a pre-shared key between the parent and child nodes, thereby excluding the influence of MN in terms of message.

3.1.5 *Summary*

Table 1 summarizes the external clock source synchronization protocols and the centralized time synchronization protocols. The accuracy in the table means the size of time errors between the nodes' clocks and the reference clock after clock synchronization. A clock synchronization protocol with rapidity in the table means that the protocol clearly indicates that the size of the time slot has little impact on the clock synchronization after applying the protocol. Table 1 shows that reducing the frequency of communication between two nodes in centralized synchronization can effectively reduce the complexity of such methods.

3.2 Distributed Clock Synchronization

Distributed clock synchronization networks are designed to be decentralized and have a loose topology, in contrast to centralized clock synchronization networks. In distributed clock synchronization, each node broadcasts time information to neighboring nodes within the communication range[33, 49], and through continuous iteration, all nodes' clocks converge to the same time according to predefined rules. Note that the converged clock does not have to be an absolute reference clock; it only needs to be the network-wide synchronized clock. Due to the necessity of repeating the clock synchronization process, distributed clock synchronization has a slower convergence speed and higher energy consumption than centralized clock synchronization[27]. Meanwhile, due to their loose topology, distributed networks can solve problems resulting from intermediate point failures and the addition of new nodes at a very low cost, and they are highly scalable. In addition, neighboring nodes' clocks influence each other, which means that distributed clock synchronization implicitly averages various network parameters (e.g., clock skew and clock offset), and the performance is almost independent of the number of nodes[33]. Due to the correlation of time information between nodes, distributed clock synchronization is susceptible to message manipulation attacks; therefore, its security is also a major concern.

Table 1. Summary of External Clock Source Synchronization Protocols and Centralized Clock Synchronization Protocols

		Protocol	Accuracy	Rapidity	Security	Robustness	Low Complexity
External clock source		Ethereum-based[5]	s		✓		
		AirSync[18]	sub-ms	✓			✓
		GPS-based[34]	ns				✓
Centralized	RRS	RBS[10]/CESP[38]	sub-μs				✓
	SRS	TPSN[29]	s				
		PTP[34]/SPTP[39]	sub-ms				
		RFC 4330[①]/EE-ASCFR[44]	sub-ms				✓
		BATS[20]	ms				✓
		FTSP[42]/RMTS[13]	sub-μs			✓	
		R-Sync[47]/STSP[9]	sub-μs		✓		
		Glossy[14]	μs			✓	
	ROS	STETS[46]	ms			✓	✓
		PBS[45]	sub-μs			✓	✓

[①]https://www.rfc-editor.org/rfc/rfc4330, Feb. 2023.

We will examine distributed clock synchronization protocols based on the level of node association.

3.2.1 Fully Distributed

In a fully distributed network, each node exchanges time information with its neighbors or with all the other nodes.

One of the objectives of clock synchronization in fully distributed networks is to minimize all clock skews, or the clock skew between any two network nodes. For this purpose, Djenouri[35] proposed the Relative Referenceless Receiver-Receiver Time Synchronization protocol (R^4Syn). In this protocol, each node receives information from other nodes, broadcasts its own information to other nodes, and then estimates the relative clock skew and offset based on the maximum likelihood. By making each node responsible for providing a reference clock, the single point of failure issue is resolved. The Emergency Broadcast Slot (EBS) scheme[3], which limits the communication time of nodes by setting a duty cycle phase, was proposed to further conserve energy. In order to prevent message collisions, the duty cycle phases of multiple nodes partially overlap. During the duty cycle phase, a node performs message transmission and clock synchronization; at other times, it enters the sleep state to conserve energy. Experiments indicate that when the duty cycle is maintained at 5%, the node is able to transmit 95% of valid messages.

In contrast to using the complex maximum likelihood estimation algorithm to obtain clock skew and offset for clock synchronization[35], the consensus method is a simpler and more efficient method for clock synchronization in distributed networks. Consensus signifies that after each node learns the clock value of its neighbors, it adjusts its own clock based on the predetermined consensus value[50]. Notably, the adjusted clock is not necessarily identical to the physical clock and is known as the "logical clock". By periodically invoking the consensus algorithm, the logical clocks of all nodes will converge on a single time source. Using average consensus as an example, each node periodically averages its own clock parameters and neighboring clock parameters so that the entire network is synchronized to the consensus reference clock at an exponential convergence speed[51]. Based on the principle of average consensus, the Gradient Time Synchronization Protocol (GTSP) was proposed in [31]. In addition, unlike minimizing all clock skew, the average consensus adopted by GTSP is equivalent to minimizing local clock skew, i.e., only the minimization of the skew among the local adjacent nodes is considered, whereas nodes with a greater distance are permitted to have a large skew. In order to accelerate the convergence speed of average consensus, it was proposed in [33] to use frame collisions (i.e., the superposition of transmission signals) to drive the network to achieve global synchronization and to allow each node to synchronize its clock by broadcasting public beacons, rather than executing the average consensus algorithm after collecting the information of all neighbors.

Similar to the average consensus, the maximum consensus algorithm is based on the maximum consensus theory[52], which aims to converge the logical clock parameters of each node to its maximum value, thereby achieving a faster convergence rate than the average consensus. Based on the maximum consensus algorithm, the Maximum Time Synchronization (MTS) protocol was proposed in [50]. This protocol is impractical as it does not account for communication delays. To be applicable to real-world scenarios, [50] further assumes that the communication delay is a positive random variable obeying a normal distribution, and proposes the Weighted Maximum Time Synchronization (WMTS) protocol, which uses smaller weights to weight the clock parameters of nodes with larger communication delays, thereby reducing the impact of these delays.

3.2.2 Clustered

The classic cluster model in the distributed network is shown in Fig.10.

