# A Survey of Clock Synchronization Over Packet-Switched Networks

Martin Lévesque and David Tipper

***Abstract*—Clock synchronization is a prerequisite for the realization of emerging applications in various domains such as industrial automation and the intelligent power grid. This paper surveys the standardized protocols and technologies for providing synchronization of devices connected by packet-switched networks. A review of synchronization impairments and the state-of-the-art mechanisms to improve the synchronization accuracy is then presented. Providing microsecond to sub-microsecond synchronization accuracy under the presence of asymmetric delays in a cost-effective manner is a challenging problem, and still an open issue in many application scenarios. Further, security is of significant importance for systems where timing is critical. The security threats and solutions to protect exchanged synchronization messages are also discussed.**

***Index Terms*—Time dissemination, packet switched networks, synchronization.**

## I. Introduction

TIGHT time synchronization is a key requirement in several packet based communication networks and real-time networked application domains, such as, automated industrial systems, and smart power grid systems. For example, communication networks such as Long Term Evolution-Advanced (LTE-A) based cellular networks require base stations and backhaul equipment to maintain time synchronization in order to provide several new features like synchronized transmissions over frequencies from adjacent base stations and interference coordination. An example of an emerging networked application is smart power grid systems which are characterized by a two-way flow of energy and end-to-end communications. The communications are largely machine-to-machine (M2M) in nature [1] and need to share synchronized information in order to improve efficiency and reliability of power delivery.

The National Institute of Standards and Technology (NIST) has recently launched a Public Working Group (PWG) on cyber-physical systems (CPSs) [2].

CPSs are systems which are co-engineered to interact with physical, computational, and communications components. CPSs will be deployed in several functional domains, including in smart infrastructures (e.g., power grid, water), smart buildings, smart healthcare, and smart transportation to name a few domains. Time synchronization will play a critical role for the realization of CPSs as reported by a subgroup of the NIST CPS PWG in a recent technical report [3]. In [3], the authors specify that significant research is required in an integrated fashion for the realization of emerging new time-aware applications, computers, and communications systems (TAACCS), decomposed into the following principle areas of research: (1) time and frequency transfer, (2) clock and oscillator design, (3) use of synchronized timing in communications networks, and (4) hardware and software architecture, applications and systems design. In this paper, we focus on the time and frequency transfer area considering the time synchronization of devices connected by packet switched networks. Note, that time synchronization includes both aligning clock frequencies, phase and time values. Hence, it is broader than syntonization where one adjusts device clocks to operate on the same frequency.

A general solution to providing time synchronization among networked devices requiring time alignment is incorporating an accurate source of time in each device (e.g., atomic clock). However, equipping each device with this technology is expensive and not practical in many scenarios (e.g., sensors in smart power grids [4]). Given the wide deployment of computing devices using simple crystal clocks, there has been considerable efforts on how to synchronize them.

Typically, to synchronize a given crystal clock, it must rely on a reference clock having a reliable source of time, such as an atomic clock or a Global Navigation Satellite System (GNSS) receiver, and then the reference time is transferred over a communications network experiencing variable hardware, network, and software delays.

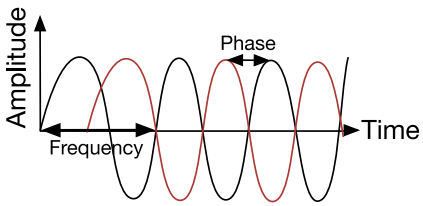
In this paper, we summarize and compare the standards providing synchronization support over packet-switched networks. Further, this survey provides an in-depth review of the recently proposed mechanisms to improve synchronization accuracy and we overview the major applications needing synchronization and their requirements.

The remainder of the paper is structured as follows. Section II first introduces the general clock synchronization problem and key notions. In Section III, we describe the main applications requiring synchronized clocks, as well as their synchronization requirements. In Section IV, we overview standardized and widely used synchronization protocols, which provide the best practices from both the industry and academia. We then outline in Section VI the specific

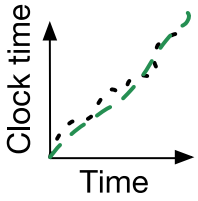
Manuscript received February 26, 2015; revised September 7, 2015, February 14, 2016, and May 24, 2016; accepted July 1, 2016. Date of publication July 12, 2016; date of current version November 18, 2016. This work was supported by NSERC Post-Doctoral Fellowship under Grant 453711-2014. *(Corresponding author: Martin Lévesque.)*

The authors are with the School of Information Science, University of Pittsburgh, Pittsburgh, PA 15260 USA (e-mail: levesque@pitt.edu).

Digital Object Identifier 10.1109/COMST.2016.2590438




(a) Time, frequency, and phase components of two signals.




(b) Time synchronization. The dashed curves represent two clocks synchronizing in order to model the true time (x-axis).

Fig. 1. Time, frequency, and phase synchronization illustration.

synchronization impairments and cover the proposed mechanisms to improve the synchronization accuracy. Next, we review the security solutions to protect exchanged synchronization messages against malicious attackers. Section VIII concludes the paper.

## II. Key Synchronization Notions

### A. Time, Frequency, and Phase Synchronization

There exist three fundamental types of synchronization, namely, based on the time, frequency, and phase [5], [6]. Fig. 1a) depicts the time, frequency, and phase components, where in this example the phase differs. It is worth noting that the clock frequency accuracy is generally given in terms of parts-per million (ppm) or parts-per billion (ppb). A frequency error at given time $t$, $f_e(t) = f(t) - f_0$, corresponds to the difference between the measurement frequency $f(t)$ and nominal frequency $f_0$ [7]. For example, an accuracy of 5 ppm means that the average clock frequency can differ by 5 Hz every 1 MHz of its given value.

The synchronization types are characterized as follow, as described in ITU-T G.8260:

- *Time Synchronization:* Time synchronization refers to the distribution of an absolute time reference to a set of real-time clocks. The synchronized nodes share a common epoch and time-scale.
- *Frequency Synchronization:* In frequency synchronized systems, the significant instants occur at the same rate for all synchronized nodes.
- *Phase Synchronization:* In systems synchronized based on the phase, the timing signals occur at the same instant.

Thus in Fig. 1a), the two signals are not synchronized by phase, but are synchronized following the same frequency. Fig. 1b) illustrates two clocks (the dashed curves) synchronizing according to the time parameter. As we described in the previous section, different synchronization types are needed depending on the application. For example, in LTE TDD, where the frequencies are shared uplink and downlink, phase and frequency synchronization must be supported. In contrast, for applications requiring to timestamp events, time synchronization must be supported. In the following, we describe in detail time synchronization, which will be required in emerging time-aware applications.

### B. Time Synchronization Concept

Clock synchronization is a challenging task as it depends on a significant number of uncertain parameters. In this section, we introduce the general problem of clock synchronization of devices connected by packet-switched networks. The synchronization process refers to the overall mechanism of keeping multiple clocks at the same time, frequency, and/or phase. Syntonization, to not confuse with synchronization, corresponds to the correction of frequent clock drifts, which is often included in the synchronization process as we will describe shortly. More specifically, we consider the scenario where $K+1$ devices connected by a packet based communication network need to be time synchronized. Further, one device acts as a master maintaining a master clock time $t_m$, whereby the master clock is the time reference having a reliable source of time. The other $K$ devices are considered slaves to the master and the $k$th slave clock time $t_{s,k}$ relies on the master clock to correct its timing. The fundamental problem of clock synchronization is formalized as follows:

$$t_{s,k} = t_m + \theta(t_m), \tag{1}$$

where $\theta(t_m)$ corresponds to the offset time of a given slave clock $k \in \{1, \ldots, K\}$ relatively to a master clock time $t_m$. The offset time $\theta(t_m)$ varies according to several hardware, network, and software variable delays. On the top of these variable delays, physical impairments, including the mechanical loads and temperature fluctuations, affect each crystal oscillator and inherently the clock behavior. From a generic perspective, the offset time is given by [8]:

$$\theta(t_m) = \gamma_{s,k} \cdot t_m + \omega_{s,k}(t_m) + \theta^0_{s,k}, \tag{2}$$

where $\gamma_{s,k}$ refers to the deterministic skew, $\omega_{s,k}(t_m)$ is a variable deviation relatively to the deterministic skew, and $\theta^0_{s,k}$ corresponds to the difference between the slave and master clocks at the initial epoch. Note that the $\gamma_{s,k}$ term corresponds to the frequent clock drift estimated by the syntonization mechanism, which is included for instance in PTP as we will cover in the subsequent sections. The variable deviation is frequently included in the skew [9], [10], thus giving:

$$\theta(t_m) = \gamma_{s,k} \cdot t_m + \theta^0_{s,k}. \tag{3}$$

In order to synchronize slave clocks to a given master clock, the master clock time $t_m$ must be transmitted to the slave clocks over a communications network, as depicted in Fig. 2. As variable hardware, network, and software delays from the master node to the slave nodes occur, the general technique to

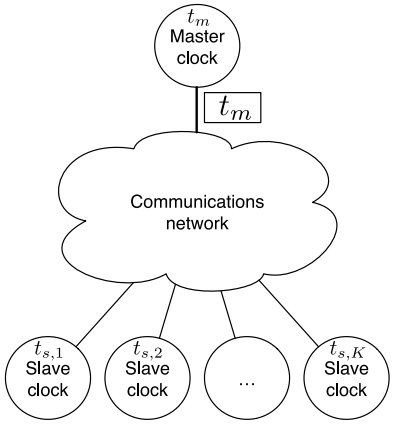



Fig. 2. Clock synchronization over packet-switched networks.

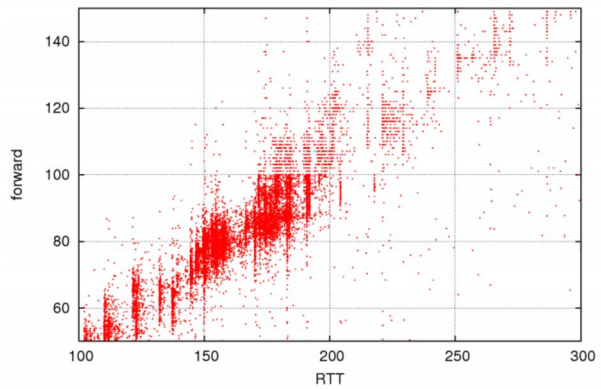



Fig. 3. Delay measurements of 25 commercial nodes in a large scale testbed (Planetlab) illustrating variable asymmetric delay ratios [11]. (y-axis (forward): one-way delay).

update $t_{s,k} \leftarrow t_m$ consists of transferring $t_m$ over the network and adjusting $t_{s,k}$ as follows:

$$t_{s,k} \leftarrow t_m + \bar{d} + \theta, \tag{4}$$

where $\bar{d}$ is the average delay between the master node to a given slave node $k$ and $\theta$ is the approximate offset. The problem of synchronizing then consists of finding the delay and offset. It is worth noting that each application domain requires a different synchronization performance based on its application requirements, as overviewed in Section III.

### C. Delay Asymmetry

The approximated delay $\bar{d}$ between the master and slave nodes is usually calculated based on the round-trip time (RTT) as follows for a given slave clock $k$:

$$\bar{d} = \frac{RTT}{2} = \frac{D_{m\to k} + D_{k\to m}}{2}, \tag{5}$$

where $D_{m\to k}$ and $D_{k\to m}$ represent the delay between the master and slave node, and conversely. Delay asymmetry occurs when $D_{m\to k} \neq D_{k\to m}$ and is due to many factors such as variable queuing/buffering delays, processing delays, route asymmetry, variations in link bandwidth, differences in physical cable lengths, and so forth. As an example of delay asymmetry, Pathak *et al.* [11] studied the delays of the Planetlab testbed consisting of 180 research/education and 25 commercial. Their measurement results show that the symmetry ratio ($\frac{D_{m\to k}}{RTT}$) was close to 0.5 for research/education connections, but significantly asymmetric for commercial connections (Fig. 3). For instance multiple RTTs were recorded as 150 ms, while the forward/one-way delays varied between 60-90 ms, thus corresponding to forward delay ratios of $\frac{60\ ms}{150\ ms} = 40\ \%$ to $\frac{90\ ms}{150\ ms} = 60\ \%$. Other studies in the literature show even larger asymmetry, such as the simulation experiments in [12].

### D. Environmental Factors

The environmental conditions in which a device operates can have an effect on the clock resulting in frequency drift

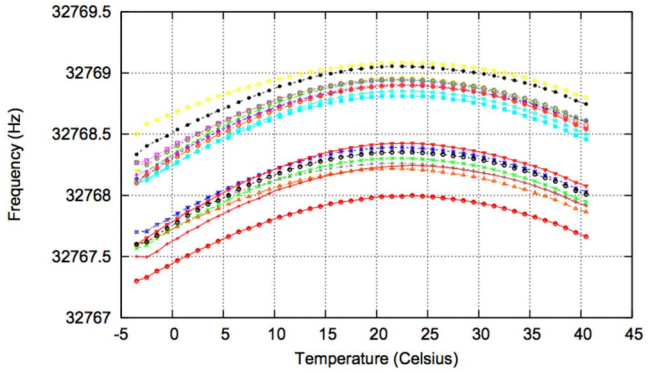



Fig. 4. Measured frequency of several crystal oscillators (nominal frequency corresponds to $f_0 = 32768.5$ Hz) [13].

for crystal based clocks and thus degrading the synchronization performance. Environmental factors that are known to effect clock accuracy are temperature, humidity, pressure, and vibration. Humidity, pressure, and vibration effects can be addressed by packaging of the oscillator, for example, hermetically sealing the crystal in vacuum or an inert gas like nitrogen eliminates most of the influence of both humidity and pressure. Temperature is more problematic, as an illustration, Fig. 4 from [13] depicts the measured frequency for several crystal oscillators found in sensor nodes for a wide range of temperatures. In this example (Fig. 4) at temperature 20 Celsius, an approximate maximum error of 0.6 Hz can be observed, corresponding to peak frequency of $\frac{0.6\ Hz}{32768.5\ Hz} \cdot 10^6 = 18.31$ ppm from the nominal frequency. Similarly, in [14], an extreme variation of 60 °C also resulted in significant synchronization errors (1-2 $\mu$s). Quick changes in temperature can have an immediate impact on clock stability, for example, a variation from -35 °C to 80 °C in 5 seconds was shown to influence the frequency of a crystal oscillator of 3 ppm according to the measurements in [15]. Temperature effects can be addressed in part by the type of oscillator and cut of the crystal, environmental controls and the use of synchronization protocols to adjust the clock.

