PTPsec: Securing the Precision Time Protocol Against Time Delay Attacks Using Cyclic Path Asymmetry Analysis

Andreas Finkenzeller*, Oliver Butowski*, Emanuel Regnath†, Mohammad Hamad*, and Sebastian Steinhorst*
*Technical University of Munich, Germany
†Siemens AG, Germany
Email: *firstname.lastname@tum.de, †firstname.lastname@siemens.com

***Abstract*—High-precision time synchronization is a vital prerequisite for many modern applications and technologies, including Smart Grids, Time-Sensitive Networking (TSN), and 5G networks. Although the Precision Time Protocol (PTP) can accomplish this requirement in trusted environments, it becomes unreliable in the presence of specific cyber attacks. Mainly, time delay attacks pose the highest threat to the protocol, enabling attackers to diverge targeted clocks undetected. With the increasing danger of cyber attacks, especially against critical infrastructure, there is a great demand for effective countermeasures to secure both time synchronization and the applications that depend on it. However, current solutions are not sufficiently capable of mitigating sophisticated delay attacks. For example, they lack proper integration into the PTP protocol, scalability, or sound evaluation with the required microsecond-level accuracy. This work proposes an approach to detect and counteract delay attacks against PTP based on cyclic path asymmetry measurements over redundant paths. For that, we provide a method to find redundant paths in arbitrary networks and show how this redundancy can be exploited to reveal and mitigate undesirable asymmetries on the synchronization path that cause the malicious clock divergence. Furthermore, we propose PTPsec, a secure PTP protocol and its implementation based on the latest IEEE 1588-2019 standard. With PTPsec, we advance the conventional PTP to support reliable delay attack detection and mitigation. We validate our approach on a hardware testbed, which includes an attacker capable of performing static and incremental delay attacks at a microsecond precision. Our experimental results show that all attack scenarios can be reliably detected and mitigated with minimal detection time.**

***Index Terms*—Security, IEEE 1588, PTP, Time Delay Attack, Time Synchronization**

I. Introduction

Precise time synchronization is indispensable for many applications and technologies, such as Smart Grids, Time Sensitive Networking (TSN), and 5G networks. To work correctly, these applications usually require synchronization accuracies on a microsecond or sub-microsecond level, and even small deviations can already have significant impacts. The consequences range from performance degradation to, in the worst case, full system failure. For example, delay attacks in Smart Grids render the control loops unstable, eventually disrupting the entire system [1].

The Precision Time Protocol (PTP) (see § II-A) is a commonly used network protocol to provide accurate time synchronization in the aforementioned scenarios. Unfortunately, PTP was initially designed without any security considerations in mind and, hence, makes certain network assumptions, such as path symmetry and message integrity, that might not always hold. Especially in spatially large networks like Smart Grids, for instance, cyber attacks targeting the network infrastructure are becoming more likely, forcing the protocol designers to reconsider security concepts in PTP. With the four-pronged approach in Annex P of the latest revision from 2019 [2], the PTP standard aims to address this problem. While the suggested cryptographic countermeasures can guarantee message integrity and protect against many other attack vectors, including message replay and spoofing attacks, they cannot counteract all known threats. In particular, time delay attacks (see § II-B) remain an unsolved problem so far, as previous work has extensively shown [3, 4, 5].

Delay attacks exploit and violate the protocol's assumption of symmetric path delays by maliciously introducing unidirectional delays into the network. In the literature, initial solutions to counteract delay attacks against PTP have been proposed recently, none of which provide an effective solution to the problem yet. Threshold-based approaches, for example, as presented in [6, 7], perform the attack detection by means of continuously monitoring the reported path delay and

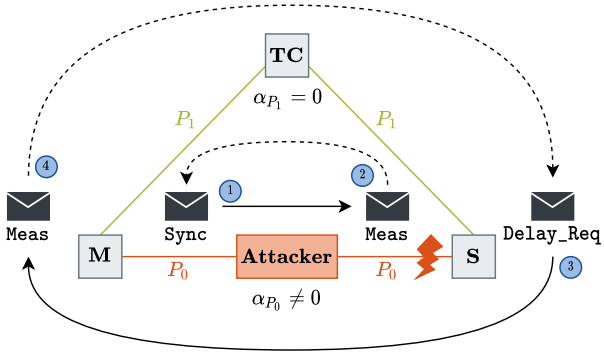


Fig. 1: The path asymmetry analysis concept to detect time delay attacks within our proposed PTPsec protocol. The PTP `Sync` message, sent via the attacked path P_0 (1), is followed by our newly introduced `Meas` message over the genuine path P_1 to complete the first round trip (2). Similarly, the exchange of the `Delay_Req` (3) and another `Meas` message (4) leads to a second circulation. This allows for cyclic RTT measurements from which we derive the current path asymmetry α_{P_0} to reveal and mitigate ongoing delay attacks.

comparing it to a previously defined threshold. However, the threshold definition is quite challenging. Static thresholds must be set very conservatively to avoid false alarms, while dynamic thresholds, which adapt based on previous measurements, cannot adequately protect the system from incremental delay attacks. Moreover, the reported path delay is not guaranteed to change at all during a successful delay attack [8]. Other solutions, including additional network guards [9, 10] or special monitoring and reporting mechanisms [11, 12] only work within a limited attacker model. Because the detection method is not sufficiently coupled to the time synchronization, attackers can distinguish PTP messages from other network traffic and handle them differently to bypass the existing mitigations. Besides, there are first efforts to countermeasures based on path redundancy. However, the available works [8, 13] only present some general ideas that, among other things, lack scalability and proper protocol integration to become applicable. Hence, there is still a great need for a secure and comprehensive solution that entirely protects PTP from time delay attacks.