Initially, each sensor node is categorized into distinct clusters based on the predefined rules. Some nodes that can belong to two clusters are referred to as cluster bridges or "gateway nodes" and they facili-

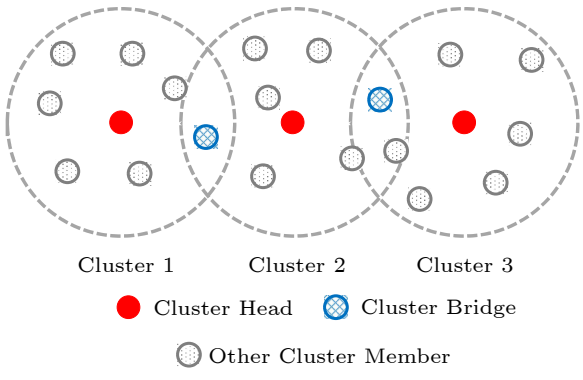


Fig.10. Classic cluster model.

tate the exchange of time-related information between clusters. Nodes within the same cluster vote for a cluster head, which can exchange time information with the cluster heads of other clusters via the cluster bridge and whose clock will serve as the cluster's reference clock. The other nodes in the cluster will either receive the one-way clock information broadcast by the cluster head or exchange two-way time information with the cluster head in order to synchronize their own clocks[53]. This method typically necessitates that the nodes in the network are homogeneous, e.g., WSN[25]. For synchronization, it is more efficient to divide the distributed network into multiple clusters. Typically, the cluster head fulfills the computational demands of the cluster, reducing the hardware requirements for sensor nodes. Notably, the cluster network is structurally similar to the multi-hop ROS network proposed in Subsection 3.1.3, but it does not require the cluster head and the cluster bridge to be the same node, resulting in a looser topology.

For clustered clock synchronization, [6] divides nodes into several clusters based on their locations and proposes the E-SATS (Efficient and Simple Algorithm for Time Synchronization) protocol. By designing a specific message exchange process based on the cluster model described above, the protocol reduces the number of data packets used in the synchronization process and achieves microsecond precision. Some studies introduce clustering techniques to cluster classification in an innovative manner[25, 27]. For example, in [25], k-means clustering was introduced to divide nodes, and a DCSP scheme with microsecond-level accuracy was proposed. Using a threshold-based k-means clustering algorithm, each node is automatically clustered into several clusters based on the varying rates of skew (VRS) of its oscillation frequency. After the cluster has been established, different clusters are synchronized at different frequencies, i.e., the clock synchronization in the cluster with the larger VRS occurs more frequently, thereby reducing the overall network's communication costs. The protocol also incorporates a VRS outlier detection mechanism and a second-order regression model detection mechanism for node verification in order to identify potentially malicious nodes.

3.2.3 Combination of Cluster and Consensus

Some studies combine consensus and cluster methods to accelerate convergence, reduce computational and energy costs, and preserve performance. The Cluster-Based Consensus Time Synchronization (CCTS) protocol was proposed in [27]. In CCTS, each cluster is first generated by clustering, then the cluster head completes inter-cluster clock synchronization by explicitly exchanging time information with cluster heads in other clusters via the gateway node, and finally the nodes in each cluster complete intra-cluster clock synchronization using the average consensus algorithm. CCTS can guarantee sub-microsecond accuracy. To further speed up the convergence, Cluster-Based Maximum Time Synchronization (CMTS)[52] uses the maximum consensus algorithm for intra-cluster clock synchronization, and other ideas are similar to [27] to further accelerate convergence.

3.2.4 Security

Cyber-physical attacks are common in fully distributed networks from a security perspective. The Node-Identification-Based Secure Time Synchronization (NiSTS) protocol[26] was proposed to filter the received time information using the timestamp correlation between different nodes and the uniqueness of clock skews, defending against the Sybil attack and the message manipulation attack in terms of information. The Attack-Tolerant Time-Synchronization Protocol (ATSP) proposed in [54] also employs comparable protection measures. Each sensor node determines the validity of the received time synchronization information by adaptively constructing and monitoring the configuration file of the synchronization behavior of neighboring nodes. In a distributed network based on consensus, the unique logical clocks can also be used for defense. The Secure Average-Consensus-Based Time Synchronization (SATS) protocol[4] combines the average consensus algorithm with an adjustment parameter check mechanism. The mechanism introduces the logical clock checking process to dynamically limit the impact of message manipulation attacks from malicious nodes, and then employs the hardware clock checking process to avoid receiving invalid data generated by such attacks. Similarly, [55] adds hardware clock and logic clock detection processes to the SMTS protocol to defend against possible message manipulation attacks in MTS.

One of the most important requirements for a clustered network is that the cluster heads cannot commit errors and can provide accurate reference time. In actual situations, however, it may be a problem that Byzantine nodes are selected as cluster

heads, causing network clock synchronization to fail. The C-sync scheme[53] suggests applying multiple cluster bridges between two clusters, where one cluster bridge node is used to support message exchange between clusters and the remaining bridges are used as supervisory nodes to monitor the accuracy of the time information broadcast by the cluster head. C-sync can guarantee sub-microsecond synchronization accuracy even in the presence of Byzantine nodes using this method.

3.2.5 *Summary*

The protocols for distributed time synchronization are summarized in Table 2. These protocols distribute errors to all nodes, thereby effectively defending against attacks from malicious nodes.

4 Applications

In this section, we will analyze the time synchronization in several common communication applications widely used on the Industrial Internet.