TABLE I
ACCURACY AND COST FOR DIFFERENT OSCILLATORS [3]

| Oscillator Type | Accuracy | Cost |
|---|---|---|
| Quartz crystal | $10^{-5}$ to $10^{-4}$ | Inexpensive |
| Rubidium | $10^{-9}$ | \$800 USD |
| Cesium | $10^{-13}$ to $10^{-12}$ | \$50000 USD |

TABLE II
BASE STATION SYNCHRONIZATION REQUIREMENTS

| Technology | Synchronization Requirement |
|---|---|
| Location services of 3GPP [18] | 200 ns |
| LTE-A [5], [20] | 1.5-5 $\mu$s |
| LTE TDD [18] | 2.5 $\mu$s |
| CDMA2000 [21] | 3 $\mu$s |
| WiMAX [18] | 5 $\mu$s |
| UMTS W-CDMA MBSFN [21] | 12.8 $\mu$s |

### E. Reference Clocks

In order to provide clock synchronization, reference sources of time need to be installed such that one or multiple master clocks distribute the time, frequency, and/or phase to the slave clocks. The reference source of time can be based on atomic clocks or GNSS as described in the following.

*1) Atomic Clocks:* There exist multiple types of atomic clocks, which provide different accuracy. Table I lists the accuracy and cost for rubidium and cesium-based atomic clocks. We also included the conventional quartz crystal oscillator for comparison. There is a clear trade-off between the clock accuracy and cost, as we can see in Table I. While the quartz crystal oscillators can be widely deployed due to their very low cost, the accuracy they provide is limited. The rubidium and cesium based atomic clocks on the other hand provide higher accuracy, but are more costly.

*a) Rubidium-based atomic clocks:* The cheapest currently available atomic clocks use the resonance frequency of the rubidium atom (*Rb*) to control the frequency of a quartz oscillator.[1] After an initialization phase of few minutes, rubidium oscillators provide an accurate clock without needing adjustment.

*b) Cesium-based atomic clocks:* Similarly, cesium beam oscillators[2] use the resonance frequency of the cesium atom (*Cs*) in order to fine-tune a quartz oscillator.

*2) Global Navigation Satellite System (GNSS):* Global Navigation Satellite Systems (GNSSs) are used in many applications, including automobiles, air navigation, spacecraft, and so forth. Note that there are multiple GNSS systems, including the Global Positioning System (GPS), GLONASS (Russia), and Galileo (in deployment in Europe). In order to use a GNSS as a reference source of time a given device needs to be equipped with a GNSS receiver [16], [17]. GNSSs use satellites equipped with atomic clocks [17] to broadcast timing and navigational signals over wide geographic areas. Each satellite continuously transmits navigational signals which can be decoded by GNSS receivers. It is worth noting that to determine its position, a GNSS receiver must read the signals from multiple satellites, while only one satellite reading is required to decode time.

The major shortcoming of GNSS is its cost, since each device requires a GNSS receiver, which becomes economically prohibitive as the number of nodes grow [18]. Additional difficulties are the poor coverage of GNSS indoors as well in shadowed areas (e.g., canyons, dense urban areas, etc) and the power consumption in processing the GNSS signal. Even with theses drawbacks, GNSSs are useful to provide a highly accurate reference clock at the nanosecond resolution.

[1]http://tf.nist.gov/general/enc-re.htm#resonancefrequency
[2]http://tf.nist.gov/general/enc-c.htm#cesiumbeam

## III. APPLICATIONS AND REQUIREMENTS

In this section, we introduce several key applications requiring robust synchronization with their specific timing requirements. Even though each application does not have the same synchronization requirement, they share common problems in meeting the desired time synchronization performance.

### A. Mobile Cellular Networks

Synchronization is particularly important in mobile cellular networks as in many scenarios one needs to synchronize the base stations. It is also useful to monitor the network performance, for instance to measure network latency [19]. Table II lists examples of base station synchronization requirements for different wireless technologies. We observe that microsecond to nanosecond accuracy is required. It is worth noting that depending on the access radio, different types of synchronization are required [5]. In particular, when the access radio is based on LTE time-division duplex (TDD), where the upstream and downstream can use the same frequency,time, phase and frequency synchronization must be supported. The different synchronization types are introduced in the next section.

Traditionally, mobile cellular operators have supported synchronization with dedicated time division multiplexing (TDM) technologies as they are mature and familiar [22], [23]. Also, concerns were noted on the reliability and scalability of GNSS in terms of resilience against jamming and the significant cost for increasing number of devices [18]. TDM-based synchronization systems provide efficient timing performance, however they have a significant drawback of having low bandwidth efficiency. According to [24] and [25], it is envisioned that the traditional TDM synchronization architecture will be replaced by a packed-based synchronization scheme due to the cost efficiency and convergence of packet-based services provided by an all IP backhaul network. Further the definition of a synchronization profile is key for mobile backhaul needs [22], [26], such that different devices can inter-operate each other.

### B. Industrial Applications

Synchronization is also critical in industrial automation and data acquisition applications to improve precision, productivity, and quality [27]. Table III provides the synchronization requirements for several industrial applications, which require millisecond to microsecond synchronization performance.

TABLE III
SYNCHRONIZATION REQUIREMENTS OF INDUSTRIAL APPLICATIONS

| Application | Synchronization Requirement |
|---|---|
| Automation (PROFINET) [15] | 1 $\mu$s |
| Flight test [28] | 15 $\mu$s |
| Navy shipboard [29] | 500 $\mu$s |
| Manufacturing (Data acquisition) [30] | 1 ms |

TABLE IV
SMART GRID SYNCHRONIZATION REQUIREMENTS

| Functionality | Synchronization Requirement |
|---|---|
| WAMS [38], [39], [41], [42] | 1 $\mu$s |
| Fault detection [39] | 1 $\mu$s |
| IEC 61850 GOOSE [40] | 1 $\mu$s-1 ms |
| Protection functions [43], [45] | 1-100 $\mu$s |
| Fault recorder [40] | 100 $\mu$s |
| Differential protection [40] | 100 $\mu$s |
| Sequence of events [38], [41] | 1 ms |
| Substation monitoring [41] | 100 ms |

A typical function in industrial applications is data acquisition of synchronized time-stamped inputs from sensors [27]–[30]. For instance, events measured in different areas of an airplane need to be precisely time-stamped in order to reconstruct the context of correlated events [28]. Also, navy shipboards require precise time measurements in order to detect, locate, and identify moving targets [29].

### C. Smart Grid

The smart grid is an enhancement of the legacy power grid consisting of an intelligent two-way flow of energy and information to support emerging distributed renewable energy resources, demand response, and to improve the grid efficiency and reliability [31]–[34]. The smart grid communications architecture can be decomposed into three main subnetworks: (*i*) wide area network (WAN) serving as the core network, (*ii*) neighborhood area network (NAN) for connecting the advanced metering infrastructure, and (*iii*) home area network (HAN) used to connect smart home appliances and sensors [35]. Note that the smart grid could have been included in the previous subsection on industrial applications, however we make a distinct subsection on this topic since timing for emerging smart grids is a critical requirement and is the subject of several recent studies [35]–[44].

Table IV lists several smart grid functionalities and their synchronization requirements, ranging from 1 $\mu$s to 100 ms, similar to the requirements of several industrial applications described in the previous subsection. Note that most of the functionalities listed in Table IV cover the areas of the power grid up to the substation. One significant modification to the grid with a stringent synchronization requirement is the wide area monitoring system (WAMS), supporting the use of synchrophasor measurements of timestamped harmonic power components for real time grid control [38], [39], [41], [42].

Note that smart grid is in part about integrating two-way flow of energy and information between the generation and customer premises. With the arrival of novel applications such as demand response, distribute energy resources (e.g., renewables like wind and solar) and advanced metering infrastructure (AMI), new communications and synchronization requirements will emerge. Synchronizing data in a fully integrated manner is expected to become a challenging requirement to support new smart grid applications, as a variety of wired and wireless communications technologies will be used to cover the WAN, NAN, and HAN areas [46].

### D. Other Applications

Many additional applications exist or are emerging that require tight synchronization of devices. As noted in the recent NIST report [3], time-aware systems, such as many Internet-of-Things applications, require highly accurate time-based devices to be deployed in large-scale. Synchronized sensors are important in law enforcement, military and crisis applications. Accurate time is a key element in such applications [47]. As a variety of audio, video, and measurement data are collected in challenging environments, the processing of this information is significantly affected if timestamps are not accurate. Further, time synchronization is also critical in the financial industry, where PTP is currently the only solution meeting the regulatory requirements [48]. For instance, NYSE Euronext, one of the world's leading exchange operators and market access providers, requires sub-100 microsecond to 10 microsecond synchronization range. Also, many Transmission Control Protocol (TCP) based networked applications would see improved performance if all end-nodes involved are synchronized, since it would allow one to consider one-way delays rather than RTTs, as proposed in Sync-TCP [49].

## IV. STANDARDS AND PROTOCOLS

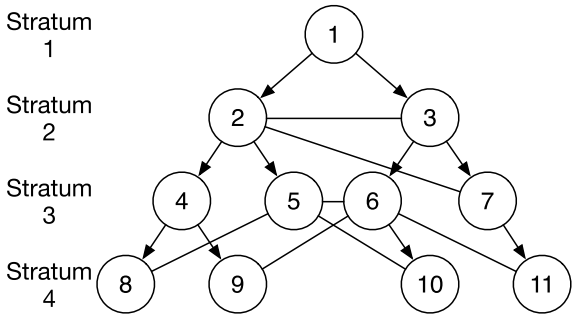
In this section, we review the main synchronization standards, including the Network Time Protocol, Precision Time Protocol, IEEE 802.1AS, SyncE, and White Rabbit.

### A. Network Time Protocol (NTP)

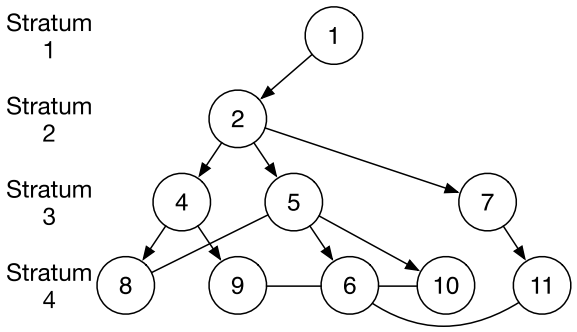
The Network Time Protocol (NTP) is one of the oldest communications protocols used in the Internet [50]–[52], in operation since before 1985. The current version, NTP version 4, is formalized in IETF RFCs 5905-5908 and provides protocol and algorithm specification, public key authentication, and data exchange format.[3] NTP is implemented as an application using User Datagram Protocol (UDP) services, meaning it operates only on the end systems between the NTP server and NTP clients, as depicted in Fig. 6. One or several NTP servers usually have a reliable time source such as from a GNSS or via an atomic clock [50], [52]. This kind of architecture hierarchy is scalable, thus explaining the wide deployment of NTP.

*a) Subnetwork synchronization:* Typically, several stratum NTP servers utilize radio or GNSS time sources to provide accurate and redundant time, which is distributed down through the NTP hierarchy of lower stratum servers [50]. Fig. 5a) depicts an example of synchronized subnetwork, consisting of four stratums, where each stratum is based

[3]The RFCs and up-to-date information on NTP are available on the NTP website: http://www.ntp.org/.




(a) Synchronized NTP subnetwork (without failure).




(b) Synchronized NTP subnetwork, where node 3 fails.

Fig. 5. NTP synchronization example.

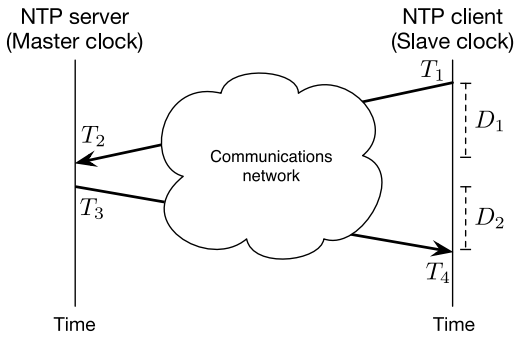



Fig. 6. The Network Time Protocol (NTP).

on the hop count from node 1. In the figure, the straight lines with arrows represent the primary synchronization paths while straight lines indicate backup paths, where the backup paths exchange information, but are not necessarily used by the receiving nodes. Fig. 5b), on the other hand, represents the same subnetwork, but in a scenario where node 3 becomes unavailable and thus the primary and backup paths are updated. For instance, the primary clock source is the one from node 5, and backup clock source is the path going through node 6. Interestingly, due the removal of node 3, the stratum number of node 6 becomes 4, since its connection to a stratum 2 node becomes unavailable.

*b) The NTP protocol:* As shown in Fig. 6, the NTP protocol consists of a periodic two-way handshake initiated by an NTP client followed by a response from the NTP server, whereby the response message contains three timestamps $T_1$, $T_2$, $T_3$, and $T_4$ is recorded at the reception. We let $D_1$ denotes the approximated delay between the client and server and server to client (noted $D_2$), estimated by $D_1 = T_2 - T_1$ and $D_2 = T_4 - T_3$, respectively. It is worth noting that the measured values of $D_1$ and $D_2$ only equal the actual each-way delays in presence of perfect synchronization. For ease of illustration in the following, however, we refer to $D_1$ and $D_2$ as delays. The following equation approximates the average delay [51], [52]:

$$\frac{D_1 + D_2}{2}. \tag{6}$$

Further, the clock offset is given by:

$$\theta_{NTP} = \frac{D_1 - D_2}{2}. \tag{7}$$

In order to achieve accurate synchronization, the upstream (client-to-server) delay and downstream (server-to-client) should be symmetrical such that $D_1 \approx D_2$. If they do not equal, then the offset will be incorrectly estimated, which is why it is important for robust NTP performance to have redundant servers and diverse paths so that such errors are removed by NTP algorithms. Since NTP operates at the application layer, it approximates both the average delay and offset in a black box manner without considering the details of the network mechanisms. In the scenario of Fig. 6, a set of intermediate nodes route the NTP packets between the NTP server and NTP clients. As the NTP packets are routed, several factors can make the connection delays asymmetrical such as routing, queuing, propagation, transmission, and processing delays. These variable delay components are described in the next section, as well as the techniques to obtain efficient synchronization performance with variable delay components. Once the NTP client receives the NTP response from the server, its clock is updated as follows:

$$t_{s,k} \leftarrow t_{s,k} - \theta_{NTP}. \tag{8}$$

NTP is cost-effective as it does not require any specific hardware and scalable due to its hierarchy structure, and capable of long term accuracy to the order of few milliseconds (msecs) [14], [50], [52]. The primary drawbacks compared to using a GNSS or atomic clock at every node solution is the extra communications overhead added for the request-response messages and the limited level of accuracy. However, it can be deployed in a low-cost manner for a variety of applications requiring worse than msec levels of synchronization.