Contributions: This work investigates cyclic Round-Trip Time (RTT) measurements to analyze network path asymmetries and the applicability for delay attack detection and mitigation in time synchronization protocols. The cyclic analysis is enabled by dedicated measurement packets that we introduce in our proposed PTPsec protocol to entangle the attack detection with the synchronization procedure. Each PTP event message invokes a subsequent measurement packet that is returned to the originator via a redundant path to complement the cyclic RTT measurement as depicted in Fig. 1. To finally mitigate ongoing delay attacks, we analyze the path asymmetry based on the two RTT measurements obtained in one PTP synchronization round and compensate for it accordingly. To the best of our knowledge, PTPsec is the first protocol that efficiently detects and mitigates time delay attacks. In particular, we:

- analyze cyclic RTT measurements and show their effectiveness for reliable path asymmetry identification (§ III),
- derive a theoretical model for delay attack detection in arbitrary networks (§ IV),
- present PTPsec, a secure PTP protocol and its implementation as an extension of the latest IEEE 1588-2019 standard that adopts our proposed attack mitigation method (§ V), and
- validate the PTPsec implementation on a realistic hardware testbed to confirm that our approach successfully detects and mitigates time delay attacks (§ VI).

II. System Model

A. Precision Time Protocol

The precision time protocol, as defined in the IEEE 1588 standard [2], is a network protocol that synchronizes devices with an accuracy of less than 1 µs. Thereby, the standard considers various clock types to account for different device behaviors in common PTP networks. Particularly, Ordinary

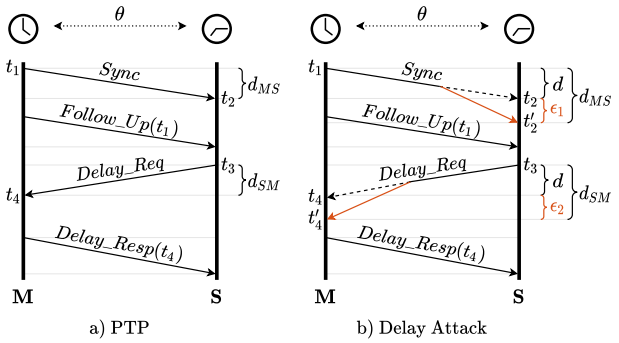


Fig. 2: a) PTP message flow with timestamping to minimize the clock offset θ. b) When attackers can delay PTP event messages (`Sync` or `Delay_Req`), they create path asymmetries that impair the clock synchronization.

Clocks (OCs) denote master and slave nodes[1] which either provide a time reference to others or synchronize their clocks to a specific master, respectively. In contrast, Transparent Clocks (TCs) do not actively synchronize to other clocks but are transparent switching devices that connect multiple OCs to form a large network. The synchronization procedure comprises two phases. In the protocol's first phase, all participating nodes announce their existence and determine the best (i.e., most accurate) clock in the network, denoted the *Grandmaster* clock, using the Best Master Clock Algorithm (BMCA). In the second phase, all other nodes start actively synchronizing their clocks to the *Grandmaster* by periodically exchanging PTP messages. Along with the exchange of `Sync` and `Delay_Req` messages, four timestamps are captured at both the transmission and reception of each packet. Here, the required use of hardware timestamping increases the synchronization accuracy significantly. The two so-called event messages are critical for the achieved synchronization accuracy since the obtained timestamps are the basis for subsequent offset calculations. With the additional assumption of symmetric path delays ($d_{MS} = d_{SM}$), which is an indispensable assumption for PTP, the current clock offset θ can be computed as:

$$\theta = \frac{(t_2 - t_1) - (t_4 - t_3)}{2} \tag{1}$$

where t_1, t_2, t_3, and t_4 denote the captured timestamps of the two event messages. The diagram in Fig. 2a depicts the captured timestamps along the entire message flow of one synchronization round. Note that the `Follow_Up` message conveying the captured transmission timestamp t_1 is only required when PTP operates in two-step mode.

B. Time Delay Attack

However, if the synchronization path is not symmetric because of malicious network activities, for example, the offset calculations are based on an invalid assumption, leading to erroneous synchronization results. So, if attackers were able

[1]In this work, we stick to the terminology used in the official standard to avoid any confusion besides being potentially inappropriate.

to deliberately delay the `Sync` and `Delay_Req` messages by ϵ_1 and ϵ_2 respectively with $\epsilon_1 \neq \epsilon_2$, as illustrated in Fig. 2b, the clock offset θ' would mistakenly compute as:

$$\begin{aligned}\theta' &= \frac{((t_2 + \epsilon_1) - t_1) - ((t_4 + \epsilon_2) - t_3)}{2} \\ &= \frac{(t_2 - t_1) - (t_4 - t_3)}{2} + \frac{\epsilon_1 - \epsilon_2}{2} \\ &= \theta + \frac{\alpha}{2}\end{aligned} \tag{2}$$

with $\alpha = d_{MS} - d_{SM} = \epsilon_1 - \epsilon_2$. Provided $\alpha = 0$, the path is perfectly symmetric, and the time synchronization is successful. However, if $\alpha \neq 0$, the path becomes asymmetric, and the synchronized clock runs behind or ahead in time of the *Grandmaster* clock, respectively. The described time delay attack can be implemented in multiple ways. For example, Barreto et al. [3] present a static delay attack scenario, where the attackers deliberately extend the optical fibers in smart grid networks to add a constant one-way delay, turning the synchronization path asymmetric. In [5], the authors perform an incremental delay attack, where the introduced asymmetric path delay gradually increases, to showcase more dynamic attacks. Both works show the general feasibility of time delay attacks and further emphasize their devastating impact on PTP, highlighting the great need for sound countermeasures.