4.1 PROFINET

PROFINET[56] is an Industrial Ethernet communication protocol based on IEC 61 158[57]. PROFINET is used for exchanging data between controllers and devices, and has become one of the most popular Industrial Ethernet solutions in manufacturing and automation environments. According to the real-time property, three communication levels are designed in PROFINET: non-real-time communication based on TCP/UDP and IP, real time (RT) communication and isochronous real time (IRT) communication. The first one is usually used in situations that do not require real-time performance, e.g., parameter setting. PROFINET RT requires the response time less than 10 ms, which is generally used for the communication between the controller and I/O equipments. PROFINET IRT requires the response time less than 1ms, which is suitable for occasions with high requirements for real-time property, e.g., motion control. The clock synchronization protocol, Precision Transparent Clock Protocol (PTCP)[58], used in PROFINET IO, is based on PTP and can provide millisecond-level precision. In PTCP, the sequence of time information exchanged between the master and slave nodes is the same as in PTP. Therefore, the same offset and delay estimation formulas are utilized. PTCP resides in the transport layer of the OSI seven-layer model, whereas PTP resides in the MAC layer of the link layer. Thus, a portion of the processing mechanism in PTCP will be constrained. Furthermore, as the number of PTP devices in the Industrial environment increases, some studies attempt to synchronize the time of PTP-based devices via the PROFINET network and reduce the clock synchronization error between the two by designing compatibility measures[59, 60].

4.2 TSN

Industry leaders consider Time-Sensitive Networking (TSN) to be the most essential technology for future Industrial communications. It enables Industry 4.0 by integrating disparate enterprise sectors, such as Information Technology (IT) and Operation Technology (OT). TSN contains a series of standards targeting clock synchronization (IEEE 802.1AS[②]), queuing enhancements (IEEE 802.1Qav[③]), time-aware

Table 2. Summary of Distributed Clock Synchronization Protocols

	Protocol	Accuracy	Rapidity	Security	Robustness	Low Complexity
Fully distributed	EBS[3]	sub-ms	✓			
	GTSP[31]	sub-μs			✓	✓
	WMTS[50]	sub-μs				✓
	NiSTS[26]/ATSP[54]/SATS[4]/SMTS[55]	sub-μs		✓		
Clustered	C-sync[53]	sub-μs		✓		
	E-SATS[6]	μs				
	DCSP[25]	μs		✓		
Cluster & consensus	CCTS[27]/CMTS[52]	sub-μs				✓

[②]IEEE Standard for Local and Metropolitan Area Networks Timing and Synchronization for Time-Sensitive Applications. IEEE Std 802.1AS-2020, 2020, pp.1–421.

[③]IEEE Standard for Local and Metropolitan Area Networks Virtual Bridged Local Area Networks—Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams. IEEE Std 802.1Qav-2009, 2010, pp.1–72.

scheduling (IEEE 802.1Qbv[④]), etc. With the help of gate control defined in Qbv, Ethernet frames can be transmitted in a specific time slot. In typical TSN applications, such as the automotive industry and Industrial automation control, the maximum time difference between the master and slave nodes is frequently required to be between 0.1 μs and 1 μs[61, 62]. TSN adopts a PTP-based clock synchronization protocol named Generalized Precision Time Protocol (GPTP), which is defined as IEEE 802.1AS. GPTP and PTP have a similar clock synchronization process, but GPTP is more focused on two very important aspects of Industrial applications: supporting low-cost crystal clocks and supportability of all devices in the network to the protocol. Therefore, GPTP is simpler than PTP in some feature designs. For example, PTP supports OSI layers 2 and 3 (both MAC and network layers), while GPTP only supports OSI layer 2 (MAC layer)[⑤].

4.3 WirelessHART

WirelessHART[63] is the first open and interoperable standard for wireless communication in the process industry. It can provide a secure, low-power, and reliable wireless communication method to the industry, based on a centralized network management architecture, as well as multi-channel Time Division Multiple Access (TDMA), redundant routes, avoidance of spatial reuse of channels, channel blacklisting, and channel hopping[63]. It is widely used in environmental monitoring, Industrial measurement and process automation, etc. It defines the data link layer, network layer, transport layer, and application layer in the ISO protocol stack, and extends the physical layer based on IEEE802.15.4[63]. In a WirelessHART network, the delay in end-to-end data transmission is often caused by two aspects: channel contention (all nodes are set to transmit packets with higher priority in the timeslot) and transmission delay (the transmission of current packets and the transmission of packets with higher priority occur in the common node)[64], which is also one of the unavoidable problems in wireless communication technology. The WirelessHART typically utilizes the RBS-based algorithm for time synchronization and supports the centralized mesh network. Researchers have enhanced the algorithm and proposed ERBS (Energy-efficient RBS)[65]. ERBS takes into account the low power needs of WirelessHART so that nodes only exchange reference messages with non-adjacent nodes. In addition, some researchers attempt to integrate FTSP and TPSN into the WirelessHART network's time synchronization[66].

4.4 WIA-PA

WIA-PA is another wireless communication standard based on the IEEE 802.15.4 Standard and is used for process automation. It defines the data link sub-layer, network layer and application layer in the ISO protocol stack, and extends the physical layer and the MAC layer based on IEEE802.15.4[67]. Compared with WirelessHART, WIA-PA supports IEEE 802.15.4 superframe structure and packet aggregation mechanism for more efficient communication in the network[67]. In topology, WIA-PA network comprises two layers: the mesh network and the star network. Consequently, its clock synchronization consists of two steps as well. All the routing devices in a mesh network synchronize with the gateway. All device nodes in a star network synchronize their time with the routing devices. WIA-PA employs beacon frames and command frames to synchronize the clocks of two nodes. Some researchers proposed that FTSP can be modified to accommodate the two-layer network architecture of WIA-PA[68].

5 Discussion

Depending on the various requirements of the Industrial Internet for clock synchronization, the time synchronization protocols can be enhanced in a variety of ways. In this section, we will summarize potential directions for enhancing Industrial Internet clock synchronization protocols based on the proposed requirements in Section 1.

5.1 Accuracy

Without taking into account attacks, the precision of clock synchronization in real-world situations

[④]IEEE Standard for Local and Metropolitan Area Networks Bridges and Bridged Networks—Amendment 25: Enhancements for Scheduled Traffc. IEEE Std 802.1Qbv-2015, 2016, pp.1–57.