### *B. Precision Time Protocol (PTP) - IEEE 1588*

The Precision Time Protocol (PTP) version 2, IEEE 1588 [53], enables precise synchronization of clocks over heterogeneous systems with accuracy in the microsecond to submicrosecond range. The supported protocols are UDP, IEEE 802.3 (Ethernet), DeviceNet, ControlNet, and PROFINET. NTP targets large-scale distributed applications that need millisecond synchronization, such as, smart grid, industrial

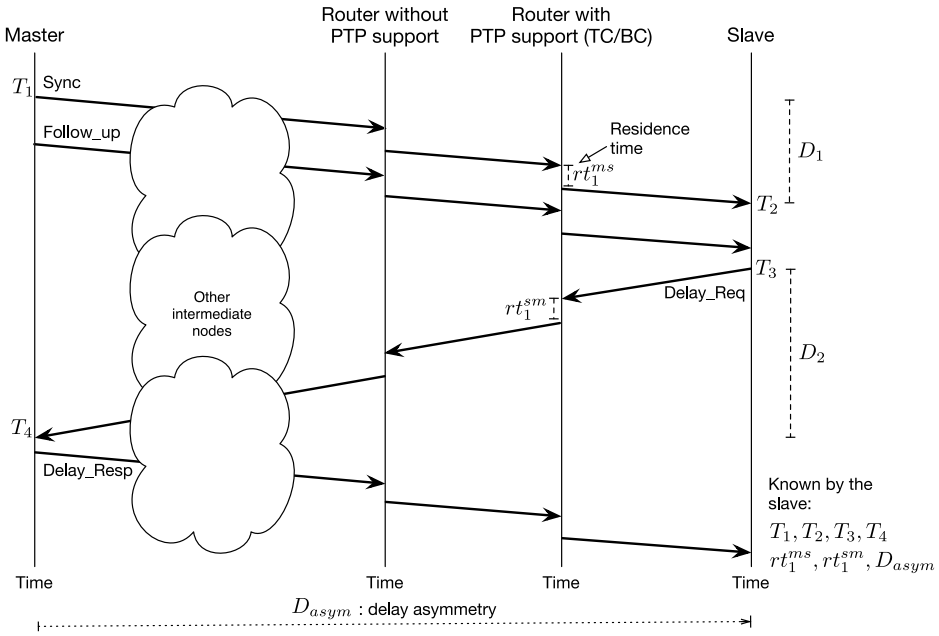



Fig. 7. The PTP synchronization protocol over a topology having routers with and without PTP support. Asymmetry is reduced by considering the residence times ($rt_1^{ms}$ and $rt_1^{sm}$) at routers with PTP support and an end-to-end delay asymmetry field ($D_{asym}$).

automation and telecommunications systems. In a fashion similar to NTP, PTP operates at the application layer.[4] Each PTP entity has one or more ports, communicating with other clocks in the network. The PTP standard defines several clock types:

- *Master clock:* The master clock represents the source of time, multiple clocks on a given path synchronize with it.
- *Ordinary clock:* An ordinary clock has a single port and synchronizes with its associated master clock. The term *slave clock* is also used to refer to an ordinary clock.
- *Boundary clock (BC):* A boundary clock has multiple ports. It can thus act as a master and slave clock. A boundary clock is normally not associated with application domain devices (e.g., actuators or sensors).
- *Transparent clock (TC):* A TC is a key clock type to meet microsecond synchronization requirements over multiple network nodes from a given slave clock to the master clock [14], [55]. Specifically, a TC is a device capable of measuring the residence time in a router/switch, which includes for instance the measurement of processing and queuing delays. The residence times are taken into account to measure accurately the end-to-end delay at the slave clock.
- *Peer-to-peer transparent clock:* The peer-to-peer transport clock ports, measures not only the residence time information, but also measures the link propagation delay between a similarly equipped port at the other end of the link.

We note that, the introduction of transparent and peer-to-peer clocks require special PTP hardware at intermediate nodes, as opposed to NTP which does not necessitate any special device between the slave and master nodes.

*c) The best master clock algorithm:* At each slave node, the best master clock algorithm is executed such that each clock synchronizes with the best clock. More specifically, each slave node scans its local ports for *Annouce* messages and selects the best one as a master where best is in terms of priority, clock class, and accuracy. Note that a failure of the master clock requires a slave node to re-elect a new master clock, whereby during this interval slave nodes are running without synchronization service [56]. To overcome this problem, the authors of [56] proposed to maintain group of master clocks and synchronize using a democratic algorithm for improved convergence time.

*d) The PTP synchronization protocol:* The overall PTP synchronization protocol is similar to NTP, however there are a certain number of improvements in PTP to meet sub-microsecond synchronization requirements. The main steps of the PTP request-response protocol, depicted in Fig. 7, are described as follows:

- The master clock periodically sends a *Sync* message to a given slave node containing the send time $T_1$. The rate of *Sync* messages is typically set to 1-2 seconds [54], [57]. For systems requiring highly precise synchronization, the rate can be set as low as few microseconds (e.g., 125 $\mu$s in [58]). Generally, increasing the *Sync* rate decreases the mean error [59].
- Optionally depending on the timestamp technique (with or without hardware support), a *Follow_up* message containing $T_1$ is sent.
- Once the message containing $T_1$ is received by the slave node, the receive time $T_2$ is recorded.
- The slave node then sends a *Delay_req* message at $T_3$ to the master node.

[4]NIST recently proposed an application framework for the PTP standard to facilitate code reutilization [54].

- Finally, the master node notes the reception time $T_4$ and sends it back to the slave node in a *Delay_resp* message. At the reception of the *Delay_resp* message, the slave node is aware of all required timestamps $(T_1, T_2, T_3, T_4)$ which are used to correct its local clock.

One significant difference with NTP is that the overall process is initiated by the master clock in PTP. Further, PTP includes techniques to improve the time synchronization accuracy, which are covered in the following.

*e) Delay and offset approximations:* The delay and offset approximations follow a similar structure as the one proposed in NTP, as detailed in Eqs. (6)-(7). However, PTP introduces three mechanisms to mitigate the negative effects of asymmetric links on the synchronization performance:

- *Residence time at intermediate nodes (TCs and BCs):* The residence time at a given router/switch consists of the time duration a PTP packet resides in the switching fabric from the input port to the output port. In Fig. 7, two residence times, $rt_1^{ms}$ and $rt_1^{sm}$, are measured at the router with PTP support. $rt_1^{ms}$ corresponds to the residence time experienced at the router with PTP support on the downstream direction (from the master to slave node), whereas $rt_1^{sm}$ corresponds to the residence time on the upstream direction. These residence time values are included in the *Sync*, *Follow_up* and *Delay_req* messages. Thus PTP, is more costly since intermediate nodes (TCs or BCs) need to support PTP.
- *Asymmetric delay parameter (noted $D_{asym}$):* If the fixed asymmetric delay properties are known for a given connection, an asymmetric delay parameter can be used [60], corresponding to the *delayAsymmetry* field in IEEE 1588. It is worth noting that the *delayAsymmetry* field is added in the *correctionField*, where the latter field integrates all correction delays.
- *Peer-to-peer path correction:* Peer-to-peer transparent ports measure the link propagation delays and include this in the *correctionField*. This mechanism helps to better estimate the delay asymmetry, at the expense of an increased communications overhead.

In Fig. 7, a slave clock synchronizes with the master clock following the PTP protocol, whereby one intermediate node has a transparent clock and measures the residence time $rt_1^{ms}$ while exchanging the *Sync* or *Follow_up* message and $rt_1^{sm}$ while exchanging the *Delay_req* message. Further, for nodes without PTP support, such as the regular router in Fig. 7, it is recommended that it process PTP messages with high priority to decrease the residence time. For improved accuracy, all nodes in the example of Fig. 7 should have transparent clocks. The IEEE 1588 standard also recommends to generate timestamps (for $T_{1-4}$) at the MAC or physical layer for improved accuracy. Let us next assume a network with PTP support and $N$ intermediate nodes implementing the end-to-end transparent clock mechanism between a master node and slave nodes. Let $D_{ms}$ and $D_{sm}$ denote the corrected downstream and upstream delays, to be defined shortly. We let $D_{asym}$ denote the delay asymmetry and note the end-to-end delay $\bar{d}_{PTP}$ is approximated as [53, Clause 11]:

$$\bar{d}_{PTP} = \frac{D_{ms} + D_{sm}}{2}, \tag{9}$$

whereby

$$\bar{d}_{PTP} = \frac{\left(D_1 - \sum_{i=1}^{N} rt_i^{ms} + D_{asym}\right) + \left(D_2 - \sum_{i=1}^{N} rt_i^{sm} - D_{asym}\right)}{2}, \tag{10}$$

where $rt_i^{ms}$ and $rt_i^{sm}$ are the residence times at intermediate node $i$ from the master to the slave and slave to master, respectively. Note that in the PTP standard, both the residence times and the asymmetry parameter are combined in a *correctionField*. However, for improved readability, we distinguish both in the following. The slave offset, integrating the correction terms to mitigate asymmetry, is given by:

$$\theta_{PTP} = T_2 - T_1 - \bar{d}_{PTP} - \sum_{i=1}^{N} rt_i^{ms} + D_{asym}, \tag{11}$$

Note that $D_{asym}$ must be set such that

$$D_{ms} = \bar{d}_{PTP} + D_{asym} \tag{12}$$

and

$$D_{sm} = \bar{d}_{PTP} - D_{asym} \tag{13}$$

hold.

Using the approximated clock offset (Eq. (11)), a slave clock $k$ can adjust its local time $t_{s,k}$ as follows:

$$t_{s,k} \leftarrow t_{s,k} - \theta_{PTP}. \tag{14}$$

Note that servo clocks are commonly used in order to frequently update the local frequency.

*f) Syntonization:* So far in this section, we mainly covered the synchronization protocol, which aims at synchronizing a slave node to its associated master clock by executing the PTP synchronization protocol. It is worth noting that *Sync* messages are sent to slave nodes at a certain transmission interval, and depending of the frequency of both clocks, the time at the slave node premise does not equal the master clock time between two *Sync* messages. Recall from Eq. (2) that the deterministic skew of the offset, noted $\gamma_{s,k} \cdot t_m$, represents the frequent slave clock deviation with respect to the master clock. To correct this deviation, PTP provides a syntonization algorithm to correct a slave clock between two *Sync* messages. Specifically, the PTP syntonization mechanism can be summarized by defining the ratio of the master clock frequency to the frequency of the slave clock [53, Clause 12]:

$$\frac{T_{1,0} - T_{1,-E}}{t_{c,0} - t_{c,-E}}, \tag{15}$$

where $T_{1,0}$ corresponds to the timestamp of the most recent *Sync* message generation time ($T_1$ in Fig. 7) and $T_{1,-E}$ for the previous *Sync* message generation time at epoch $-E$. $t_{c,0}$ and $t_{c,-E}$ correspond to the corrected timestamps of the *Sync* once

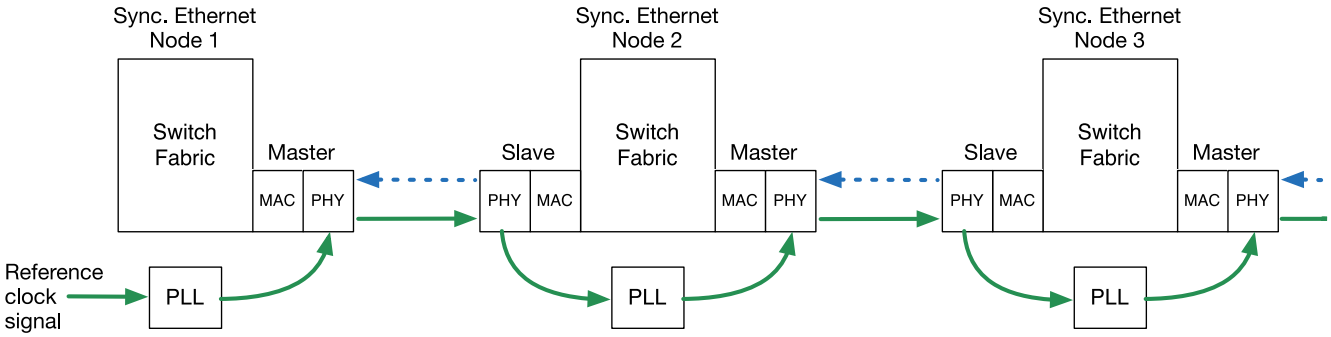



Fig. 8. Synchronous Ethernet (SyncE) principle [68].

received by the slave node (corresponding to approximately $T_2$ in Fig. 7) at epochs 0 and $-E$, respectively. The corrected timestamp at any epoch $X$ corresponds to:

$$t_{c,X} \leftarrow T_{1,X} + \bar{d}_{PTP} + \sum_{i=1}^{N} rt_i^{ms} + D_{asym}. \tag{16}$$

The measured performance of PTP under different conditions has been reported in the literature. In testbeds following most of the PTP standard recommendations, a synchronization accuracy of approximately 50 ns was achieved [23], [45]. A synchronization precision of 2 ns under ideal conditions was obtained by implementing all PTP features in hardware, including the timestamping function [59]. However, under the presence of multi-hop communications with non PTP routers, significantly worst synchronization accuracies of as low as 450 $\mu$s were measured under the presence of asymmetric delays [61]. Furthermore, as the number of cascaded transparent/boundary clocks increases, the synchronization error increases nonlinearly as demonstrated in [14] and [55]. Therefore, it is important to understand the limitations of PTP depending on the application environment and requirements.

*g) PTP profiles:* Profiles were introduced in PTP in order to allow other standards (e.g., ITU standards) to adapt PTP for specific applications. For instance, the telecom profile ITU-T G.8265.1 [62] has been released to allow end-to-end frequency distribution and synchronization and ITU-T G.8275.1/2 for time/phase distribution in LTE-A TDD access networks. Similarly, IEEE C37.238-2011 defines a set of PTP parameters and options to be used in Smart Grid power systems applications utilizing Ethernet communications.

### C. IEEE 802.1 TSN, IEEE 802.1AS

IEEE 802.1 Time Sensitive Networking (TSN) is a set of standards being developed to support time synchronized low latency traffic transport on IEEE 802 based networks. Some of the key features are given as follows [63], [64]:

- *IEEE 802.1AS-2011:* Layer 2 time synchronization, corresponding to a specific IEEE 1588 (PTP) profile.
- *IEEE 802.1Qat-2010:* Admission control for bridges in order to guarantee resources for audio and video streams.
- *IEEE 802.1Qav-2010:* Enhancement of the frame forwarding mechanism for time sensitive streams in order to improve the latency.
- *IEEE 802.1BA-2011:* Definition of usage profiles to provide interoperable communications between time sensitive devices.