C. Network Model

A typical PTP deployment comprises an entire network, including many devices that shall be synchronized. Therefore, we model a given PTP network as the graph $\mathcal{G} = (V, E)$ with $V = \{v_0, \ldots, v_n\}$, $n \in \mathbb{N}$ denoting the set of vertices and $E = \{e_0, \ldots, e_m\} \subseteq V \times V$, $m \in \mathbb{N}$ the set of edges. Each vertex $v_i \in V$ represents a node participating in the PTP protocol. For proper synchronization, we require at least the elected *Grandmaster* node and one additional slave device that synchronizes to the master. Thus, we assume our vertex set V to contain these two nodes at the minimum, which we denote for further reference as $M \in V$ for the master and $S \in V$ for the slave node, respectively. All edges $e_i \in E$ are considered bidirectional and model the connecting links between all involved network devices. Furthermore, we assume $\mathcal{G}$ to be connected. Hence, at least one path connects M and S for the PTP message exchange, which we denote as P_0. Also, we extend the notation of α to express the asymmetry α_{e_i} of a specific edge $e_i \in E$. Additionally, we define the path asymmetry α_{P_i}, which is experienced by all traversing packets on path P_i, as the sum of the individual link asymmetries of all composing links:

$$\alpha_{P_i} = \sum_j \alpha_{e_j}, \forall e_j \in P_i. \tag{3}$$

Finally, we assume full control of the packet routing, which can be achieved, for example, by leveraging Software-defined Networking (SDN) capabilities [7].

D. Attacker Model

There exist many attacks against time synchronization protocols, particularly against PTP [14]. In this work, however, we mainly focus on delay attacks and related attacker capabilities due to their criticality. Similar to [3, 5], we deny the attackers internal access to the protocol, i.e., the possibility to directly modify PTP header fields. If this was possible, the attackers were powerful enough to manipulate the synchronization on a protocol level by changing timestamps and the correction field contents, which would render delay attacks superfluous. Security protocols, such as IPsec and MACsec, can be appropriate remedies to ensure message integrity. Furthermore, attackers are not allowed to compromise hosts or intermediary network devices in a way that would grant them internal protocol access or allow them to update the clocks directly with a similar reasoning.

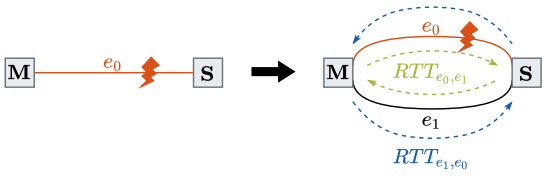
Nevertheless, we assume the attackers to have full knowledge of the network topology, including knowledge about deployed PTP clock types. Additionally, they are fully aware of any implemented protection methods (e.g., IPsec). After compromising a link, the attackers can delay all passing packets individually. More precisely, they can deliberately add unidirectional delays to selected messages in both transmission directions independently. This fine-grained packet delay is even possible with specific protection mechanisms enabled, such as traffic encryption, as shown in [4]. Moreover, the attackers are able to simultaneously perform multiple delay attacks precisely synchronized at different locations in the network if they compromise more than one link. There are no restrictions on the number of compromised links or the strategy of how attackers perform joint attacks. However, we assume that the selection of hijacked links will remain constant for a sufficiently long period to allow for steady-state analysis. Particularly, the attackers may freely change the introduced delay for an attacked link over time but not attack a different link. Finally, we require all nodes that participate in the PTP protocol to be honest, i.e., to follow the protocol as specified in the standard or in our proposed PTPsec protocol, respectively.

III. Asymmetry Analysis

If we could precisely measure unidirectional delays in the network, time delay attacks would be easily detected. However, accurate one-way path delay measurements are a challenging problem in network theory [15]. In the following analysis, we benefit from the work of Gurewitz et al. [16, 17] on one-way delay estimation which we extend to derive our cyclic path asymmetry analysis. Additionally, we propose a general path asymmetry model that forms the basis for the delay attack detection method presented in § IV. First, we start with a simple two-node example from which we develop the general model for arbitrary networks.

A. Two-Node Scenario

Given a simple synchronization scenario with only two nodes M and S and one attacked edge e_0, as shown in Fig. 3a, we are trying to estimate the link asymmetry $\alpha_{e_0} \neq 0$. However, it is impossible to reliably determine α_{e_0} with only



a) One Attacked Link b) Two Links for Cyclic Structure

Fig. 3: Cyclic path asymmetry analysis illustrated with two nodes. While there is no efficient method to calculate the asymmetry with only one link (a), a second link enables a cyclic structure for further analysis (b). Two RTT measurements in opposing directions can be smartly combined to determine the link asymmetry α_{e_0} of the attacked link e_0.

one available link. Thus, we need at least one additional link to obtain a circular structure that we can use for further analysis.

1) One redundant link: If there exists a second edge e_1 which is redundant to e_0, both edges form a cycle, as depicted in Fig. 3b. Now, we can perform a cyclic RTT measurement using e_0 and e_1 by forwarding a packet from M to S on link e_0 which is subsequently returned to M via e_1. Upon arrival at node M, the elapsed time RTT_{e_0,e_1} computes as:

$$RTT_{e_0,e_1} = t_{in} - t_{eg} \tag{4}$$

where t_{in} is the packet's measured ingress and t_{eg} the egress timestamp at node M. Additionally, we conduct a second RTT measurement RTT_{e_1,e_0} with reverse transmission direction, i.e., forwarding the packet on link e_1 to S and returning it via e_0. Both measurements include unidirectional delays of e_0, however, in opposite directions. For RTT_{e_0,e_1}, edge e_0 contributes in forward direction (M to S) while RTT_{e_1,e_0} contains its delay in backward direction (S to M). If we further assume that link e_1 is perfectly symmetric ($\alpha_{e_1} = 0$), i.e., the contributed link delay is equal in both directions, we can calculate the asymmetry of link e_0 as:

$$\alpha_{e_0} = RTT_{e_0,e_1} - RTT_{e_1,e_0} \tag{5}$$

Note that the opposing one-way delays of link e_1 cancel out due to its symmetry and we are left with the desired difference of both unidirectional delays of e_0. Here, the essential part is the cyclic measurement approach capturing the opposing one-way delays of a single link isolated in distinct measurements. Furthermore, the two RTTs measurements originate from the same clock avoiding the need for any time synchronization. Also note that the same measurements could have likewise been taken on node S since the starting point in a cyclic measurement is irrelevant.