[⑤]IEEE Standard for Local and Metropolitan Area Networks Timing and Synchronization for Time-Sensitive Applications. IEEE Std 802.1AS-2020, 2020, pp.1–421.

is primarily dependent on the clock model and various delays. In terms of the clock model, simple protocols for synchronization directly compensate clock offsets through a single round of time information exchange[29]. By employing the first-order linear clock model and fitting multiple rounds of time information within a sliding window based on linear regression, e.g., linear regression in FTSP, maximum likelihood estimation[13, 35], clock skew and offset can be estimated with increased precision. Regarding delay estimation, there are numerous delays that impact the precision of clock synchronization, e.g., random access delay of MAC layer[22], asymmetric transmission delay between the master and slave nodes. Some researchers employ statistical knowledge in an effort to improve the accuracy of delay estimations. [35, 50], for instance, model the communication delay between nodes as a positive random variable with a Gaussian distribution. In addition, suppressing clock noise with filter technology (e.g., the Kalman filter) is advantageous for enhancing the precision of clock synchronization.

5.2 Rapidity

Distributed networks are centered on the speed, which is typically measured by the convergence rate. By enhancing the parameter update algorithm, such as by designing a consensus algorithm, the iteration cycle can be shortened and the convergence speed can be increased. In addition, the clock parameter sharing mechanism among nodes proposed in [13] is also used to achieve rapid convergence.

5.3 Security

In the Industrial Internet scenario, numerous types of attacks are inevitable. Clock synchronization protocols should therefore incorporate a defense mechanism. Possible enhancements to security include the followings.

- *Node Identification.* According to certain rules, the network system may identify malicious nodes and prevents attacks from these nodes. For example, some studies use the uniqueness of clock skew and the spatial correlation of RRS for node identification, and identify malicious nodes based on node identity[21], which is an effective method.
- *Information Identification.* Message manipulation attacks and Sybil attacks can be avoided by evaluating the veracity of information in received packages. Information-based defense methods include, but are not limited to, adding a clock parameter checking mechanism[4, 55], comparing the time information of parent nodes and grandparent nodes in spanning tree networks, etc.
- *Trusted Clock Source.* If the received time information originates from a trusted sender, there is no need to be concerned about its security, such as with a blockchain-based clock synchronization scheme. In [5], the free tamper-proof timestamp data provided by the public blockchain is used to synchronize the clock of nodes, and the nodes do not broadcast time information, thus preventing the influence of malicious nodes. In [49], a public verifiable ledger in a closed blockchain is used to record time information. Due to the consensus mechanism of the blockchain, an attacker cannot forge the wrong time.
- *Encryption Techniques.* Some studies introduce Encryption techniques to ensure the secure exchange of time information among nodes[7, 48]. For example, symmetric pre-shared keys are used in [48]. Paired cryptography is used in [69] to secure the time synchronization protocol in WSN. Due to the need for additional authentication information, encrypting typically requires additional time, storage, and communication costs[7, 25].

5.4 Robustness

Usually, robustness is concerned in centralized clock synchronization protocols. It can be enhanced by rationally designing the method of communication between nodes. For instance, SOR protocols utilize one-way communication, which can effectively resolve the issue where newly joined nodes cannot be synchronized.

5.5 Low Complexity

For general nodes, the clock synchronization protocols can be simplified by addressing the following issues.

- *Computing and Storage Resource Usage.* The nodes in the Industrial Internet network have the phenomenon of unbalanced computing and storage resources, e.g., the super nodes mentioned in the ROS model[45]. By concentrating on computing and storage needs in these super nodes, the consumption of computing and storage resources on sensor nodes can be drastically decreased. Moreover, with the development of cloud computing, some studies introduce edge

cloud architecture to establish a digital twin model of the Industrial network[16], concentrating on resource consumption on edge cloud servers.

- *Energy Consumption.* In addition to decreasing the consumption of computing and storage resources, it is advantageous for battery-powered nodes to communicate during the design node duty cycle phase and maintain the sleep state at other times[3, 31]. In addition, certain protocols support the time synchronization of nodes based on event triggering as opposed to periodic triggering, which can reduce unneeded computing, communication, and energy costs.

5.6 OSI Layer Optimization

In the majority of protocols introduced in Section 3, the MAC timestamp is a crucial piece of time information that necessitates a cross-layer design with specific hardware[11], which may not be available in practical applications. To support the Industrial Internet without the MAC timestamp, it is necessary to look for time information in other network protocol stack layers. For example, [11] proposes the Mesh Time Synchronization (MTP) protocol for IP-based wireless mesh networks, which is accurate to the millisecond level. A physical phenomenon based time synchronization scheme was proposed in [17], which utilizes concurrent passively observed physical phenomena to estimate the time information associated with those physical phenomena without using timestamps.

6 Related Work

Although several survey papers on time synchronization related concepts already exist in the literature[1, 70–74], none of these previous surveys have focused on the essential characteristics required by the Industrial Internet. A review from 2004[70] examines synchronization issues and compares various synchronization techniques. The references [71] and [72] examine the clock models in wireless sensor networks and present methods for synchronizing time in these networks. A work completed in 2016[73] classifies the methods for message passing based time synchronization in wireless sensor networks based on the network structure formation, the synchronization interval, and the synchronization message overhead. Synchronization for both computer networks and complex networks is discussed in [74]. Estimators play a crucial role in the analysis of various protocols. [1] focuses on clock synchronization over packet-switched networks, and the primary protocols used in computer networks (NTP, PTP, SyncE, and White Rabbit) are compared.