The IEEE 802.1AS group sets standards for Audio Video Bridging (AVB) to transport packets for time sensitive and high quality multimedia traffic [65], [66]. It is also aimed for use in cyber-physical systems and sensor networks.[5] AVB traffic requires robust jitter and precise synchronization and should be prioritized. Furthermore, IEEE 8021AS ensures efficient synchronization performance to support both IEEE 802.3 (Ethernet) and IEEE 802.11 (WiFi) technologies. The AVB industry requires time synchronization to the order of 500 ns [66]. More specifically, an IEEE 802.1AS bridge implements an IEEE 1588 boundary clock and the end users implement an ordinary clock. Thus, IEEE 802.1AS does not provide new timing mechanisms, but reuses the existing PTP standard. However, compared to PTP, IEEE 802.1AS simplifies some PTP features to improve the stability over large networks. For instance, the grandmaster clock selection in IEEE 802.1AS provides faster grandmaster failover. IEEE 802.1AS also provides media-dependant delay measurement improvement for IEEE 802.11 (WiFi), IEEE 802.3, and other coordinated shared networks (CSNs). As IEEE 802.1AS is based on PTP, for the remainder of this paper, we do not further elaborate on IEEE 802.1AS.

### D. Synchronous Ethernet (SyncE) - ITU-T G.8262

Another significant recent synchronization standard is Synchronous Ethernet (SyncE), standardized by the International Telecommunication Union (ITU) in ITU-T G.8262 [67]. Note that the native Ethernet protocol works only between two adjacent hops without any need of frequency synchronization, where the frequency clock accuracy is $\pm 100$ ppm. Effectively, the preamble in Ethernet frames allows a PHY layer receiver to initialize a new transmission and extract the bit stream directly. The fundamental synchronization principle of SyncE, depicted in Fig. 8, consists of a clock signal transmitted at the physical layer of

[5]Tutorial: The Time-Synchronization Standard from the AVB/TSN suite IEEE Std 802.1ASTM-2011. http://ieee802.org/1/files/public/docs2014/as-kbstanton-8021AS-tutorial-0714-v01.pdf

TABLE V
COMPARISON OF THE PRIMARY SYNCHRONIZATION/TIME TECHNOLOGIES

| | **NTP (RFC 5905-5908)** | **PTP (IEEE 1588-2008)** | **SyncE (ITU-T G.8262)** | **White Rabbit** |
|---|---|---|---|---|
| *Applications* | Internet, large-scale networks. | Telecommunications, industrial automation, smart grid, and audio video bridging. | Telecommunications and industrial automation. | Particle accelerator, cosmic particle detector. |
| *Target Accuracy* | Millisecond. | Microsecond. | $\pm$4.6 ppm. | Sub-nanosecond. |
| *Asymmetry Mitigation* | No. | Yes: (*i*) Residence time measurement, (*ii*) Asymmetry parameter $D_{asym}$, and (*iii*) Peer-to-peer path correction. | Not required. | Yes. |
| *Pros* | Cost-effective. Easy to implement and deploy. | Reliable synchronization. High accuracy. | Based on Ethernet. High accuracy. Reliable end-to-end exchange. | Sub-nanosecond accuracy. Deterministic packet delivery. |
| *Cons* | Unreliable synchronization (operates as a black box). Millisecond to second accuracy. | Most nodes need to implement the protocol. Installation expensive. | Operates at the physical layer - all nodes need to implement SyncE. Can distribute the frequency only. | Need to support both PTP and SyncE, costly. |

Ethernet and propagated to all Ethernet nodes with SyncE support in a given synchronized network. Once a reference signal is received, the slave acts as a timing master. The recovered clock at each intermediate hop (green thick arrow in Fig. 8) is cleaned with a phase-locked loop (PLL) and is then passed to a clock multiplier unit (CMU) which then generates the output clock. The PLL attenuates the accumulated clock jitter. Timing devices in a SyncE network must support $\pm$4.6 ppm in free-run, corresponding to the worst case performance scenario [68]. Further, SyncE works on a hierarchical master-slave manner, as depicted in Fig. 8. The SyncE node generating the reference clock transmits the clock signal to its adjacent Ethernet node slave port. The receiving slave port then forwards the signal to the transceiver, which acts as a master port. As the reference clock signal gets propagated over all adjacent SyncE nodes, the overall network gets synchronized. Also, the PHY layer sends transmissions continuously in parallel with the regular data transmissions, in a fashion to Synchronous Optical Networks (SONETs) [68]. For in-depth implementation discussions on SyncE in telecommunication systems, the interested readers are referred to the recent survey [68].

It is worth noting that in a given network path, all nodes must support SyncE to be synchronized. Further, as it operates at the physical layer, only the frequency can be recovered, which is a major reason for its deployment in mobile networks to synchronize, for instance, the frequency allocations efficiently [69]–[71]. Therefore, SyncE does not provide Time-of-Day (ToD) synchronization, which is a key feature of the PTP protocol. As pointed out in [70], SyncE and PTP are complementary; SyncE can not only be used to provide tight frequency synchronization, but also to improve the PTP performance, where PTP can use frequency timing inputs from SyncE. The combination of SyncE and PTP was used in a telescope for nanosecond timing resolution [72]. Further, the White Rabbit synchronization system, described in the following, also combines PTP and SyncE in particle accelerators and detectors.

### *E. White Rabbit*

White Rabbit (WR) is a synchronization technology combining the PTP and SyncE standards in order to meet sub-nanosecond accuracy requirements for particle accelerators and cosmic particle detectors [73]–[75]. These applications require deterministic and reliable data delivery as well as sub-nanosecond synchronization accuracy. A key feature provided by PTP used in White Rabbit is the residence time measurement to take into account the intermediate hop variable delays. WR aims at detecting asymmetry undetected by PTP in order to significantly improve the synchronization accuracy. Some of the key improvements and mechanisms proposed by WR are listed below:

- *WR link setup:* Prior to executing the regular PTP protocol, an initialization procedure operates for node identification, syntonization, and for providing the links parameters.
- *WR link delay model:* The delay from a master $m$ to slave $s$ is estimated according to fixed and variable parameters:

$$\Delta_{tx} + \delta_{ms} + \Delta_{rx}, \tag{17}$$

  where $\Delta_{tx}$ and $\Delta_{rx}$ are fixed delays for the transmission circuitry at the transmission and reception sides, respectively, and $\delta_{ms}$ is a variable delay of the transmission medium considering the fiber refractive indexes.
- *Syntonization using SyncE:* For improved syntonization, SyncE is used between master and slave nodes to share the same frequency.

### *F. Comparison*

Table V compares the principal synchronization technologies, that is, NTP, PTP, SyncE, and White Rabbit. We observe that there is a clear trade-off between cost and synchronization accuracy. NTP was initially designed to provide synchronization for large-scale networks such as the Internet and aims at meeting millisecond accuracy requirements. On the other hand, PTP requires more investment and configuration efforts,

but meets microsecond synchronization requirements. SyncE is expected to be widely used in mobile networks as it provides precise frequency synchronization. Some major pros of SyncE are that it is based on Ethernet, and it provides reliable end-to-end frequency synchronization. However, one significant drawback is that all nodes all depend on a specific physical layer technology. For sub-nanosecond accuracy requirements, White Rabbit can be used, which combines both PTP and SyncE, but is the most costly option and cannot be used for large scale deployment (e.g., for the Internet) currently.

### G. Applications

Revisiting the motivating applications of Section II we note the protocols and standards being adopted.

*1) Mobile Cellular Networks:* The technologies considered for cost-effective mobile cellular network synchronization are GNSS, IEEE 1588, and Synchronous Ethernet. Although GNSS provides nanosecond synchronization accuracy and can be widely deployed in the United States, indoor deployment, for instance required for small indoor cells, is difficult due to low GNSS satellite signal reception. However, SyncE operates at the physical layer and is not intended for time of day distribution. Therefore, it is likely that a combination of GNSS/GPS as the time source with IEEE 1588 Precision Time Protocol (PTP) for distribution of time together with SynchE for distribution of frequency will be adopted [22], [70] in mobile cellular networks.

*2) Industrial Applications:* Most of the industrial applications of Section II-B are planning to use PTP. In [28], the implemented flight testbed was shown to provide less than 100 nanoseconds synchronization performance using PTP with hardware support, thus meeting the 15 $\mu$s requirement. Also, a navy testbed using PTP was shown to provide approximately 17 microseconds precision, thus also meeting their requirement of 500 $\mu$s.

*3) Smart Grid:* As in the industrial applications, PTP was chosen to support microsecond synchronization precision in several recent works [38]–[43]. Note that substations currently use a variety of architectures to have time synchronization services, including GNSS and IRIG [39]. With an increasing number of Intelligent Electrical Devices (IEDs) (e.g., voltage regulators, protective relays, recloser controllers, various sensors, etc.) to smarten power grids, there is an increasing interest to distribute synchronized time in a cost-effective manner, for example by using PTP [39]. Equipping each IED with a dedicated GNSS module would be economically impractical [42]. Furthermore, the IEDs may be housed indoors or underground. NIST recently designed a testbed to verify the synchronization performance capabilities of PTP for power distribution applications, as depicted in Fig. 9 [39]. In this testbed, the ordinary clocks (OCs) synchronize with the grandmasters (GMs) interconnected via boundary or transparent clocks (BCs/TCs) with PTP hardware support. The pulse per second (PPS) system records all clocks simultaneously to calculate statistics and quantify the synchronization performance. A traffic generator is also interconnected with the TCs/BCs in order to challenge the synchronization performance with variable traffic loads. The testbed allows one to verify the performance of both the synchronization system as well as validate current and future smart grid applications. The results in their experiments show robust synchronization accuracy of approximately $\pm$ 60 ns, thus meeting the most challenging requirement of 1 $\mu$s.

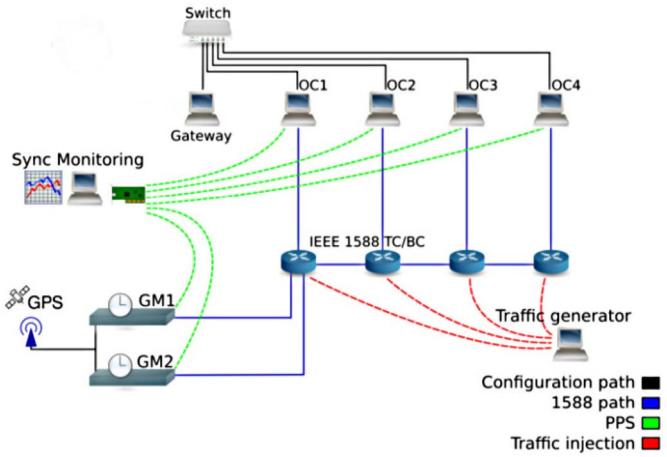



Fig. 9. NIST PTP testbed for power applications [39].

## V. Protocols for Wireless Sensor Networks (WSNs)

As previously discussed, there is an increasing need to timestamp events in IoT applications and time-aware systems. Distributed low-cost sensing devices need to be widely deployed to monitor physical phenomena such as temperature, light, pressure, and so forth [76]. These applications often use wireless communications and can be classified as Wireless Sensor Networks (WSNs). WSNs need to support data sensing, data communications, and processing, and provide energy supply in a low-cost manner. WSN nodes typically contain a battery and are thus energy constrained. Further, the collected data often needs to be associated with a temporal metadata, and thus time synchronization needs to be supported by the WSN. In the previous section, the main protocols and standards usually used in computer networks were overviewed. They are however not necessarily applicable for WSN applications which require efficient energy utilization [77]. Also, WSNs are bandwidth constrained compared to wired networks, and thus the synchronization traffic need to be limited. Furthermore, WSNs are generally duty cycled embedded systems, characterized by long sleep state intervals and short transmission state intervals [7]. In the following, we review the main synchronization protocols specifically proposed for WSNs, which can be categorized into static and dynamic protocols.

### A. Static Protocols

In static protocols for WSNs, a given node synchronizes to its associated reference node following a fixed periodic transmission interval $\tau$. The PTP and NTP protocols described in the previous section also have such a parameter.

*1) Reference Broadcast Synchronization (RBS):* Elson *et al.* [78] proposed the reference broadcast synchronization (RBS) protocol for WSNs in order to synchronization clusters of nodes, where each one can communicate with each other. The reference node, referred to as beacon node, broadcasts a beacon message to the cluster nodes every $\tau$ second. The receiving nodes then transmit their respective reception times. Therefore, such a scheme does not need to deal with sending and transmission delays, thus reducing the potential sources of error. Based on the receiving times, each node calculates a relative time-scale.

*2) Timing-Sync Protocol for Sensor Networks (TPSN):* Timing-sync Protocol for Sensor Networks (TPSN) considers a tree like hierarchy, where the root node is the source of time [79]. Each node at a given level synchronizes to the lower levels. The overall TPSN principle consists of two phases. In the first phase, the level discovery phase, the root node broadcasts a discovery packet, then the receiving nodes assign its level to one and broadcast again a discovery packet. It is broadcasted until the overall WSN is constructed. Then the synchronization phase begins, where the root node sends a synchronization packet. When the level 1 nodes receive the packets, it executes a two-way communications between each other. To reduce asymmetry, MAC layer timestamps are used. According to [76], such MAC timestamps are more convenient in WSNs compared to computer networks, since the WSN nodes architecture is more coupled to the lower layers.

*3) Tini-Sync and Mini-Sync (TS-MS):* To simplify the synchronization mechanism for WSNs, Sichitiu and Veerarittiphan [80] proposed Tini-Sync and Mini-Sync. TS-MS contains two fundamental mechanisms. Firstly, the data collection step consists of recording a set of two or more timestamp pairs using two-way packet exchanges between a reference node and given slave node. Then, using the collected data points, a line fitting mechanism is used in order to bound the clock offset and skew.

*4) Lightweight Time Synchronization (LTS):* In [81] the Lightweight Time Synchronization (LTS) protocol was proposed in two versions: (*i*) Centralized multi-hop LTS and (*ii*) Distributed multi-hop LTS. In the centralized scheme, the reference node initializes the synchronization process, where three packets are exchanged. In the distributed protocol, each node to be synchronized initializes the process with the reference node, where only two packets need to be exchanged. One significant novelty proposed by the LTS protocol is the construction of a minimum spanning tree at each synchronization round. Further, if there are intermediate nodes between the slave and reference nodes, all intermediate nodes need to be synchronized first. More specifically, each node synchronizes to its immediate parent node.

*5) Flooding Time Synchronization Protocol (FTSP):* The Flooding Time Synchronization Protocol (FTSP), as opposed to TPSN and its predecessors, is a unidirectional, ad-hoc, and tree-based protocol, where a root node is elected to provide the source of time [82]. As it is a static protocol, the synchronization rounds are based on a fixed interval $\tau$. The $\tau$ parameter is manually configured based on the operating environment. Fig. 10 depicts an FTSP execution example, where the root node, 2, broadcasts a Time Message containing notably its identifier, sequence number, and its timestamp. The nodes in the transmission range receive the Time Message and adjust their local times. The receiving nodes increase the sequence number and broadcast the message. As the process is one-way, the potential time errors are reduced, and FTSP uses MAC layer timestamping. Further, FTSP takes into account the interrupt request delays from the micro-controller (e.g., handling, encoding, decoding, and byte alignment times) for improved precision [76]. Further, FTSP uses linear regression to deal with the clock skew according to a set of received messages. Further, FTSP is the default synchronization protocol in TinyOS,[6] a popular operating system for WSNs.