2) N redundant links: In the previous example with one redundant edge, the asymmetry measurement only worked because we assumed the additional link e_1 to be symmetric. However, if the second link was also asymmetric due to another delay attack, for example, attackers could thwart the measurement attempt while still successfully introducing unidirectional delays on e_0. For that, they need to add similar one-way delays ϵ to both links e_0 and e_1 but in opposing directions, so that the delays cancel out each other in (5) while the individual links become asymmetric. In such a case, we need another edge e_2 with symmetric link delay for compensation. This additional link increases the number of cycles in our network allowing for further independent cyclic measurements. With two redundant edges, we can perform two asymmetry measurements that include e_0 leading to:

$$\begin{aligned} \alpha^{(1)} &= RTT_{e_0,e_1} - RTT_{e_1,e_0} \\ \alpha^{(2)} &= RTT_{e_0,e_2} - RTT_{e_2,e_0} \end{aligned} \tag{6}$$

with $\alpha^{(1)}$ and $\alpha^{(2)}$ being independent estimates for the targeted link asymmetry α_{e_0}. Although $\alpha^{(1)}$ might equal to zero indicating perfect link symmetry due to the mentioned attack strategy, $\alpha^{(2)}$ will yield the correct estimate because of the demanded link symmetry of e_2 and the reasoning of (5).

Similarly, we can derive the general case of n redundant edges (in addition to e_0) for the two-node scenario. From $2n$ pairwise RTT measurements RTT_{e_0,e_i} and RTT_{e_i,e_0}, $i \in [1,n]$, we get n estimates for the link asymmetry α_{e_0}:

$$\begin{aligned} \alpha^{(1)} &= RTT_{e_0,e_1} - RTT_{e_1,e_0} \\ \alpha^{(2)} &= RTT_{e_0,e_2} - RTT_{e_2,e_0} \\ &\vdots \\ \alpha^{(n)} &= RTT_{e_0,e_n} - RTT_{e_n,e_0} \end{aligned} \tag{7}$$

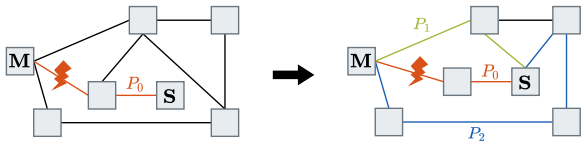
Furthermore, we can state the following important finding:

> **Finding 1:** In order to successfully determine α_{e_0}, we need at least one link e_i, $i \in [1,n]$ with symmetric link delay $\alpha_{e_i} = 0$, which is redundant to e_0. Using this symmetric link e_i, the asymmetry measurement yields a correct estimate $\alpha^{(i)} = RTT_{e_0,e_i} - RTT_{e_i,e_0} = \alpha_{e_0}$.

This necessary condition directly results from the previous reasoning that attackers could introduce various unidirectional delays to cancel out each other in the cyclic measurements if we allowed all links to be asymmetric.

B. Multi-Node Scenario

Realistic networks usually comprise more than two nodes. Therefore, we need to derive a general path asymmetry model that is applicable to networks of any size. Similar to the two-node scenario, we attempt to estimate the asymmetry α_{P_0} on the synchronization path P_0. We equally require at least one redundant path P_1 in addition to P_0, as exemplified in Fig. 4, to enable the cyclic measurements previously proposed in § III-A. Note that P_1 must not share any edge with P_0 to be considered fully redundant since all edges of P_0 are contributing to its total path asymmetry as defined in (3). If a single joint edge existed, it would affect both paths equally and make a cyclic measurement with independent forward and backward delays impossible. Therefore, we need P_0 and P_1 to be edge-disjoint: $P_0 \cap P_1 = \emptyset$. With this requirement, we can formulate the general approach to estimate path asymmetries in arbitrary networks. Given the network $\mathcal{G} = (E, V)$, the



a) One Attacked Path b) Multiple Redundant Paths

Fig. 4: Redundant path principle in multi-node networks. To efficiently estimate the path asymmetry α_{P_0}, we require other edge-disjoint paths P_i to enable cyclic RTT measurements.

synchronization path P_0, and n redundant paths P_i, $i \in [1, n]$ between nodes M and S, we perform $2n$ cyclic RTT measurements for each pair P_0 and P_i, $i \in [1, n]$. From these measurements, we derive the following set of equations to get n estimates for the path asymmetry α_{P_0}:

$$\begin{aligned} \alpha^{(1)} &= RTT_{P_0,P_1} - RTT_{P_1,P_0} \\ \alpha^{(2)} &= RTT_{P_0,P_2} - RTT_{P_2,P_0} \\ &\vdots \\ \alpha^{(n)} &= RTT_{P_0,P_n} - RTT_{P_n,P_0} \end{aligned} \tag{8}$$

Moreover, we state the generalized finding from our path asymmetry analysis in arbitrary networks:

Finding 2: For all symmetric paths P_i, $i \in [1, n]$ with $\alpha_{P_i} = 0$, we get correct estimates $\alpha^{(i)} = RTT_{P_0,P_i} - RTT_{P_i,P_0} = \alpha_{P_0}$. Hence, we require at least one redundant path to be symmetric in order to successfully determine the desired path asymmetry α_{P_0} for the synchronization path P_0.

IV. Attack Detection and Mitigation

A. Detection Criteria

From (2), we know that successful time delay attacks result in non-zero path asymmetries. Hence, we can now use the previously derived path asymmetry model to detect and mitigate these attacks in PTP networks. With $n+1$ total edge-disjoint paths P_i, $i \in [0, n]$ connecting nodes M and S, where P_0 is used for PTP synchronization, we yield n independent asymmetry estimates for α_{P_0}, as stated in (8). Based on these estimates, we define the detection criterion for an ongoing delay attack on path P_0 impeding the time synchronization between M and S as follows:

$$delay\_attack \leftarrow \begin{cases} True, & \exists\, \alpha^{(i)} \neq 0,\ i \in [1, n] \\ False, & \text{otherwise} \end{cases} \tag{9}$$

Additionally, we know that the cyclic measurements only work if at least one path is symmetric. Thus, we can successfully detect delay attacks that include up to n attacked paths. Furthermore, we can try to determine the actual position of the attacked paths by analyzing the estimated path asymmetries $\alpha^{(i)}$ and search for attack configurations that match the obtained result. It turns out that we can cluster all paths into two groups, where one group is either entirely genuine and the other fully attacked or vice versa. However, this decision cannot be made without further assumptions, such as limiting the number of attackers to an upper bound of $\#\,attackers \leq \lfloor \frac{n}{2} \rfloor$.