7 Conclusions

In this survey, we discussed clock synchronization for the Industrial Internet. We listed the requirements for clock synchronization in the Industrial Internet scenario and provided an overview of three clock synchronization methods. A physical crystal oscillator was used to model the clock, and a potential delay in the synchronization process was introduced. On the basis of the clock model and delay, a general mathematical method for synchronizing clocks was proposed. We presented a summary of clock synchronization protocols applicable to the Industrial Internet scenarios and classified and discussed each protocol using a multi-layer classification method. According to the reference clock source, the protocols were initially classified into external clock source synchronization and internal message exchange synchronization in the multi-layer classification. The internal message exchange synchronization method was subdivided into two types, centralized clock synchronization and distributed clock synchronization, according to the topological relationship between nodes. For centralized clock synchronization, according to the communication mode between parent nodes and child nodes in spanning trees, we summarized three modes: RRS, SRS, and ROS, and discussed protocols under each mode; for distributed clock synchronization, according to the degree of association among nodes in the distributed network, we summarized three modes: fully distributed, clustered, and consensus, and discuss ed protocols under each mode. Notably, security is an important aspect of the Industrial Internet, therefore we also focused on it. We provided examples of time synchronization protocol applications on the Industrial Internet. In order to further enhance time synchronization, we provided a summary of feasible improvement directions from the angles of precision, security, performance, and OSI layer optimization. Despite the fact that clock synchronization is a subject that has been extensively researched, for next-generation applications that require robust and secure clock synchronization in the Industrial Internet, there are still some issues that require our future attention:

• reasonable allocation and coordination of storage, computing, communication, and power resources in nodes with limited resources;

• guarantee of clock synchronization security in Industrial Internet;

• adaptive clock synchronization for complex Industrial environments.

References

[1] Lévesque M, Tipper D. A survey of clock synchronization over packet-switched networks. *IEEE Communications Surveys & Tutorials*, 2016, 18(4): 2926–2947. DOI: 10.1109/COMST.2016.2590438.

[2] Cintuglu M H, Mohammed O A, Akkaya K, Uluagac A S. A survey on smart grid cyber-physical system testbeds. *IEEE Communications Surveys & Tutorials*, 2017, 19(1): 446–464. DOI: 10.1109/COMST.2016.2627399.

[3] Yadav P, McCann J A, Pereira T. Self-synchronization in duty-cycled Internet of Things (IoT) applications. *IEEE Internet of Things Journal*, 2017, 4(6): 2058–2069. DOI: 10.1109/JIOT.2017.2757138.

[4] He J P, Cheng P, Shi L, Chen J M. SATS: Secure average-consensus-based time synchronization in wireless sensor networks. *IEEE Trans. Signal Processing*, 2013, 61(24): 6387–6400. DOI: 10.1109/TSP.2013.2286102.

[5] Regnath E, Shivaraman N, Shreejith S, Easwaran A, Steinhorst S. Blockchain, what time is it? Trustless date-time synchronization for IoT. In *Proc. the 2020 International Conference on Omni-layer Intelligent Systems*, Aug. 31–Sept. 2, 2020. DOI: 10.1109/COINS49042.2020.9191420.

[6] Chalapathi G S S, Chamola V, Guranarayanan S *et al.* E-SATS: An efficient and simple time synchronization protocol for cluster-based wireless sensor networks. *IEEE Sensors Journal*, 2019, 19(21): 10144–10156. DOI: 10.1109/JSEN.2019.2922366.

[7] Richards D, Abdelgawad A, Yelamarthi K. How does encryption influence timing in IoT? In *Proc. the 2018 IEEE Global Conference on Internet of Things*, Dec. 2018. DOI: 10.1109/GCIoT.2018.8620133.

[8] Zhang K, Liang X H, Lu R X, Shen X M. Sybil attacks and their defenses in the Internet of Things. *IEEE Internet of Things Journal*, 2014, 1(5): 372–383. DOI: 10.1109/JIOT.2014.2344013.

[9] Qiu T, Liu X Z, Han M, Ning H S, Wu D O. A secure time synchronization protocol against fake timestamps for large-scale Internet of Things. *IEEE Internet of Things Journal*, 2017, 4(6): 1879–1889. DOI: 10.1109/JIOT.2017.2714904.

[10] Elson J, Girod L, Estrin D. Fine-grained network time synchronization using reference broadcasts. In *Proc. the 5th Symposium on Operating Systems Design and Implementation*, Dec. 2002, pp.147–163.

[11] Beke T, Dijk E, Ozcelebi T, Verhoeven R. Time synchronization in IoT mesh networks. In *Proc. the 2020 International Symposium on Networks, Computers and Communications*, Oct. 2020. DOI: 10.1109/ISNCC49221.2020.9297296.

[12] Mani S K, Durairajan R, Barford P, Sommers J. An architecture for IoT clock synchronization. In *Proc. the 8th International Conference on the Internet of Things*, Oct. 2018, p.17. DOI: 10.1145/3277593.3277606.

[13] Shi F R, Tuo X G, Yang S X, Lu J, Li H L. Rapid-flooding time synchronization for large-scale wireless sensor networks. *IEEE Trans. Industrial Informatics*, 2020, 16(3): 1581–1590. DOI: 10.1109/TII.2019.2927292.

[14] Ferrari F, Zimmerling M, Thiele L, Saukh O. Efficient network flooding and time synchronization with glossy. In *Proc. the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, Apr. 2011, pp.73–84.

[15] Huan X T, Kim K S. Per-hop delay compensation in time synchronization for multi-hop wireless sensor networks based on packet-relaying gateways. *IEEE Communications Letters*, 2020, 24(10): 2300–2304. DOI: 10.1109/LCOMM.2020.3002705.

[16] Jia P Y, Wang X B, Shen X M. Digital-twin-enabled intelligent distributed clock synchronization in industrial IoT systems. *IEEE Internet of Things Journal*, 2021, 8(6): 4548–4559. DOI: 10.1109/JIOT.2020.3029131.

[17] Jia P Y, Wang X B, Shen X M. Passive network synchronization based on concurrent observations in industrial IoT systems. *IEEE Internet of Things Journal*, 2021, 8(18): 14028–14038. DOI: 10.1109/JIOT.2021.3070242.

[18] Zhu S P, Zheng X L, Liu L, Ma H D. AirSync: Time synchronization for large-scale IoT networks using aircraft signals. In *Proc. the 17th Annual IEEE International Conference on Sensing, Communication, and Networking*, Jun. 2020. DOI: 10.1109/SECON48991.2020.9158433.