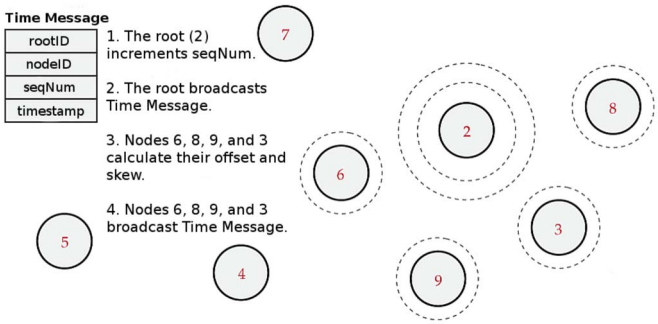



Fig. 10. FTSP execution [76].

## B. Dynamic Protocols

The dynamic synchronization protocols for WSNs basically monitor node properties, such as the variation of the clock frequency, in order to adjust the transmission interval $\tau$. Temperature and manufacturing inaccuracies are the two most significant frequency error sources [7], [83]. Manufacturing imprecision is static over time, however temperature changes frequently and thus needs to be taken into account dynamically. Recall in the static approaches, $\tau$ is fixed depending on the environment. In the following, we review the dynamic synchronization protocols.

*a) Rate adaptive time synchronization (RATS):* In Ganeriwal *et al.* [84], part of the Roseline project,[7] proposed the Rate Adaptive Time Synchronization (RATS) which considers a set of timestamp samples in order to model a linear clock model. More specifically, using the timestamp samples, a prediction error $E_p$ is compared to a known application error bound $E_{max}$. $\tau$ is multiplicatively increased when $E_p \leq E_{max}$ and decreased otherwise.

## C. Temperature Compensated Time Synchronization (TCTS)

In [85], the authors focused on the temperature, which is a significant source of clock drift. Their proposal, TCTS, has

[6]FTSP in TinyOS: http://tinyos.stanford.edu/tinyos-wiki/index.php/FTSP.

[7]The Roseline Project is a collaborative project aiming at building a system stack that enables new ways for clock hardware, operating system, network services, and applications to learn, maintain and exchange information about time in the context of emerging CPSs. https://sites.google.com/site/roselineproject/.

TABLE VI
COMPARISON OF THE SYNCHRONIZATION PROTOCOLS FOR WSNS

| | **RBS** [78] | **TPSN [79]** | **TS-MS [80]** | **LTS [81]** | **FTSP [82]** | **RATS [84]** | **TCTS [85]** | **D-FTSP [89]** |
|---|---|---|---|---|---|---|---|---|
| **Messages Protocol** | | | | | | | | |
| Reference broadcast | X | | | | | | | |
| One-way | | | | | X | | | X |
| Two-way | | X | X | X | | X | X | |
| **Multi-hop** | | | | | | | N/A | |
| All nodes | | X | | X | X | X | | X |
| Subset of nodes | X | | X | | | | | |
| **Communications Interval** ($\tau$) | | | | | | | | |
| Fixed | X | X | X | X | X | | | |
| Dynamic - Error estimation | | | | | | X | | X |
| Dynamic - Calibration | | | | | | | X | |

two main phases, namely the calibration phase and compensation phase. In the calibration phase, a calibration table is constructed, which contains the frequency errors for a range of temperatures. To do so, each node takes temperature sampling and determines the frequency error. Once the calibration table is filled, the reference node increases $\tau$ and then the compensation phase starts. In the compensation phase, the calibration table is used for correcting the frequency by taking temperature samplings. One significant advantage of TCTS is that it can continue synchronizing when the connection becomes unavailable.

Recently as part of the Roseline project, [86], [87] MEMS-based oscillators were proposed, offering a silicon-based alternative to crystal based clocks for WSN nodes. One of its key components is a temperature-to-digital converter which enables precise compensation of the temperature noise by measuring the MEMS resonant frequency.

### D. Dynamic FTSP (D-FTSP)

Shannon *et al.* [76], [88], [89] proposed dynamic FTSP by adapting the transmission interval $\tau$ such that a specific application phase error bound $\varepsilon$ is met. The approach is similar to a preliminary dynamic mechanism proposed by Schmid *et al.* [7]. A key parameter taken into account by D-FTSP is the clock drift, noted $\phi$, which correspond to the variation of a clock frequency. From a given sensor node perspective, the maximum clock drift $\phi_{max}$ of the neighbors is taken. Then, the relation between $\varepsilon$ and $\tau$ is given in D-FTSP as follows:

$$\varepsilon = \phi_{max} \cdot \frac{\tau^2}{2}. \tag{18}$$

D-FTSP also considers an average phase error $\varepsilon_{avg}$ bound to not exceed. Then, in order to find $\tau$, it is increased such that $\varepsilon < \varepsilon_{avg}$ holds. D-FTSP has been experimentally investigated and was shown to obtain similar synchronization performance compared to FTSP in a stable temperature environment, while reducing as much as 75% the number of synchronization transmissions, which significantly helps at reducing the energy consumption - a key property for WSNs.

### E. Comparison

Static synchronization protocols for WSNs consider a pre-configured communications interval $\tau$ in order to synchronize a cluster of sensor nodes to a reference source node. The FTSP and TPSN protocols use MAC layer timestamping in order to reduce the communications delay uncertainties, thus requiring access to lower layers. As opposed to its predecessors, FTSP uses a one-way synchronization scheme in order to improve energy efficiency. However, static synchronization protocols do not take into account dynamic environmental parameters, such as the temperature, which influence the frequency drift and time synchronization performance. To overcome this shortcoming, dynamic synchronization protocols have been proposed. Those dynamic protocols usually consider a phase error threshold and adapt $\tau$ in order to meet the threshold. Adapting $\tau$ dynamically allows to decrease the number of synchronization messages and thus energy consumption. Table VI summarizes the main synchronization protocols for WSNs. Interested readers are referred to [76] for in depth readings on D-FTSP and other synchronization protocols for wireless sensor and WiFi networks.

## VI. ACCURACY IMPROVEMENT MECHANISMS

Recent works introduced new mechanisms to improve the synchronization performance over packet-switched networks. Fig. 11 depicts our categorization of the main improvement mechanisms. At a high level, the improvement mechanisms can be grouped into those that require new specification/configuration, new software, and/or new hardware deployments. Generally, software and configuration-based mechanisms are cheaper to implement compared to hardware-based schemes, which are all described in detail in the following. In this section, we focus on the techniques for improving the accuracy of the NTP and PTP standards, which are the most widely investigated synchronization protocols.

### A. Configuration-Based Improvement Mechanism

*1) Quality-of-Service (QoS):* To mitigate the negative effect of variable delays, especially at intermediate nodes, the PTP standard recommends two main techniques: (*i*) processing PTP messages with high priority and (*ii*) measuring residence times to avoid poor synchronization performance with asymmetric delays. For routers having quality-of-service (QoS), supporting prioritized packets based on their packet class, they can be configured such that PTP messages are processed first with high priority, thus reducing the residence times. Adopting this

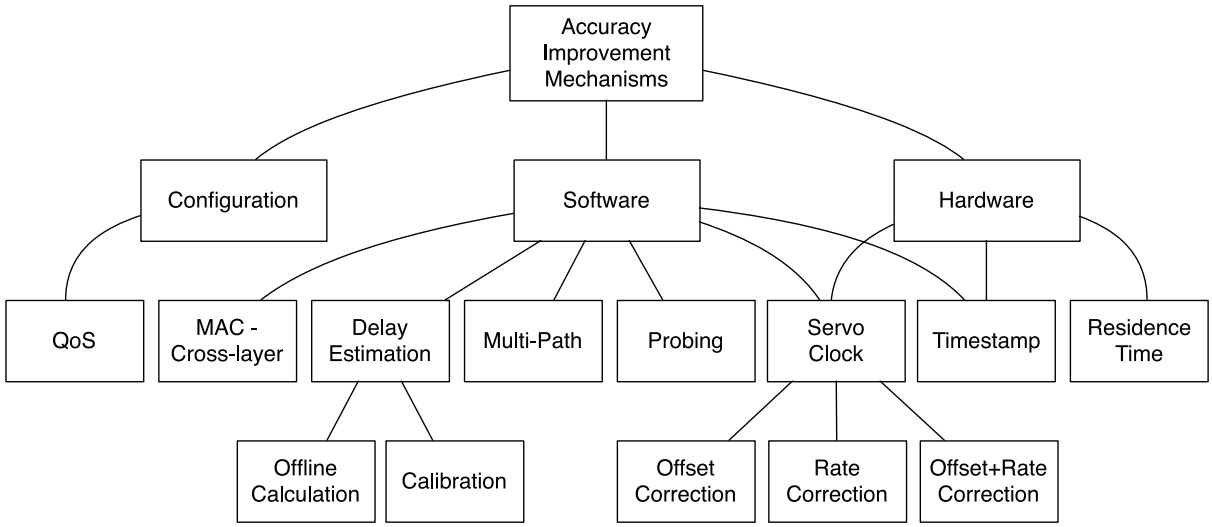



Fig. 11. Synchronization improvement schemes over packet-switched networks.

approach is cost-effective, since only a few configurations are required at each node with QoS support. However, it is worth noting that if multiple PTP messages access the same router at the same time, there will still be non negligible queuing delays, as noted in [90]. To overcome this issue, a token-based mechanism was proposed, whereby a given slave node can transmit PTP messages only when it holds a token [90]. The proposed token-based was shown to be operational in a PTP system without introducing significant additional traffic.

### B. Software-Based Improvement Mechanisms

*1) MAC Cross-Layer:* Cross-layer optimization is a well known technique for improving the performance of systems following the waterfall-like principle of the Open Systems Interconnection (OSI) communications model, whereby instead of communicating between layers following the interfaces, one or several layers are bypassed for optimization purposes. PTP and NTP were both designed to be implemented at the application layer to facilitate wide deployment. These standards are therefore not aware of the implementations and characteristics of the communications mediums. It is worth noting that significant delays are experienced between the application and physical layers. For example, several wireless systems implement a distributed coordination function (DCF) to manage wireless channels, such as IEEE 802.11 (WiFi). The DCF mechanism contains a random backoff mechanism to avoid collision. Such a random mechanism makes the synchronization operation to be challenging since the communications performance is not deterministic. WiFi, thus, has a time division multiplexing protocol for carrying timestamps by periodically exchanging beacons. This protocol was extended in [91] by incorporating PTP messages in beacon frames, thus significantly improving the synchronization performance. The obtained experimental results was shown to provide 1.5 $\mu$s mean error by embedding PTP in the WiFi chipset, as opposed to approximately 7 $\mu$s by using the PTP standard. The MAC chipset needs however to be modified to take into account this cross-layer optimization.

*2) Delay Estimation:* As discussed in previous sections, asymmetric links can significantly increase the time synchronization mean error. PTP introduces different mechanisms to improve the synchronization performance with asymmetric links by introducing correction mechanisms as we introduced in the previous section. Those correction mechanisms however require extra hardware support and asymmetry delay measurement. A different strategy consists of estimating the asymmetry by either using offline calculation models or via calibration. In the offline calculation approach, the asymmetric delays are estimated by modeling analytically the main network performance components [92]–[94]. One scheme proposed by [92] consists of considering the transmission delay as a major probable asymmetry delay component:

$$\frac{F}{R_{s,n}}, \tag{19}$$

where $F$ is the frame size and $R_{s,n}$ the data rate between node $s$ and adjacent node $n$. Note that with this model, an asymmetry delay is present when $R_{s,n} \neq R_{n,s}$. The obtained results were shown to give $10^{-3}$ ns offset error. Such a model might give a close approximation in certain systems, however it does not account for the propagation, queueing, and protocol overhead delays. Note that the transmission delay is negligible with a large capacity $R_{s,n}$ or in networks using a random backoff mechanism for channel access [95].

For improved delay approximations, a calibration technique can be used, which corresponds to the actual experimental performance measurement under a variety of network conditions [96]–[99]. Using real-world measurements, the exact delays can be used by the synchronization algorithm to take into account asymmetric delays. In [97], the authors proposed to estimate the delays using a measurement-driven scheme by correlating path delay and traffic load under the main assumption that link delay is a function of the load (e.g., at high loads, queuing is a major delay component). The experimental results showed an approximately 11% estimation error minimization. Also, packet-switched networks are event-triggered systems, as the physical mediums are

shared statistically. Performance bounds can be quantified to meet sub-microsecond synchronization accuracy using a traffic shaping mechanism as proposed and evaluated in [99].

*3) Multi-Path:* The IEEE 1588 PTP standard recommends the same path for upstream and downstream communications to decrease the probability of having asymmetric delays. Some works looked at extending PTP with multi-path support [100]. Maintaining multiple paths between slaves and masters improve redundancy, which inherently improves availability and decreases recovery time after a link failure [57]. One drawback of PTPv2 is that the architecture depends on a single time source, which makes it vulnerable to byzantine failures [48]. Further, multiple paths can not only be used to improve availability, but also for enhancing synchronization accuracy, as proposed by the *Slave Diversity* mechanism [101]. The Slave Diversity technique, similarly to the antenna diversity mechanism in wireless communications, combines multiple path delays to mitigate the negative effects on the synchronization performance of having variable asymmetric delays. The performance improvement was illustrated in [101], where a few microseconds synchronization accuracy was achieved by combining paths with $\pm 10$ $\mu$s variable delays. A Kalman filter for multi-path network synchronization was proposed in [102], and introduces a memory effect to obtain more accurate estimate of the delay by using at least 2 alternate paths.

*4) Probing:* NTP and PTP protocols are probing-based schemes in that control packets (*Sync*, *Delay_req*, etc.) containing the timestamps are routed in the network in order to both transfer the timestamps and probe the network to estimate the approximate end-to-end delay and offset. Several works have investigated different probing-based schemes by sending probe messages and analyzing the sending and receiving events [103]–[106]. In [103] and [104], the authors extended the PTP protocol by adding probe bursts after the operation of the PTP protocol in order to approximate the upstream and downstream data rates and so the end-to-end delay. However, their proposals, resulting in improved synchronization, were only investigated over single hop connections and will be difficult to implement over multi-hop paths. In a similar way, the authors in [105] proposed to duplicate *Sync* messages to find the delay over multiple intermediate nodes without a transparent clock. A synchronization accuracy of approximately 1 ms was obtained even with a delay jitter of 2-8 ms. However, such a scheme cannot mitigate asymmetric delays as only the delay from the master to slave nodes were probed. Also, probing messages were added before and after *Sync* and *Delay_req* messages in [106] in order to detect line utilization, which was shown to improve the synchronization accuracy from $\pm 55$ $\mu$s to 1 $\mu$s.