Algorithm 1 Adapted Ford-Fulkerson Algorithm

Input: Network $\mathcal{G} = (V, E)$, Source $M \in V$, Sink $S \in V$
Output: Set of edge-disjoint paths $\mathcal{P}$ from M to S

1: $paths \leftarrow \emptyset$
2: $flow(e) \leftarrow 0$ **for each** $e \in E$
3: **while** $p \leftarrow$ find M-S path with $flow(e) = 0\ \ \forall e \in p$ **do**
4: $\quad paths$.insert(p)
5: $\quad flow(e) \leftarrow 1$ **for each** $e \in p$
6: **end while**
7: **return** $paths$

B. Attack Mitigation

If an attack is detected, we can furthermore use the measured path asymmetry α_{P_0} of the synchronization path P_0 to mitigate the ongoing attack. For that, we calculate the rectified clock offset θ_{rect} that is compensating malicious path asymmetries using the reported PTP offset θ_{rep} from (1) and the insight gained from (2):

$$\theta_{rect} = \theta_{rep} - \frac{\alpha_{P_0}}{2} \tag{10}$$

This rectified clock offset can be used as input for PTP's control algorithm to securely update the local clock oscillator despite any ongoing attack.

C. Finding Redundant Paths

Another essential aspect of the presented method is finding redundant paths in a given network $\mathcal{G}$ which results to finding edge-disjoint paths to obtain full redundancy. For that, we present an approach to derive all eligible paths by leveraging both Menger's theorem [18] and the Max-Flow-Min-Cut theorem [19]. Menger states there are k pairwise edge-disjoint M-S paths if and only if S is still reachable from M after removing $k-1$ arbitrary edges from the graph [18]. Hence, we are interested in the minimum edge cut k that is required to disconnect the two nodes M and S in $\mathcal{G}$. By introducing the non-negative capacity function $c(e) = 1$, which assigns each edge $e_i \in E$ the maximum capacity of one, this problem is equivalent to maximizing the flow from M to S, as stated by the Max-Flow-Min-Cut theorem. Finally, the maximum flow can be efficiently computed by the Ford-Fulkerson algorithm [20] in polynomial time. Thus, to derive the number of edge-disjoint M-S paths in a given PTP network $\mathcal{G}$, we propose an adapted version of the Ford-Fulkerson algorithm which is depicted in Algorithm 1. First, we initialize the set of existing paths and the assigned flow values for each edge to zero (lines 1-2). Then, we repeatedly search for M-S paths containing only edges with remaining capacity, i.e., with $flow(e) = 0$ (line 3) and insert them into the set of existing paths (line 4). The search can be accomplished, for example,

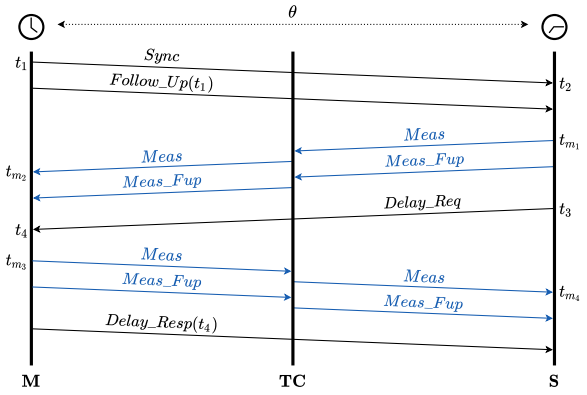


Fig. 5: Proposed PTPsec message flow to protect against time delay attacks. After the reception of PTP event messages (`Sync` and `Delay_Req`), dedicated (`Meas`) messages are returned to the originator via a redundant network path, here indicated with the additional TC, to enable cyclic path asymmetry measurements. The `Meas_Fup` messages convey the captured timestamps t_{m_1} and t_{m_3}, respectively, when PTP operates in two-step mode.

with the breadth-first search algorithm. For each found path, we additionally update the flow values of all comprising edges to mark their capacity as consumed (line 5) for subsequent search iterations. Finally, we return the resulting set of desired edge-disjoint paths (line 7).

D. Multipoint Synchronization

In typical PTP deployments, many nodes synchronize to a single *Grandmaster* clock. Thus, we need to find redundant paths for all eligible synchronization paths. For this purpose, we run Algorithm 1 for every node, which shall be synchronized to the *Grandmaster* before the normal protocol flow. Moreover, we can leverage SDN capabilities to dynamically recalculate redundant paths on any network change.

V. Proposed PTPsec

As already highlighted in § II-A, the latest IEEE 1588-2019 standard (PTP v2.1) is not secure against time delay attacks. Hence, we propose the PTPsec protocol, which integrates appropriate countermeasures into the existing IEEE standard to improve PTP's resilience against such attacks. Recall that in our attacker model, attackers can precisely delay individual packets. Particularly, they could selectively delay only PTP messages if these were distinguishable and independent from our measurement packets. Therefore, we need to entangle the RTT measurements with the PTP synchronization procedure to prevent bypassing the presented attack detection and mitigation approach. For that, we aim to integrate the two critical event messages `Sync` and `Delay_Req` directly into the RTT measurements. As a consequence, we ensure that deliberate packet delays, which impede the time synchronization, also equally affect the proposed detection approach.