[19] Nishi H, Song E Y, Nakamura Y, Lee K B, Liu Y C, Tsang K F. Time synchronization of IEEE P1451.0 and P1451.1.6 standard-based sensor networks. In *Proc. the 47th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2021. DOI: 10.1109/IECON48115.2021.9589904.

[20] Huan X T, Kim K S, Lee S, Lim E G, Marshall A. A beaconless asymmetric energy-efficient time synchronization scheme for resource-constrained multi-hop wireless sensor networks. *IEEE Trans. Communications*, 2020, 68(3): 1716–1730. DOI: 10.1109/TCOMM.2019.2960344.

[21] Huan X T, Kim K S, Zhang J Q. NISA: Node identification and spoofing attack detection based on clock features and radio information for wireless sensor networks. *IEEE Trans. Communications*, 2021, 69(7): 4691–4703. DOI: 10.1109/TCOMM.2021.3071448.

[22] Bhandari S, Wang X B. Prioritized clock synchronization for event critical applications in wireless IoT networks. *IEEE Sensors Journal*, 2019, 19(16): 7120–7128. DOI: 10.1109/JSEN.2019.2912938.

[23] Schmid T, Shea R, Charbiwala Z, Friedman J, Srivastava M B, Cho Y H. On the interaction of clocks, power, and synchronization in duty-cycled embedded sensor

nodes. *ACM Trans. Sensor Networks*, 2010, 7(3): Article No. 24. DOI: 10.1145/1807048.1807053.

[24] Yang S J, Xu C Q, Guan J F, Zhang T. Event-based diffusion Kalman filter strategy for clock synchronization in WSNs. In *Proc. the 2018 International Conference on Networking and Network Applications*, Oct. 2018, pp.270–276. DOI: 10.1109/NANA.2018.8648770.

[25] Jia PY, Wang XB, Zheng K. Distributed clock synchronization based on intelligent clustering in local area industrial IoT systems. *IEEE Trans. Industrial Informatics*, 2020, 16(6): 3697–3707. DOI: 10.1109/TII.2019.2937331.

[26] Wang Z W, Zeng P, Kong L H, Li D, Jin X. Node-identification-based secure time synchronization in industrial wireless sensor networks. *Sensors*, 2018, 18(8): 2718. DOI: 10.3390/s18082718.

[27] Wu J, Zhang L Y, Bai Y, Sun Y S. Cluster-based consensus time synchronization for wireless sensor networks. *IEEE Sensors Journal*, 2015, 15(3): 1404–1413. DOI: 10.1109/JSEN.2014.2363471.

[28] Kadambar S, Chavva A K R. Low complexity ML synchronization for 3GPP NB-IoT. In *Proc. the 2018 International Conference on Signal Processing and Communications*, Jul. 2018, pp.307–311. DOI: 10.1109/SPCOM.2018.8724439.

[29] Dian F J, Yousefi A, Somaratne K. A study in accuracy of time synchronization of BLE devices using connection-based event. In *Proc. the 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference*, Oct. 2017, pp.595–601. DOI: 10.1109/IEMCON.2017.8117156.

[30] Gore R N, Lisova E, Åkerberg J, Björkman M. CoSiNeT: A lightweight clock synchronization algorithm for industrial IoT. In *Proc. the 4th IEEE International Conference on Industrial Cyber-Physical Systems*, May 2021, pp.92–97. DOI: 10.1109/ICPS49255.2021.9468174.

[31] Sommer P, Wattenhofer R. Gradient clock synchronization in wireless sensor networks. In *Proc. the 2009 International Conference on Information Processing in Sensor Networks*, Apr. 2009, pp.37–48.

[32] Li Y, Chen S, Lin F J. A coarse timing synchronization method of low SNR OFDM systems for IoT. In *Proc. the 2018 IEEE International Conference on Integrated Circuits, Technologies and Applications*, Nov. 2018, pp.166–167. DOI: 10.1109/CICTA.2018.8705961.

[33] Alvarez M A, Spagnolini U. Collision vs non-collision distributed time synchronization for dense IoT deployments. In *Proc. the 2017 IEEE International Conference on Communications*, May 2017. DOI: 10.1109/ICC.2017.7997469.

[34] Idrees Z, Granados J, Sun Y, Latif S, Gong L, Zou Z, Zheng L R. IEEE 1588 for clock synchronization in industrial IoT and related applications: A review on contributing technologies, protocols and enhancement methodologies. *IEEE Access*, 2020, 8: 155660–155678. DOI: 10.1109/ACCESS.2020.3013669.

[35] Djenouri D. *R⁴Syn*: Relative referenceless receiver/receiver time synchronization in wireless sensor networks. *IEEE Signal Processing Letters*, 2012, 19(4): 175–178. DOI: 10.1109/LSP.2012.2185491.

[36] Cheng S Y, Cai Z P, Li J Z, Gao H. Extracting kernel dataset from big sensory data in wireless sensor networks. *IEEE Trans. Knowledge and Data Engineering*, 2017, 29(4): 813–827. DOI: 10.1109/TKDE.2016.2645212.

[37] Su W, Akyildiz I F. Time-diffusion synchronization protocol for wireless sensor networks. *IEEE/ACM Trans. Networking*, 2005, 13(2): 384–397. DOI: 10.1109/TNET.2004.842228.

[38] Gong F Y, Sichitiu M L. CESP: A low-power high-accuracy time synchronization protocol. *IEEE Trans. Vehicular Technology*, 2016, 65(4): 2387–2396. DOI: 10.1109/TVT.2015.2417810.

[39] Resner D, Fröhlich A A, Wanner L F. Speculative precision time protocol: Submicrosecond clock synchronization for the IoT. In *Proc. the 21st International Conference on Emerging Technologies and Factory Automation*, Sept. 2016. DOI: 10.1109/ETFA.2016.7733533.