*5) Servo Clock (Software):* Recall from Eq. (3) that a given slave clock relatively to its associated master clock is characterized by a rate (or frequency) and offset deviation [107]. The slave nodes can implement servo clocks locally to estimate these two components. In recent works, different strategies have been used to develop robust servo clocks, since software-only implementations are more cost-effective compared to hardware-based improvement mechanisms [108]. One technique to estimate the relative clock rate is the use of a Kalman filter [108], [109]. Kalman filters are widely used in telecommunications to estimate unknown variables given a set of observed measurements and random noise variables. In [108], the authors proposed a model based on the least-squares method by considering a set of inter-arrival times and assuming that synchronization messages are exchanged at exactly every second. Using such a technique, the authors noted an offset on the order of 1 $\mu$s with bandwidth usage 1-90 %, thus providing tolerance to cross-traffic. Further, this method was shown to provide improved convergence time (the time taken to stabilize the synchronization accuracy after the initialization phase) compared to the PTP standard. Note that for precise synchronization, the clock rate can be periodically adjusted using an adder-based clock. For instance in [110], the tuning resolution of the adder-based clocks was set to 10 ns.

*6) Timestamp (Software):* Apart from the stability of the clock oscillators and variable network delays, the quality of the timestamping method (to obtain times $T_1, T_2, T_3, T_4$ in NTP and PTP) plays a crucial role in the achievable accuracy [111]. The timestamping methods can be either software or hardware based. A software based timestamping method is executed in a given application with the assistance of the operating system features, while a hardware based method is operated at the MAC or physical layer at the network interface in order to timestamp prior to forwarding the frame to the communications channel. It is worth noting that software clocks tend to be less stable and reliable as multiple impairments are experienced. To overcome this problem, Ridoux and Veitch [112], [113] and Pásztor and Veitch [114] proposed to use the central processing unit (CPU) oscillator, which provides a stable rate and can be accessed via the time stamp counter (TSC) register efficiently. The stable rate property provides more precise timestamps. By exploiting the CPU oscillator, they proposed TSCclock to significantly improve the synchronization performance without hardware support. In a two hops network [112], the synchronization accuracy could be improved to the order of microsecond resolution, as opposed to the millisecond synchronization performance of NTP.

### C. Hardware-Based Improvement Mechanisms

Even though software-based improvement approaches are more cost-effective, hardware-based improvement mechanisms are necessary to further improve the synchronization performance, but at the expense of an increased cost. For microsecond and sub-microsecond synchronization requirements, hardware support is generally used.

*1) Servo Clock (Hardware):* Meier *et al.* [58] realized the PTP functions entirely in hardware in order to meet sub-microsecond synchronization performance. The developed field-programmable gate array (FPGA) takes into account the residence times, timestamp, and syntonization (clock correction via a servo loop). By implementing everything in hardware, including the syntonization function, a maximum offset of 40 ns was measured. Note that synchronization is often used in low-power devices, such as sensors and actuators. In these devices, typically periodic events occur, whereby

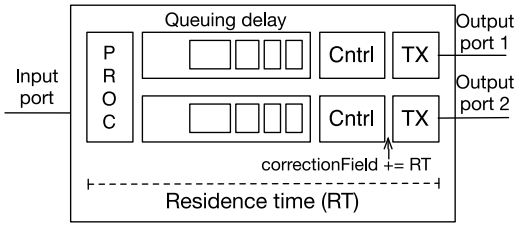



Fig. 12. Router with PTP transparent/boundary clock support.

ideally a sleep mode mechanism operates between events to decrease power consumption [115], [116]. In such devices, a servo clock becomes key to keep synchronizing to a master node without exchanging extensive messages. However, a servo clock can significantly increase the energy consumption. In [117], the authors proposed a deep-sleep mode to correct the clock offset and maintain low power usage.

*2) Timestamp (Hardware):* The IEEE 1588 standard recommends the use of hardware assisted timestamps to reduce time errors due to the timestamping function [53], [118]. Typically, as PTP is an application layer standard, the timestamps are copied in the PTP messages at the application level. However, this approach of timestamping messages cannot provide precise timestamps due to many factors. First, the operating system frequently interrupts application processes for dynamic resource allocation (e.g., random access memory (RAM), CPU, etc.). More significantly, the transport, network, and MAC layers all add variable delays. For instance, frames are buffered at the MAC layer prior to transmission, and the experienced queuing delay is variable depending on the traffic load. Therefore, a significant time difference can occur between the timestamp creation and message transmission on the communications medium. By using hardware timestamping, the timestamp is injected in the frame just before it is forwarded to the communications medium using a media independent interface (MII) as described in [119], thus significantly improving the timestamping accuracy. According to the measurements in [119], a physical timestamping device has an error deviation of 2-16 ns.

*3) Residence Time:* As discussed in the previous section, transparent and boundary clocks allow one to measure residence times at intermediate nodes. This mechanism, requiring hardware support, is part of the IEEE 1588 recommendation for microsecond synchronization accuracy. Fig. 12 depicts a switch/router (with one input port and two output ports) supporting the residence time measurement mechanism [120]. Basically, one timestamp $t_1$ is created when the packet arrives in the router from the input port and another one $t_2$ just before the forwarding process is done at the output port, whereby the residence time equals $RT = t_2 - t_1$. By doing so, the processing, queuing, control, and transmission delays are summed and can be used to improve the synchronization performance. The residence time mechanism is also used in White Rabbit (WR) synchronization systems.

## VII. Security

Security is of increasing importance for systems where time is critical, since malicious attacks to the clock synchronization service could cause damage and issues to the equipment and service reliability [46], [121]. For example, in [122], the authors showed that spoofing the GNSS signal at smart grid synchrophasors can result in deteriorated transmission line fault detection, incorrect voltage stability monitoring and poor event localization.

In general, the security of synchronization protocols can be addressed in either an end-to-end or hop-by-hop fashion. In the end-to-end approach, secured packets are exchanged between the source and destination nodes without any modification at intermediate nodes, where the destination node authenticates the source of the packets by verifying an integrity field. In the hop-by-hop security approach, the source, destination, and intermediate nodes share the encryption key, thus allowing the intermediate nodes to encrypt/decrypt or re-authenticate modified packets, which is required for instance at transparent clocks to update the correction field in PTP.

### A. Security Threats and Requirements

The IETF TICTOC working group published an RFC discussing the threats and defining the security requirements for time protocols, focusing on NTP and PTP [123].

*1) Threats:* Attackers can be categorized as internal, external, man in the middle (MITM), and packet injector. Internal attackers have access to a certain part of the network or own cryptographic keys, while external attackers do not have access to the keys or part of the network, but have access to the traffic. MITM attackers are able to intercept and change in-transit packets, while the traffic injector attackers create attacks by generating packets. The main threats which can be orchestrated by an attacker are described in the following:

- *Packet manipulation:* In such attack, an MITM attacker receives and modifies packets and forwards them to the destination.
- *Spoofing:* A spoofing attacker is able to act as a legitimate node and generate protocol packets, typically by acting as the master.
- *Replay attack:* In a replay attack, an attacker records packets and replays them at a later time without any modification.
- *Rogue master attack:* Similar to the spoofing attack, a rogue master attacker acts as a legitimate master, but as opposed to spoofing, it changes the behavior of the master election process.
- *Packet interception and removal:* In this attack an MITM drops time protocol packets to disrupt the service.
- *Packet delay manipulation:* The packet delay manipulation attack is active when an MITM attacker receives packets and relays them after an added unnecessary delay.
- *Layer 2/3 denial of service (DoS) attack:* An attacker compromises the service availability by flooding for instance the MAC layer.
- *Cryptographic performance attack:* An attacker generates a large amount of synchronization protocol packets with invalid encrypted parameters.
- *DoS attack against the synchronization protocol:* An attacker generates excessive synchronization protocol

TABLE VII
SYNCHRONIZATION THREATS SUMMARY [123]

| *Attack* | *Impact* | | | *Attacker type* | | | |
|---|---|---|---|---|---|---|---|
| | **False time** | **Accuracy degradation** | **DoS** | **MITM (Internal)** | **Injection (Internal)** | **MITM (External)** | **Injection (External)** |
| **Manipulation** | ✓ | | | ✓ | | | |
| **Spoofing** | ✓ | | | ✓ | ✓ | | |
| **Replay attack** | ✓ | | | ✓ | ✓ | | |
| **Rogue master attack** | ✓ | | | ✓ | ✓ | | |
| **Interception and removal** | | ✓ | ✓ | ✓ | | ✓ | |
| **Packet delay manipulation** | ✓ | | | ✓ | | ✓ | |
| **Layer 2/3 DoS attack** | | | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Cryptography performance attack** | | | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Time protocol DoS attack** | | | ✓ | ✓ | ✓ | | |
| **Master time source attack** | ✓ | | | ✓ | ✓ | ✓ | ✓ |

packets in order to degrade the source or destination performance.

- *Master time source attack:* As the grandmaster relies on an external source of time, an attacker can attack the time source, for example by jamming the GNSS signal. It is worth noting that GNSS is very susceptible to jamming and, with the advances in Software Defined Radio (SDR), increasingly sophisticated spoofing attacks are more likely to occur.

The security threats are summarized in Table VII, which are categorized depending on the attacker type. We observe by looking at the internal and external categories that the number of security threats can be significantly decreased by using authentication techniques and secured encryption keys.

*2) Requirements:* According to [123], the security requirements which must be provided are:

- Clock identity authentication and authorization.
- Authentication and authorization of the control messages.
- Protocol packet integrity.
- Must support either end-to-end security or hop-by-hop security.
- Spoofing prevention.
- Replay protection.
- Key freshness and security association.
- Protection against packet delay and interception attacks.

It is worth noting that DoS attacks and performance protection are not required at this time.

Attempts at securing time synchronization have begun to appear, IEEE 1588 contains an annex aimed at protecting PTP packets [53, Annex K]. As PTP is often deployed in networking infrastructures that support multiple applications also requiring security, several recent works proposed to use Internet Protocol Security (IPsec) rather than the IEEE 1588 Annex K to secure PTP packets [124], [125].

### B. Annex K of IEEE 1588

The IEEE 1588 provides an experimental extension for secure exchange of PTP packets between master and slave nodes. Fig. 13a) depicts the protocol stack structure using Annex K of IEEE 1588, whereby an overhead of 30 bytes is used for security fields [124]. More specifically, it provides the following security features [126]:

- *Authentication:* The nodes are authenticated following a challenge request/response protocol [127]. HMAC-SHA256 and HMAC-SHA1-96 symmetric hash functions were selected to generate the integrity check values (ICVs).
- *Integrity:* Man-in-the-middle attacks are avoided by including authentication and ICV in all exchanged PTP packets.
- *Replay protection:* Replay attacks are prevented via the verification of sequence counters.

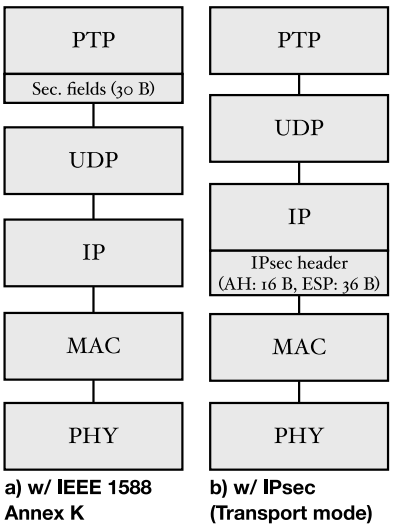



Fig. 13. Protocol structures of PTP with Annex K and IPsec.

Note that confidentiality is not covered in Annex K of IEEE 1588 [126]. Therefore, PTP packets need not be encrypted, which significantly reduces the security computational overhead. It is worth noting that the mechanisms adopted by Annex K to secure PTP packets should not interfere with the time functionality performance. As we outlined in the previous section, delay and jitter play a key role on the synchronization performance. There exists thus a trade-off between the synchronization performance and security [128]. The following overhead issues and solutions were outlined in [128]:

- *Initialization phase:* At startup, when no security association is established, a given master node sending a multicast *Sync* message to the slaves will cause a generation of at least three authentication messages for each slave node in a short period of time, which can significantly increase the network load.

- *Computation load:* It is worth noting that a slave node often has limited embedded computation capabilities (e.g., sensors and actuators), and increasing the synchronization interval and security overhead might decrease the synchronization accuracy due to increased CPU utilization.
- *Correction field update:* At each intermediate node having a transparent clock, the correction field for the residence time must be updated and the ICV needs to recalculated. To avoid this issue, special hardware can be used to provide on-the-fly recomputation security support.

### C. Internet Protocol Security (IPSec)

Annex K of IEEE 1588 provides authentication, integrity, and replay protection services to prevent PTP attacks. However, as multiple applications might share the links used for exchanging PTP packets, having a security framework specifically for PTP might not be the most efficient protocol architecture. Rather, several works proposed to secure PTP packets using a network layer security scheme, such as IPsec, providing security to all applications transparently [124], [125]. IPsec is embedded in IP with an extra IPsec header, as depicted in Fig. 13b). It allows two operation functions: (*i*) Authentication only and (*ii*) Encapsulating Security Payload (ESP). The authentication function offers similar services to Annex K of IEEE 1588; a cryptographic ICV for integrity and a sequence number mechanism to protect against replay attacks. The ESP function, however, further extends the authentication mode services by providing the confidentiality of the data with encrypted payloads.

The authors of [124] and [125] pointed out that intermediate nodes with either transparent or boundary clocks can significantly decrease the PTP accuracy with IPsec, since these intermediate nodes cannot operate actively in the synchronization protocol (e.g., by summing the residence times). However, such an architecture provides secure time synchronization at low cost. Notice that changing a timestamp requires recalculation of the frame check sequence (FCS) and UDP checksum. To mitigate variable delays and jitters, resulting in poor synchronization performance, MAC and hardware timestampers can be configured at transparent clocks [124].

### D. Wireless Sensor Networks

Providing secure time synchronization in wireless sensor networks (WSNs) is a challenging requirement as wireless sensors have constrained resources [129]. A generic attacker model considered in [130] consists of an attacker able to control the shared communications channel by modifying, inserting, eavesdropping, and blocking messages sent between two sensors [130]. There exist three main attack types in regard to the synchronization control messages: (*i*) Modification of the timestamps, (*ii*) Forging and replay attacks, and (*iii*) Delayed message transmission [130]. The two first attack types can be avoided by using conventional security mechanisms providing integrity and authentication. However, the third type of attack is challenging, since both sender and receiver nodes usually are not aware of the exact delays experienced in the network. Such attack, referred to as *pulse-delay attack* consists of jamming an incoming message and then replay it with a delay [129], [130]. The proposed solutions to mitigate this issue are based on delay thresholds estimations; the synchronization procedure is considered only when the calculated delay is close to the estimated minimum and maximum delays interval [129], [130]. However, such a solution might not be practical in industrial systems, since for each technology and configuration, one would need to measure the delay.