Let RTT_{P_0,P_i} be the RTT measurement including the synchronization path P_0 and a redundant path P_i to estimate the path asymmetry α_{P_0}. Then, the `Sync` message, which is sent from node M to S via P_0, already initiates the first RTT measurement. Once received at node S, an immediate response is returned to M via the redundant path P_i to finish the round trip. Since this response has a different purpose than existing PTP message types, we introduce the new message type `Meas` in our proposed PTPsec protocol. Similar to `Sync` messages, also `Meas` packets are PTP event messages due to the requested timestamps at packet transmission and reception. We denote the captured `Meas` timestamps with an additional m, e.g., t_{m_1}, to distinguish them from the four default PTP timestamps. If the protocol operates in two-step mode, we require an additional new message `Meas_Fup` to forward the captured timestamp in a separate follow-up message. To continue the message flow, S sends the expected `Delay_Req` message via P_0 to M which is immediately followed by another `Meas` message to S via P_i upon reception to finish the second measurement cycle yielding RTT_{P_i,P_0}. Finally, M answers the request with a `Delay_Resp` message to complete the synchronization protocol. Fig. 5 shows the full sequence of our proposed PTPsec protocol. With all captured timestamps in this iteration, the two RTT measurements compute as:

$$\begin{aligned} RTT_{P_0,P_i} &= (t_{m_2} - t_1) - (t_{m_1} - t_2) \\ RTT_{P_i,P_0} &= (t_{m_4} - t_3) - (t_{m_3} - t_4) \end{aligned} \tag{11}$$

from which we derive the path asymmetry estimate for P_0:

$$\alpha_{P_0} = RTT_{P_0,P_i} - RTT_{P_i,P_0} \tag{12}$$

Note that in the presented procedure, both critical PTP event messages are fully integrated into the path asymmetry analysis. Consequently, any malicious delay equally impacts the synchronization and the detection mechanism, preventing attackers from bypassing our approach even if individual PTP messages can be distinguished and delayed. Since the proposed PTPsec protocol requires additional measurement packets for the asymmetry analysis, it also introduces a message overhead compared to conventional PTP. In particular, four supplementary packets are sent on each of the n redundant network path per PTP synchronization cycle, resulting in the following total number of PTPsec messages per cycle:

$$\#\,packets = 4 + 4n \tag{13}$$

Note the packet count's linear increase with the number of redundant paths. However, the actual number of redundant paths to consider for securing the protocol is a security parameter that can be adjusted depending on the assumed attack model. Furthermore, PTP messages usually account only for a small part of the entire network traffic which alleviates the impact of the packet increase.

VI. Evaluation

For reliable evaluation, we validate the performance of our proposed PTPsec protocol under various realistic attack scenarios on our hardware testbed.

A. Hardware Setup

The hardware setup consists of two PCs serving as master and slave to be synchronized. Both machines are operating on

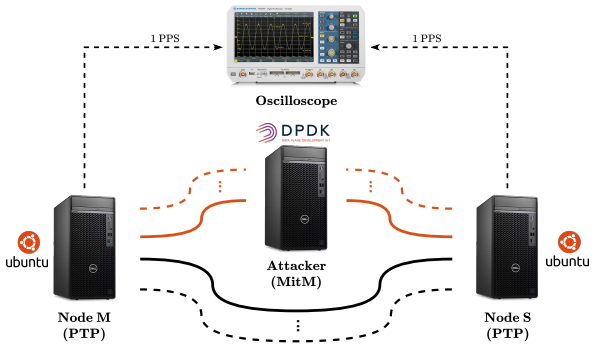


Fig. 6: Hardware testbed for comprehensive evaluation of our proposed approach under realistic circumstances. The synchronization path is interrupted by a MitM attacker delaying selected PTP event messages in transit to diverge the slave clock. We closely monitor the malicious clock offset with the oscilloscope by comparing the generated PPS signals.

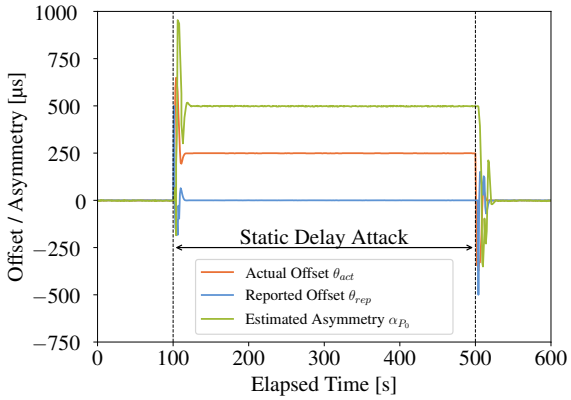


Fig. 7: Static delay attack with $\epsilon_1 = 500\,\mu s$ targeting the PTP `Sync` message between $t = 100\,s$ and $t = 500\,s$. The difference between the measured clock offset (orange) and the reported clock offset (blue) confirms the ongoing attack. Nevertheless, our proposed asymmetry analysis successfully detects this malicious behavior (green).

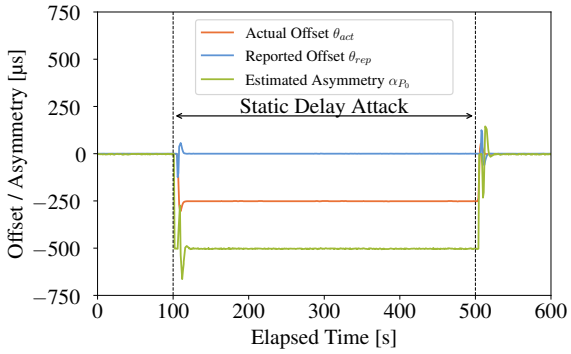


Fig. 8: Static delay attack with $\epsilon_2 = 500\,\mu s$ targeting the PTP `Delay_Req` message between $t = 100\,s$ and $t = 500\,s$. The difference between the measured clock offset (orange) and the reported clock offset (blue) confirms the ongoing attack. However, our proposed asymmetry analysis successfully detects this malicious behavior (green).