[40] Raju N, Hasan K F. A feasibility study on SNTP and SPoT protocols on time synchronization in Internet of Things. arXiv: 2010.09219, 2020. https://arxiv.org/abs/2010.09219, Dec. 2022.

[41] Bansal M, Gupta A. Out-degree based clock synchronization in wireless networks using precision time protocol. In *Proc. the 2018 IEEE International Conference on Advanced Networks and Telecommunications Systems*, Dec. 2018. DOI: 10.1109/ANTS.2018.8710042.

[42] Maróti M, Kusy B, Simon S, Lédeczi Œ. The flooding time synchronization protocol. In *Proc. the 2nd International Conference on Embedded Networked Sensor Systems*, Nov. 2004, pp.39–49. DOI: 10.1145/1031495.1031501.

[43] Sheu J P, Hu W K, Lin J C. Ratio-based time synchronization protocol in wireless sensor networks. *Telecommunication Systems*, 2008, 39(1): 25–35. DOI: 10.1007/s11235-008-9081-5.

[44] Kim K S, Lee S, Lim E G. Energy-efficient time synchronization based on asynchronous source clock frequency recovery and reverse two-way message exchanges in wireless sensor networks. *IEEE Trans. Communications*, 2017, 65(1): 347–359. DOI: 10.1109/TCOMM.2016.2626281.

[45] Noh K L, Serpedin E, Qaraqe K. A new approach for time synchronization in wireless sensor networks: Pairwise broadcast synchronization. *IEEE Trans. Wireless Communications*, 2008, 7(9): 3318–3322. DOI: 10.1109/TWC.2008.070343.

[46] Qiu T, Chi L, Guo W *et al.* STETS: A novel energy-efficient time synchronization scheme based on embedded networking devices. *Microprocessors and Microsystems*, 2015, 39(8): 1285–1295. DOI: 10.1016/j.micpro.2015.07.006.

[47] Qiu T, Zhang Y S, Qiao D J, Zhang X Y, Wymore M L, Sangaiah A K. A robust time synchronization scheme for Industrial Internet of Things. *IEEE Trans. Industrial Informatics*, 2018, 14(8): 3570–3580. DOI: 10.1109/TII.2017.2738842.

[48] Navas R E, Toutain L. LATe: A lightweight authenticated time synchronization protocol for IoT. In *Proc. the*

2018 Global Internet of Things Summit, Jun. 2018. DOI: 10.1109/GIOTS.2018.8534565.

[49] Fan K, Wang S Y, Ren Y H, Yang K, Yan Z, Li H, Yang Y T. Blockchain-based secure time protection scheme in IoT. *IEEE Internet of Things Journal*, 2019, 6(3): 4671–4679. DOI: 10.1109/JIOT.2018.2874222.

[50] He J P, Cheng P, Shi L, Chen J M, Sun Y X. Time synchronization in WSNs: A maximum-value-based consensus approach. *IEEE Trans. Automatic Control*, 2014, 59(3): 660–675. DOI: 10.1109/TAC.2013.2286893.

[51] Schenato L, Fiorentin F. Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks. *Automatica*, 2011, 47(9): 1878–1886. DOI: 10.1016/j.automatica.2011.06.012.

[52] Wang Z W, Zeng P, Zhou M T, Li D, Wang J T. Cluster-based maximum consensus time synchronization for industrial wireless sensor networks. *Sensors*, 2017, 17(1): 141. DOI: 10.3390/s17010141.

[53] Shivaraman N, Schuster P, Ramanathan S, Easwaran A, Steinhorst S. C-Sync: The resilient time synchronization protocol. In *Proc. the 19th ACM/IEEE International Conference on Information Processing in Sensor Networks* (*poster*), Apr. 2020, pp.333–334. DOI: 10.1109/IPSN48710.2020.00-20.

[54] Hu X, Park T, Shin K G. Attack-tolerant time-synchronization in wireless sensor networks. In *Proc. the 27th Conference on Computer Communications*, Apr. 2008, pp.41–45. DOI: 10.1109/INFOCOM.2008.17.

[55] He J P, Chen J M, Cheng P, Cao X H. Secure time synchronization in wireless sensor networks: A maximum consensus-based approach. *IEEE Trans. Parallel and Distributed Systems*, 2014, 25(4): 1055–1065. DOI: 10.1109/TPDS.2013.150.

[56] Feld J. PROFINET-scalable factory communication for all applications. In *Proc. the 2004 IEEE International Workshop on Factory Communication Systems*, Sept. 2004, pp.33–38. DOI: 10.1109/WFCS.2004.1377673.

[57] PROFIBUS Nutzerorganisation eV. Application layer protocol for decentralized periphery and distributed automation; specification for PROFINET; Version 2.1, June 2006. *Order*, (2. 722). https://webstore.iec.ch/publication/59893, Feb. 2023.

[58] Fontanelli D, Macii D, Rinaldi S, Ferrari P, Flammini A. Performance analysis of a clock state estimator for PROFINET IO IRT synchronization. In *Proc. the 2013 IEEE International Instrumentation and Measurement Technology Conference*, May 2013, pp.1828–1833. DOI: 10.1109/I2MTC.2013.6555730.

[59] Ferrari P, Flammini A, Marioli D, Rinaldi S, Sisinni E, Taroni A, Venturini F. Clock synchronization of PTP-based devices through PROFINET IO networks. In *Proc. the 2008 IEEE International Conference on Emerging Technologies and Factory Automation*, Sept. 2008, pp.496–499. DOI: 10.1109/ETFA.2008.4638445.

[60] Ferrari P, Flammini A, Rinaldi S, Sisinni E. On the seamless interconnection of IEEE1588-based devices using a PROFINET IO infrastructure. *IEEE Trans. Industrial Informatics*, 2010, 6(3): 381–392. DOI: 10.1109/TII.2010.2051954.