## VIII. Conclusion

In this survey, we discussed clock synchronization over packet-switched networks for telecommunications and industrial applications. Although GNSS, atomic clocks, dedicated TDM channel, or timing equipment (e.g., IRIG) could be used to synchronize clocks, these approaches are not currently economically viable to synchronize distributed nodes in large-scale networks. We covered the major applications requiring robust synchronization, including telecommunications, industrial automation, and intelligent power grid and discussed their timing requirements. Next, we surveyed the main synchronization standards, namely NTP, PTP, SyncE, and White Rabbit. We outlined a trade-off between cost and synchronization accuracy. NTP is cost-effective, easy to implement and deploy, but can only provide millisecond accuracy. On the other hand, PTP typically requires more deployment effort and cost, but can meet microsecond requirements. We further overviewed the synchronization protocols designed for WSNs, which aim at reducing the energy consumption by adapting the number of synchronization messages. For all packet-based synchronization protocols, a key challenge is to avoid asymmetric end-to-end delays. These asymmetric delays occur due to variable queuing, transmission, and processing delays. To overcome this major problem, transparent clocks were introduced in the PTP standard, which measures the processing, routing, and queuing delays to correct asymmetric delays. We also categorized the accuracy improvement mechanisms, which can be classified into configuration-, software-, and hardware-based techniques. Software- and configuration-based approaches using the PTP standard are generally more cost-effective as no new hardware equipment is required, and can provide microsecond synchronization performance. However, for highly precise synchronization performance, hardware support is required for precise timestamping and to take into account the residence times. Also, security is a major concern for protecting the timing messages against malicious attackers which could not only disrupt services, but even cause damage and failure to the equipment. Integrity and authenticity are the two main required security attributes which can be provided by the Annex K of PTP or via IPsec.

While synchronization is a widely investigated research topic, their are several open issues for next-generation applications requiring robust synchronization over packet-switched networks:

- *Cost-effective synchronization over multi-hop asymmetric connections.* The current PTP solution for reliable and precise synchronization over multiple hops consists of

measuring the residence time at each node with special hardware support. This solution is economically not viable in large-scale networks and especially on low-power devices. The proposed solutions to mitigate asymmetry in a cost-effective manner are mainly based on delay estimation or probing. Estimation delay techniques can be interesting for single hop networks having simple delay-throughput performance profile. However, in the context of heterogeneous and multi-hop networks, such an approach becomes non viable and challenging to deploy. The probing-based proposals have interesting potentials, but were only investigated under simplistic conditions (single-hop with low network loading).

- *Security and synchronization on devices with limited resources.* Sensors and actuators, used in industrial automation and smart grid applications have limited computing and power capabilities as well as long lifetimes. The synchronization service should use few computing resources and provide high accuracy in a secure manner. This is especially needed in devices supporting real-time control of physical systems. Cryptography algorithms usually require significant resources in terms processing and extra bandwidth utilization [131]. There exists thus a challenging trade-off in this context between resource utilization and security robustness.

## REFERENCES


[1] K. V. Katsaros *et al.*, "Information-centric networking for machine-to-machine data delivery: A case study in smart grid applications," *IEEE Netw.*, vol. 28, no. 3, pp. 58–64, May/Jun. 2014.

[2] NIST Cyber-Physical Systems Public Working Group (CPS PWG). (Jan. 2015). *NIST CPS PWG Webinar Update*. [Online]. Available: http://www.cpspwg.org/Portals/3/docs/Resources/NIST%20CPS%20PWG%20update%20webinar%2015Jan2015%20final.pdf

[3] M. Weiss *et al.*, "Time-aware applications, computers, and communication systems (TAACCS)," Nat. Inst. Stand. Technol., U.S. Dept. Commerce, Gaithersburg, MD, USA, NIST Tech. Rep. 1867, Feb. 2015.

[4] D. Ilic, S. Karnouskos, and P. G. Da Silva, "Sensing in power distribution networks via large numbers of smart meters," in *Proc. IEEE PES Int. Conf. Exhibit. Innov. Smart Grid Technol. (ISGT Europe)*, Berlin, Germany, 2012, pp. 1–6.

[5] D. Bladsjö, M. Hogan, and S. Ruffini, "Synchronization aspects in LTE small cells," *IEEE Commun. Mag.*, vol. 51, no. 9, pp. 70–77, Sep. 2013.

[6] "Series G: Transmission systems and media, digital systems and networks–definitions and terminology for synchronization in packet networks," Int. Telecommun. Union (ITU), Geneva, Switzerland, ITU-T G.8260, Feb. 2012.

[7] T. Schmid *et al.*, "On the interaction of clocks, power, and synchronization in duty-cycled embedded sensor nodes," *ACM Trans. Sensor Netw.*, vol. 7, no. 3, p. 24, 2010.

[8] G. Giorgi and C. Narduzzi, "Modeling and simulation analysis of PTP clock servo," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 155–161.

[9] R. Exel, T. Bigler, and T. Sauter, "Asymmetry mitigation in IEEE 802.3 Ethernet for high-accuracy clock synchronization," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 3, pp. 729–736, Mar. 2014.

[10] E. Corell, P. Saxholm, and D. Veitch, "A user friendly TSC clock," in *Proc. Passive Act. Meas. Conf.*, Adelaide, SA, Australia, 2006, pp. 141–150.

[11] A. Pathak, H. Pucha, Y. Zhang, Y. C. Hu, and Z. M. Mao, "A measurement study of Internet delay asymmetry," in *Passive and Active Network Measurement*. Heidelberg, Germany: Springer, 2008, pp. 182–191.

[12] J.-H. Choi and C. Yoo, "Analytic end-to-end estimation for the one-way delay and its variation," in *Proc. IEEE 2nd Consum. Commun. Netw. Conf.*, Las Vegas, NV, USA, 2005, pp. 527–532.

[13] R. Sugihara and R. K. Gupta, "Clock synchronization with deterministic accuracy guarantee," in *Wireless Sensor Networks* (LNCS 6567), P. J. Marrón and K. Whitehouse, Eds. Heidelberg, Germany: Springer, 2011, pp. 130–146. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-19186-2_9

[14] C. Na, D. Obradovic, R. L. Scheiterer, G. Steindl, and F.-J. Goetz, "Synchronization performance of the precision time protocol," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 25–32.

[15] S. Schriegel and J. Jasperneite, "Investigation of industrial environmental influences on clock sources and their effect on the synchronization accuracy of IEEE 1588," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 50–55.

[16] Z. Li, D. C. Dimitrova, T. Braun, and D. Rosário, "Highly accurate evaluation of GPS synchronization for TDOA localization," in *Proc. IFIP Wireless Days (WD)*, Valencia, Spain, Nov. 2013, pp. 1–3.

[17] W. Lewandowski and C. Thomas, "GPS time transfer," *Proc. IEEE*, vol. 79, no. 7, pp. 991–1000, Jul. 1991.

[18] A. Magee, "Synchronization in next-generation mobile backhaul networks," *IEEE Commun. Mag.*, vol. 48, no. 10, pp. 110–116, Oct. 2010.

[19] L. Cosart, "Precision packet delay measurements using IEEE 1588v2," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 85–91.

[20] "Timing and synchronization for LTE-TDD and LTE-advanced mobile networks," Symmetricom, San Jose, CA, USA, Tech. Rep. WP_TIMESYNCLTE/080313, 2013. Accessed on Jan. 23, 2015. [Online]. Available: https://www.aventasinc.com/whitepapers/WP-Timing-Sync-LTE-SEC.pdf

[21] M. Ouellette, G. Garner, and S. Jobert, "Simulations of a chain of telecom boundary clocks combined with synchronous Ethernet for phase/time transfer," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Munich, Germany, 2011, pp. 105–113.

[22] P. Donegan, "IEEE 1588 timing for mobile backhaul: The road to 'plug & play,"' Heavy Reading, New York, NY, USA, Tech. Rep., 2011. Accessed on Jan. 23, 2015. [Online]. Available: http://www.portals.aviatnetworks.com/exLink.asp?11527326OE13W14I75165336

[23] D. Latremouille, K. Harper, and R. Subrahmanyan, "An architecture for embedded IEEE 1588 support," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 128–133.

[24] R. Subrahmanyan, "Implementation considerations for IEEE 1588v2 applications in telecommunications," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 148–154.

[25] M. Pinchas and R. Cohen, "A combined PTP and circuit-emulation system," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 143–147.

[26] J.-L. Ferrant *et al.*, "Development of the first IEEE 1588 telecom profile to address mobile backhaul needs," *IEEE Commun. Mag.*, vol. 48, no. 10, pp. 118–126, Oct. 2010.

[27] K. Harris, "An application of IEEE 1588 to industrial automation," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 71–76.

[28] H. Mach, E. Grim, O. Holmeide, and C. Calley, "PTP enabled network for flight test data acquisition and recording," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 110–115.

[29] M. E. Glass and K. F. O'Donoghue, "Navy shipboard time synchronization service options and analysis," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 105–109.

[30] D. M. Anand, D. Sharma, Y. Li-Baboud, and J. Moyne, "EDA performance and clock synchronization over a wireless network: Analysis, experimentation and application to semiconductor manufacturing," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–6.

[31] IEEE-SA Standards Board, *IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation With the Electric Power System (EPS), End-Use Applications, and Loads*, IEEE Standard 2030-2011, pp. 1–126, Sep. 2011.

[32] W. Wang, Y. Xu, and M. Khanna, "A survey on the communication architectures in smart grid," *Comput. Netw.*, vol. 55, no. 15, pp. 3604–3629, Oct. 2011.

[33] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 5–20, 1st Quart. 2013.

[34] J. Shannon, H. Melvin, R. O. hOgartaigh, and A. Ruzzelli, "Synchronisation challenges within future smart grid infrastructure," in *Proc. 1st Int. Conf. Smart Grids Green Commun. IT Energy Aware Technol.*, Mestre, Italy, 2011, pp. 22–27.

[35] P. Ferrari, A. Flammini, S. Rinaldi, M. Rizzi, and E. Sisinni, "Time of arrival estimation in power line communication systems for home smart grids," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 83–88.

[36] A. Flammini, S. Rinaldi, and A. Vezzoli, "The sense of time in open metering system," in *Proc. IEEE Int. Conf. Smart Meas. Future Grids (SMFG)*, Bologna, Italy, 2011, pp. 22–27.

[37] F. Fujikawa, "Evaluation of applicability to WAMPAC (wide area monitoring protection and control) of IEEE 1588," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Tainan, Taiwan, 2012, pp. 593–598.

[38] D. M. Anand, J. G. Fletcher, Y. Li-Baboud, J. Amelot, and J. Moyne, "Using clock accuracy to guide model synthesis in distributed systems: An application in power grid control," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 7–12.

[39] J. Amelot *et al.*, "An IEEE 1588 time synchronization testbed for assessing power distribution requirements," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 13–18.

[40] F. Steinhauser, C. Riesch, and M. Rudigier, "IEEE 1588 for time synchronization of devices in the electric power industry," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 1–6.

[41] G. S. Antonova *et al.*, "Standard profile for use of IEEE std 1588-2008 precision time protocol (PTP) in power system applications: IEEE PES PSRC working group H7/Sub C7 members and guests," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, San Francisco, CA, USA, 2012, pp. 1–6.

[42] M. Lixia, C. Muscas, and S. Sulis, "Application of IEEE 1588 to the measurement of synchrophasors in electric power systems," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–6.

[43] J.-C. Tournier and X. Yin, "Improving reliability of IEEE1588 in electric substation automation," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 65–70.

[44] H. Xin, L. Wenmeng, Y. Song, Z. Daonong, and D. Qiwei, "Smart substation IEC61588 time synchronization system and security evaluation," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Austin, TX, USA, 2014, pp. 97–101.

[45] D. M. E. Ingram, P. Schaub, D. A. Campbell, and R. R. Taylor, "Evaluation of precision time synchronisation methods for substation applications," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, San Francisco, CA, USA, 2012, pp. 1–6.

[46] "The emerging interdependence of the electric power grid & information and communication technology," Pac. Northwest Nat. Lab., U.S. Dept. Energy, Washington, DC, USA, Tech. Rep. PNNL-24643, 2015.

[47] T. Koskiahde, J. Kujala, and T. Norolampi, "A sensor network architecture for military and crisis management," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 110–114.

[48] P. V. Estrela, S. Neusüß, and W. Owczarek, "Using a multi-source NTP watchdog to increase the robustness of PTPv2 in financial industry networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Austin, TX, USA, 2014, pp. 87–92.

[49] X. Wu, M. C. Chan, A. L. Ananda, and C. Ganjihal, "Sync-TCP: A new approach to high speed congestion control," in *Proc. IEEE Int. Conf. Netw. Protocols*, Princeton, NJ, USA, 2009, pp. 181–192.

[50] D. L. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

[51] M. Ullmann and M. Vögeler, "Delay attacks—Implication on NTP and PTP time synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–6.

[52] T. Neagoe, V. Cristea, and L. Banica, "NTP versus PTP in computer networks clock synchronization," in *Proc. IEEE Int. Symp. Ind. Electron.*, Montreal, QC, Canada, 2006, pp. 317–362.

[53] IEC Technical Committee 65 and IEEE Standards Association (IEEE-SA) Standards Board, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard 1588(E)-2008, 2008.

[54] E. Y. Song and K. Lee, "An application framework for the IEEE 1588 standard," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 23–28.

[55] D. Mohl and M. Renz, "Improved synchronization behavior in highly cascaded networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 96–99.

[56] G. Gaderer, S. Rinaldi, and N. Kerö, "Master failures in the precision time protocol," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 59–64.

[57] S. Meier and H. Weibel, "IEEE 1588 applied in the environment of high availability LANs," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 100–104.

[58] S. Meier, H. Weibel, and K. Weber, "IEEE 1588 syntonization and synchronization functions completely realized in hardware," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 1–4.

[59] D. Rosselot, "Simple, accurate time synchronization in an Ethernet physical layer device," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 123–127.

[60] N. Simanic, R. Exel, P. Loschmidt, T. Bigler, and N. Kerö, "Compensation of asymmetrical latency for Ethernet clock synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Munich, Germany, 2011, pp. 19–24.

[61] R. Zarick, M. Hagen, and R. Bartoš, "The impact of network latency on the synchronization of real-world IEEE 1588-2008 devices," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 135–140.

[62] "PTP profile for frequency distribution without timing support from the network," Int. Telecommun. Union (ITU), Geneva, Switzerland, ITU-T G.8265.1, Jun. 2010.