Ubuntu 20.04 and run an implementation of our PTPsec protocol, which is based on *linuxptp-v3.1.1*. The two OCs use Intel i210 Network Interface Cards (NICs) for IEEE 1588-compliant hardware timestamping support and are connected with wired Ethernet connections. The synchronization path is interrupted by an attack device running a Data Plane Development Kit (DPDK) application to act as a transparent L2 switch. Moreover, it exploits its Machine-in-the-Middle (MitM) position to deliberately delay either `Sync` or `Delay_Req` messages to implement an effective time delay attack against PTP, as described in § II-B. The other path is a direct point-to-point connection and constitutes a redundant path that is considered genuine and symmetric for all following experiments. Extending the setup with more redundant paths follows the same principle, as illustrated in Fig. 6. To monitor the actual clock offset between master and slave, we compare the generated Pulse-Per-Second (PPS) signal phases on a Rohde&Schwarz RTB oscilloscope.

B. Asymmetry Detection

In total, we present three experiments to validate the detection performance of our protocol. We attack the two relevant PTP event messages `Sync` and `Follow_Up` and assume that `Meas` and `Meas_Fup` are only sent via the symmetric redundant path. Attacking the newly introduced measurement packets instead of the PTP messages would lead to similar results because for the asymmetry analysis, it is irrelevant which specific path is attacked provided one genuine path exists.

1) Static Delay (`Sync`): In the first experiment, we start with a static attack scenario, where the attacker adds a constant delay of $\epsilon_1 = 500\,\mu s$ to all passing `Sync` messages. The attack starts at $t = 100\,s$ and lasts until $t = 500\,s$, as shown in Fig. 7. During that period, we observe that the actual clock offset increases to $\theta_{act} = 250\,\mu s$, i.e., by half the introduced one-way delay, while the reported PTP clock offset θ_{rep} remains at zero. This behavior clearly shows the success of the delay attack because the actual clock offset changes to the expected value given by (2) without being reported by conventional PTP. Furthermore, the estimated synchronization path asymmetry rises to $\alpha_{P_0} = 500\,\mu s$ during the attack period. Since $\alpha_{P_0} > 0$, the path delay from node M to node S is higher than in the opposite direction, and we conclude an ongoing delay attack targeting the `Sync` message. Hence, the applied attack was successfully detected.

2) Static Delay (`Delay_Req`): We repeat the previous attack in the second experiment with the only difference in the targeted PTP message. This time, the attacker delays passing `Delay_Req` messages instead, which results in a path asymmetry in the opposite direction. As Fig. 8 depicts, the reported PTP clock offset θ_{rep} remains at zero while the actual clock offset $\theta_{act} = -250\,\mu s$ and the estimated path asymmetry $\alpha_{P_0} = -500\,\mu s$ both show a change of sign. The negative values match our expectations due to the different PTP message that was targeted. These results confirm that the attack detection works independently of the present asymmetry direction.

3) Incremental Delay Attack: Finally, we change the attack method to an incremental delay attack to evaluate the detection performance more dynamically. Therefore, we gradually increase the packet delay for passing `Sync` messages to slowly

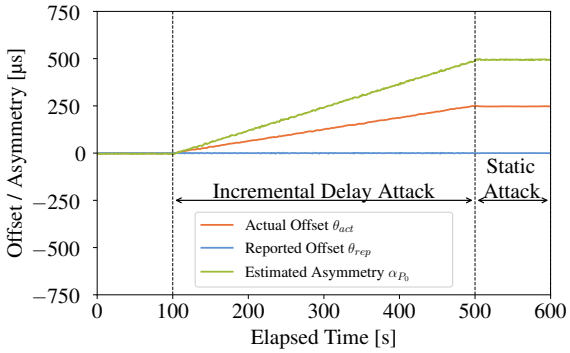


Fig. 9: Incremental delay attack, where the offset is gradually increased by $\Delta\epsilon = 1.25\,\mu s$ starting at $t = 100\,s$ with $\epsilon_1 = 0\,\mu s$ to a final level of $\epsilon_1 = 500\,\mu s$. The attack targets the PTP `Sync` message. The difference between the measured clock offset (orange) and the reported clock offset (blue) confirms the ongoing attack. Nevertheless, our proposed asymmetry analysis successfully detects this malicious behavior (green).

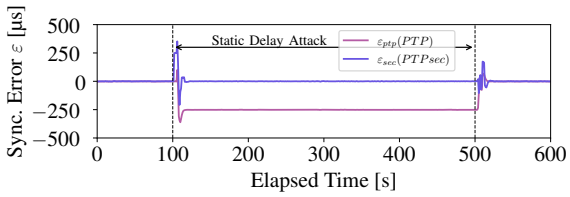


Fig. 10: Remaining clock error after delay attack compensation considering the estimated path asymmetry.

diverge the slave's clock. The attack starts at $t = 100\,s$. Each second, we increase the added delay by $\Delta\epsilon = 1.25\,\mu s$ to eventually reach a level of $\epsilon_1 = 500\,\mu s$ at $t = 500\,s$ which remains as a static offset until the end of the experiment. From the results in Fig. 9, we observe that not only the actual clock offset gradually ascends from $\theta_{act} = 0\,\mu s$ to $\theta_{act} = 250\,\mu s$, but also the estimated path asymmetry α_{P_0} slowly follows the incremental delay to reach a final level of $\alpha_{P_0} = 500\,\mu s$. Therefore, we conclude that our proposed protocol also reliably detects incremental delay attacks.

C. Attack Mitigation

As the previous experiments show, our proposed PTPsec protocol reliably detects the malicious path asymmetry in all attack scenarios. To further evaluate the attack mitigation performance, we analyze the synchronization errors ε_{ptp} and ε_{sec} of both conventional PTP and PTPsec, respectively, during a static delay attack by comparing the reported clock offset to the actual clock offset measured by the oscilloscope. For PTPsec, we use the rectified offset from (10). As the results in Fig. 10 illustrate, PTPsec reliably mitigates the attack.