[61] Val I, Seijo ó, Torrego R, Astarloa A. IEEE 802.1AS clock synchronization performance evaluation of an integrated wired-wireless TSN architecture. *IEEE Trans. Industrial Informatics*, 2022, 18(5): 2986–2999. DOI: 10.1109/TII.2021.3106568.

[62] Zhao Y, Yang Z, He X W, Wu J H, Cao H, Dong L, Dang F, Liu Y H. E-TSN: Enabling event-triggered critical traffic in time-sensitive networking for industrial applications. In *Proc. the 42nd International Conference on Distributed Computing Systems*, Jul. 2022, pp.691–701. DOI: 10.1109/ICDCS54860.2022.00072.

[63] Chen D J, Nixon M, Mok A. WirelessHART™: Real-Time Mesh Network for Industrial Automation. Springer, 2010.

[64] Saifullah A, Xu Y, Lu C Y, Chen Y X. End-to-end delay analysis for fixed priority scheduling in WirelessHART networks. In *Proc. the 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, Apr. 2011, pp.13–22. DOI: 10.1109/RTAS.2011.10.

[65] Wang Y J, Qian Z H, Wang G Q, Zhang X. Research on energy-efficient time synchronization algorithm for wireless sensor networks. *Journal of Electronics & Information Technology*, 2012, 34(9): 2174–2179. DOI: 10.3724/SP.J.1146.2012.00236.

[66] Huang T, Huang S Z. Low power WirelessHART network time synchronization protocol. Chinese Journal of Electron Devices, 2014, 37(1): 85–88. DOI: 10.3969/j.issn.1005-9490.2014.01.021. (in Chinese)

[67] Liang W, Zhang X L, Xiao Y, Wang F Q, Zeng P, Yu H B. Survey and experiments of WIA-PA specification of industrial wireless network. *Wireless Communications and Mobile Computing*, 2011, 11(8): 1197–1212. DOI: 10.1002/wcm.976.

[68] He N, Liu F. Research on time synchronization of WIA-PA industrial wireless networks. In *Proc. the 2009 International Conference on Computational Intelligence and Software Engineering*, Dec. 2009. DOI: 10.1109/CISE.2009.5363213.

[69] Rahman M, El-Khatib K. Secure time synchronization for wireless sensor networks based on bilinear pairing functions. *IEEE Trans. Parallel and Distributed Systems*, 2010. DOI: 10.1109/TPDS.2010.94.

[70] Sivrikaya F, Yener B. Time synchronization in sensor networks: A survey. *IEEE Network*, 2004, 18(4): 45–50. DOI: 10.1109/MNET.2004.1316761.

[71] Faizulkhakov Y R. Time synchronization methods for wireless sensor networks: A survey. *Programming and Computer Software*, 2007, 33(4): 214–226. DOI: 10.1134/S0361768807040044.

[72] Lasassmeh S M, Conrad J M. Time synchronization in wireless sensor networks: A survey. In *Proc. the 2010 IEEE SoutheastCon*, Mar. 2010, pp.242–245. DOI: 10.1109/SECON.2010.5453878.

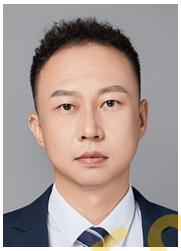
[73] Sarvghadi M A, Wan T C. Message passing based time synchronization in wireless sensor networks: A survey. *In-*

ternational Journal of Distributed Sensor Networks, 2016, 12(5): 1280904. DOI: 10.1155/2016/1280904.

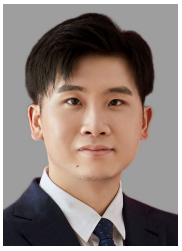
[74] Puttnies H, Danielis P, Sharif A R, Timmermann D. Estimators for time synchronization—Survey, analysis, and outlook. *IoT*, 2020, 1(2): 398–435. DOI: 10.3390/iot1020023.

Fan Dang received his B.E. and Ph.D. degrees in software engineering from Tsinghua University, Beijing, in 2013 and 2018, respectively. He is a research assistant professor in the Global Innovation Exchange, Tsinghua University, Beijing. He is a member of CCF, ACM, and IEEE. His research interests include the Industrial Internet, edge computing, and mobile security

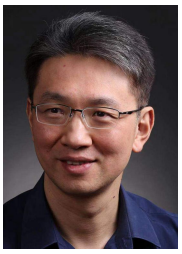
Xi-Kai Sun is an undergraduate student in the Department of Automation, Tsinghua University, Beijing. His research interests include the Industrial Internet and mobile security.



Ke-Bin Liu received his B.S. degree in computer science from Tongji University, Shanghai, in 2004. He received his M.S. and Ph.D. degrees in computer science from Shanghai Jiao Tong University, Shanghai, in 2007 and 2010, respectively. He is a research associate professor in the Global Innovation Exchange, Tsinghua University, Beijing. He is a member of CCF, ACM and IEEE. His research interests include Internet of Things, pervasive computing, network diagnosis, and artificial intelligence.



Yi-Fan Xu received his B.S. degree in software engineering from Tsinghua University, Beijing, in 2020. He is currently working toward his Ph.D. degree in the School of Software, Tsinghua University, Beijing. His research interests include the Industrial Internet.



Yun-Hao Liu received his B.E. degree from the Department of Automation, Tsinghua University, Beijing, in 1995. He received his M.A. degree from Beijing Foreign Studies University, Beijing, in 1997. He received his M.S. and Ph.D. degrees in computer science and engineering from Michigan State University, East Lansing, in 2003 and 2004, respectively. He is a professor in the Department of Automation and the dean of the Global Innovation Exchange, Tsinghua University, Beijing. He is a fellow of CCF, ACM and IEEE. He is the Editor-in-Chief of ACM Transactions on Sensor Networks and Communications of the CCF. His research interests include Internet of Things, wireless sensor networks, indoor localization, the Industrial Internet, and cloud computing.