[63] S. Kehrer, O. Kleineberg, and D. Heffernan, "A comparison of fault-tolerance concepts for IEEE 802.1 time sensitive networks (TSN)," in *Proc. IEEE Emerg. Technol. Factory Autom. (ETFA)*, Barcelona, Spain, 2014, pp. 1–8.

[64] *IEEE Draft Standard for a Transport Protocol for Time Sensitive Applications in a Bridged Local Area Network*, IEEE Standard P1722-rev1/D13, pp. 1–217, Apr. 2015.

[65] J.-C. Tournier and K. Weber, "Differences and similarities between the audio video bridges and power system profiles for IEEE 1588," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 19–24.

[66] M. D. J. Teener and G. M. Garner, "Overview and timing performance of IEEE 802.1AS," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 49–53.

[67] International Telecommunication Union, Telecommunication Standardization Sector, "Timing characteristics of synchronous Ethernet equipment slave clock (EEC)," ITU-T Rec. G.826/Y.1362, Jan. 2015.

[68] J. Aweya, "Implementing synchronous Ethernet in telecommunication systems," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1080–1113, 2nd Quart. 2014.

[69] J.-L. Ferrant *et al.*, "Synchronous Ethernet: A method to transport synchronization," *IEEE Commun. Mag.*, vol. 46, no. 9, pp. 126–134, Sep. 2008.

[70] S. Rodrigues, "IEEE-1588 and synchronous Ethernet in telecom," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 138–142.

[71] "Synchronous Ethernet explained," ALBEDO Telecom, Barcelona, Spain, Tech. Rep., 2012. Accessed on Jan. 23, 2015. [Online]. Available: http://www.albedotelecom.com/src/lib/WP-SynE-explained.pdf

[72] J. Preston *et al.*, "Novel timing method using IEEE 1588 and synchronous Ethernet for Compton telescope," in *Proc. IEEE Nucl. Sci. Symp. Conf. Record*, Knoxville, TN, USA, 2010, pp. 1404–1407.

[73] M. Lipiński, T. Wlostowski, J. Serrano, and P. Alvarez, "White rabbit: A PTP application for robust sub-nanosecond synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Munich, Germany, 2011, pp. 25–30.

[74] M. Lipinski *et al.*, "Performance results of the first White Rabbit installation for CNGS time transfer," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, San Francisco, CA, USA, 2012, pp. 1–6.

[75] C. Prados, "White Rabbit transparent clock," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 13–17.

[76] J. Shannon, "Improved techniques for time synchronisation over WiFi and wireless sensor networks," Ph.D. dissertation, Dept. Inf. Technol., NUI Galway, Galway, Ireland, 2013.

[77] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 281–323, 2005.

[78] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Oper. Syst. Rev.*, vol. 36, pp. 147–163, 2002.

[79] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sensor Syst.*, Rome, Italy, 2003, pp. 138–149.

[80] M. L. Sichitiu and C. Veerarittiphan, "Simple, accurate time synchronization for wireless sensor networks," in *Proc. Wireless Commun. Netw. (WCNC)*, vol. 2. New Orleans, LA, USA, 2003, pp. 1266–1273.

[81] J. Van Greunen and J. Rabaey, "Lightweight time synchronization for sensor networks," in *Proc. 2nd ACM Int. Conf. Wireless Sensor Netw. Appl.*, San Diego, CA, USA, 2003, pp. 11–19.

[82] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst.*, Baltimore, MD, USA, 2004, pp. 39–49.

[83] T. Schmid, Z. Charbiwala, Z. Anagnostopoulou, M. B. Srivastava, and P. Dutta, "A case against routing-integrated time synchronization," in *Proc. 8th ACM Conf. Embedded Netw. Sensor Syst.*, Zürich, Switzerland, 2010, pp. 267–280.

[84] S. Ganeriwal, D. Ganesan, H. Shim, V. Tsiatsis, and M. B. Srivastava, "Estimating clock uncertainty for efficient duty-cycling in sensor networks," in *Proc. 3rd Int. Conf. Embedded Netw. Sensor Syst.*, San Diego, CA, USA, 2005, pp. 130–141.

[85] T. Schmid, Z. Charbiwala, R. Shea, and M. B. Srivastava, "Temperature compensated time synchronization," *IEEE Embedded Syst. Lett.*, vol. 1, no. 2, pp. 37–41, Aug. 2009.

[86] M. H. Perrott *et al.*, "A temperature-to-digital converter for a MEMS-based programmable oscillator with frequency stability and integrated jitter," *IEEE J. Solid-State Circuits*, vol. 48, no. 1, pp. 276–291, Jan. 2013.

[87] F. S. Lee *et al.*, "A programmable MEMS-based clock generator with sub-ps jitter performance," in *Proc. Symp. VLSI Circuits (VLSIC)*, Honolulu, HI, USA, 2011, pp. 158–159.

[88] J. Shannon and H. Melvin, "A dynamic wireless sensor network synchronisation protocol," Digit. Technol. (DT), College Eng. Informat. NUI Galway, Galway, Ireland, Tech. Rep., 2011. [Online]. Available: http://pel.it.nuigalway.ie/index_files/content/DT11_J_Shannon.pdf

[89] J. Shannon, H. Melvin, and A. G. Ruzzelli, "Dynamic flooding time synchronisation protocol for WSNs," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, Anaheim, CA, USA, 2012, pp. 365–371.

[90] A. Ciuffoletti, "Preventing the collision of requests from slave clocks in the precision time protocol (PTP)," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 6, pp. 2096–2103, Jun. 2011.

[91] A. Mahmood, G. Gaderer, H. Trsek, S. Schwalowsky, and N. Kerö, "Towards high accuracy in IEEE 802.11 based clock synchronization using PTP," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Munich, Germany, 2011, pp. 13–18.

[92] S. Lee, S. Lee, and C. Hong, "An accuracy enhanced IEEE 1588 synchronization protocol for dynamically changing and asymmetric wireless links," *IEEE Commun. Lett.*, vol. 16, no. 2, pp. 190–192, Feb. 2012.

[93] Z. Chaloupka, N. Alsindi, and J. Aweya, "Clock synchronization over communication paths with queue-induced delay asymmetries," *IEEE Commun. Lett.*, vol. 18, no. 9, pp. 1551–1554, Sep. 2014.

[94] M. Hajikhani, T. Kunz, and H. Schwartz, "A recursive method for bias estimation in asymmetric packet-based networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Austin, TX, USA, 2014, pp. 47–52.

[95] H. Beyranvand, M. Lévesque, M. Maier, and J. A. Salehi, "FiWi enhanced LTE-A HetNets with unreliable fiber backhaul sharing and WiFi offloading," in *Proc. IEEE INFOCOM*, 2015, pp. 1275–1283.

[96] A. Mahmood, R. Exel, and T. Sauter, "Delay and jitter characterization for software-based clock synchronization over WLAN using PTP," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1198–1206, May 2014.

[97] P. A. Frangoudis, A. Ksentini, Y. Hadjadj-Aoul, and G. Boime, "PTPv2-based network load estimation," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 101–106.

[98] R. Exel, "Mitigation of asymmetric link delays in IEEE 1588 clock synchronization systems," *IEEE Commun. Lett.*, vol. 18, no. 3, pp. 507–510, Mar. 2014.

[99] E. A. Lee and S. Matic, "On determinism in event-triggered distributed systems with time synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 56–63.

[100] A. Shpiner, Y. Revah, and T. Mizrahi, "Multi-path time protocols," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 1–6.

[101] T. Mizarahi, "Slave diversity: Using multiple paths to improve the accuracy of clock synchronization protocols," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, San Francisco, CA, USA, 2012, pp. 1–6.

[102] G. Giorgi and C. Narduzzi, "Kalman filtering for multi-path network synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Austin, TX, USA, 2014, pp. 65–70.

[103] S. Lee, "An enhanced IEEE 1588 time synchronization algorithm for asymmetric communication link using block burst transmission," *IEEE Commun. Lett.*, vol. 12, no. 9, pp. 687–689, Sep. 2008.

[104] S. Lv, Y. Lu, and Y. Ji, "An enhanced IEEE 1588 time synchronization for asymmetric communication link in packet transport network," *IEEE Commun. Lett.*, vol. 14, no. 8, pp. 764–766, Aug. 2010.

[105] S. Schriegel, H. Trsek, and J. Jasperneite, "Enhancement for a clock synchronization protocol in heterogeneous networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–5.

[106] T. Murakami and Y. Horiuchi, "Improvement of synchronization accuracy in IEEE 1588 using a queuing estimation method," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–5.

[107] R. Exel and F. Ring, "Improved clock synchronization accuracy through optimized servo parametrization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 65–70.

[108] A. Machizawa, T. Iwawma, and H. Toriyama, "Software-only implementations of slave clocks with sub-microsecond accuracy," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 17–22.

[109] H. Abubakari and S. Sastry, "IEEE 1588 style synchronization over wireless link," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 127–130.

[110] N. Kerö, T. Kernen, T. Müller, and M. Deniaud, "Analysis of lock times of PTP slave under varying network load conditions using class based queuing," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Lemgo, Germany, 2013, pp. 18–23.

[111] R. Exel and P. Loschmidt, "High accurate timestamping by phase and frequency estimation," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–6.

[112] J. Ridoux and D. Veitch, "Ten microseconds over LAN, for free," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 105–109.

[113] J. Ridoux and D. Veitch, "The cost of variability," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 29–36.

[114] A. Pásztor and D. Veitch, "PC based precision timing without GPS," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, Marina del Rey, CA, USA, 2002, pp. 1–10.

[115] P. Ferrari, A. Flammini, D. Marioli, E. Sisinni, and A. Taroni, "Non invasive time synchronization for ZigBee wireless sensor networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 121–126.

[116] A. Ageev, D. Macii, and A. Flammini, "Towards an adaptive synchronization policy for wireless sensor networks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 115–120.

[117] M. Buevich, N. Rajagopal, and A. Rowe, "Hardware assisted clock synchronization for real-time sensor networks," in *Proc. IEEE Real Time Syst. Symp. (RTSS)*, Vancouver, BC, Canada, 2013, pp. 268–277.

[118] C. Riesch, C. Marinescu, and M. Rudigier, "Experimental verification of the egress and ingress latency correction in PTP clocks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Austin, TX, USA, 2014, pp. 59–64.
[119] P. Loschmidt, R. Exel, A. Nagy, and G. Gaderer, "Limits of synchronization accuracy using hardware support in IEEE 1588," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 12–16.
[120] D. Köhler, "A practical implementation of an IEEE1588 supporting Ethernet switch," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 134–137.
[121] A. Treytl and B. Hirschler, "Security flaws and workarounds for IEEE 1588 (transparent) clocks," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Brescia, Italy, 2009, pp. 1–6.
[122] Z. Zhang, S. Gong, A. D. Dimitrovski, and H. Li, "Time synchronization attack in smart grid: Impact and analysis," *IEEE Trans. Smart Grid*, vol. 4, no. 1, pp. 87–98, Mar. 2013.
[123] T. Mizrahi, "Security requirements of time protocols in packet switched networks," Internet Eng. Task Force (IETF), Fremont, CA, USA, Request for Comments 7384, Oct. 2014.
[124] A. Treytl and B. Hirschler, "Securing IEEE 1588 by IPsec tunnels—An analysis," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Portsmouth, NH, USA, 2010, pp. 83–90.
[125] T. Mizrahi, "Time synchronization security using IPsec and MACsec," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Munich, Germany, 2011, pp. 38–43.
[126] C. Önal and H. Kirrmann, "Security improvements for IEEE 1588 Annex K: Implementation and comparison of authentication codes," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, San Francisco, CA, USA, 2012, pp. 1–6.
[127] A. Treytl, G. Gaderer, B. Hirschler, and R. Cohen, "Traps and pitfalls in secure clock synchronization," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Vienna, Austria, 2007, pp. 18–24.
[128] A. Treytl and B. Hirschler, "Practical application of 1588 security," in *Proc. IEEE Symp. Precis. Clock Synchronization Meas. Control Commun. (ISPCS)*, Ann Arbor, MI, USA, 2008, pp. 37–43.
[129] C. Benzaid, A. Saiah, and N. Badache, "An enhanced secure pairwise broadcast time synchronization protocol in wireless sensor networks," in *Proc. IEEE Euromicro Int. Conf. Parallel Distrib. Netw. Based Process. (PDP)*, Turin, Italy, 2014, pp. 569–573.
[130] S. Ganeriwal, C. Pöpper, S. Čapkun, and M. B. Srivastava, "Secure time synchronization in sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 11, no. 4, pp. 1–35, Jul. 2008.
[131] D. Maimut and K. Ouafi, "Lightweight cryptography for RFID tags," *IEEE Security Privacy*, vol. 10, no. 2, pp. 76–79, Mar./Apr. 2012.

**Martin Lévesque** received the M.Sc. degree in computer science from UQAM, QC, Canada, in 2010, and the Ph.D. degree in telecommunications from INRS, Université du Québec, in 2014. In summer 2013, he was a Visiting Scholar with EDF Research and Development, Clamart, France, where he researched on advanced smart grid multisimulations. He is currently a Post-Doctoral Researcher with the University of Pittsburgh, Pittsburgh, PA, USA, funded by the prestigious Post-Doctoral Fellowship Program from the Natural Sciences and Engineering Research Council of Canada. He was a recipient of the master and doctoral scholarships while pursuing his graduate studies, and a co-recipient of the Best Paper Award presented at the IEEE International Conference on Design of Reliable Communication Networks in 2015. He served as a Reviewer for numerous major journals. He is a member of the IEEE IES Technical Committee on Smart Grids.

**David Tipper** received the B.S. degree in electrical engineering from Virginia Tech, Blacksburg, VA, USA, in 1980, and the M.S. degree in systems engineering and the Ph.D. degree in electrical engineering from the University of Arizona, Tucson, AZ, USA, in 1984 and 1988, respectively.

He is the Director of the Graduate Telecommunications and Networking Program, and a Professor with the School of Information Sciences, University of Pittsburgh, Pittsburgh, PA, USA. His research was supported by grants totaling over $8.8 million from various government and corporate sources such as the National Science Foundation, the Defense Advanced Research Project Agency, the National Institute of Standards and Technology, the Army Research Office, IBM, and AT&T. He has co-authored a book entitled *The Physical Layer of Communication Systems* (Artech House, 2006), and co-edited and contributed to *Information Assurance: Dependability and Security in Networked Systems* (Morgan Kaufmann, 2008). His current research interests include network design, network reliability performance analysis techniques, and information security.

Dr. Tipper serves as a Co-Guest Editor for a Special Issue on Advances in Network Planning, which appeared in the *IEEE Communications Magazine* in 2014, and Telecommunication Systems on Reliable Networks Design and Modeling in 2013.