D. Detection Time

In addition to the general detection and mitigation capabilities, we further examine the timing behavior of the proposed approach. Particularly, we investigate the elapsed time between the start of the attack and its visibility in the measured path asymmetry, i.e., the detection time of our method. This time is critical since it potentially opens a small window for attackers

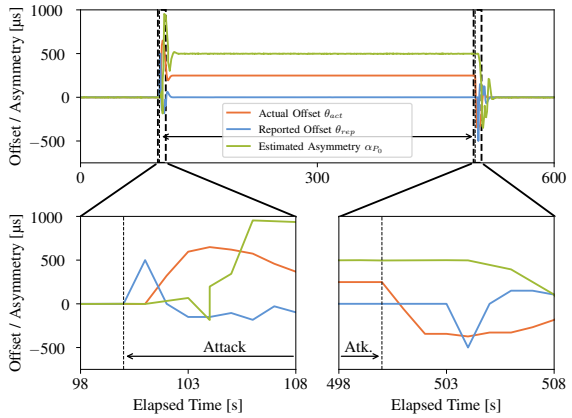


Fig. 11: Timing analysis of our attack detection approach. The magnified plots at the bottom illustrate our method's fast detection time at the start and the end of the applied attack.

during which the deployed attack is effective without an appropriate system response and, thus, needs to be minimized. In our setup, the `Sync` message timeout interval is set to 1 s which also predetermines the clock update and asymmetry measurement rate to approx. this period. As the plot in Fig. 11 shows, the measured path asymmetry follows the actual clock offset almost immediately within less than five time steps at the start and the end of the attack.

VII. Discussion

The proposed PTPsec protocol has successfully detected and counteracted the delay attacks in all conducted experiments validating the effectiveness of our approach. Since the chosen scenarios are representative of all known time delay attack strategies, we can conclude that our cyclic asymmetry analysis enables reliable attack mitigation provided redundant paths are available in the network. Interestingly, this redundancy requirement is already enforced by some communication standards, such as TSN, so our secure protocol could be deployed in related applications at no cost. In all other cases, the introduced procedure in Algorithm 1 can serve as a network analysis tool to evaluate the current security level regarding time delay attacks. Once a bottleneck has been identified, the network can be selectively patched to meet the specified requirements. Regarding the protocol overhead of PTPsec, the increased security is not for free. Since both the number of successfully detected asymmetric paths and the required packet count scale with the number of redundant network paths, it is a trade-off between desired security and acceptable network load that system designers have to make.

VIII. Related Work

Attack detection and prevention in time synchronization have been broadly covered by research in recent years. The proposed solutions to protect PTP from time delay attacks can be mainly clustered into three groups. First, there exist threshold-based detection techniques. The idea is to continuously monitor the reported clock offset and path delay and decide whether or not the system is under attack based on

a defined threshold. The threshold can be either static, as proposed by Ullmann et al. [21], or dynamically updated over time, as presented in [6, 7]. However, the performance of these techniques strongly depends on the chosen threshold, which is quite challenging. The second category comprises solutions that rely on special guard devices and additional reporting systems deployed in the network. Alghamdi et al. [22] present the idea of a trusted supervisor node that performs anomaly detection in the network to detect ongoing attacks. Moussa et al. [9] propose a similar idea, including a dedicated guard node that participates in the time synchronization protocol but does not update its clock. Instead, it only compares the results to another time reference to reveal potential delay attacks. However, the guard node only secures its own synchronization path, and other nodes that do not share the same path are still vulnerable. In [11], Moussa et al. refine their initial proposal and introduce a more sophisticated approach, where all nodes additionally send timestamped reports to a specific network time reference node. These reports are exchanged by means of custom event messages that are not related to the PTP protocol stack. Thus, sophisticated attackers can distinguish report messages from actual PTP messages and react differently to disrupt time synchronization without being noticed. Moradi et al. [12] present a similar method, including a dedicated reporting scheme for attack detection. The third class of countermeasures utilizes path redundancy for time delay attack protection. There exist some works, such as [23, 24, 25], which cover path redundancy for network-based time synchronization to improve the provided synchronization accuracy. While not explicitly mentioning any security aspects, the works already indicate the capabilities of redundant path approaches, which could also be adopted in the security domain. In [13], Mizrahi proposes redundant paths as countermeasure for delay attacks and presents a theoretical analysis of this idea. Furthermore, Neyer et al. [8] perform supplementary experiments on a hardware testbed to showcase the general applicability of redundant paths as viable improvement for time synchronization security.

Comparative Analysis: Table I presents an additional comparison between our proposed solution and existing works. The comparison is based on the adopted approach, the ability to detect static, incremental, and selective delay attacks, the scalability of the proposed solution, and the experimental validation method employed (hardware setup, simulation, or no validation). While all the solutions are capable of detecting static attacks, [6, 7, 21] fall short of fully detecting incremental attacks due to the threshold-based approach. Others, like [11, 12, 22], are still vulnerable to selective delay attacks, where the attackers distinguish between PTP and other network traffic. In contrast to PTPsec, many other solutions do not adequately scale either due to the need for a recurring setup phase after each network change, as seen in [6, 21], including single points of failure [9, 22], or due to limitations in the system model [8]. Furthermore, hardware testbed validation is essential for ensuring the effectiveness of proposed detection methods, which most of the compared solutions are lacking. Based on Table I, the three most comparable solutions to our system are [7], [13], and [8]. However, [7] cannot reliably detect incremental delay attacks because of the threshold-based approach while additionally requiring an initial learning phase that invalidates with any network change. Furthermore, the presented solutions in [13] and [8] lack experimental validation and scalability, respectively. Moreover, while partially providing sound detection approaches, none of the compared works can actually mitigate time delay attacks which is an exclusive feature of PTPsec.

Table I: Delay Attack Detection Approaches Comparison

Solution	**Approach**	**Attack Resistance**			**Scal.**	**Exp.**
		Static	Incr.	Select.		
[6]	Threshold	●	◑	●	○	◑
[21]	Threshold	●	◑	●	○	○
[7]	Threshold	●	◑	●	◑	●
[22]	Guard	●	●	○	○	○
[9]	Guard	●	●	●	○	◑
[11]	Guard	●	●	○	●	◑
[12]	Guard	●	●	○	●	◑
[13]	Path Red.	●	●	●	●	○
[8]	Path Red.	●	●	●	○	●
PTPsec	**Path Red.**	●	●	●	●	●

○ no or n/a, ◑ partially, ● yes

IX. Conclusion

This work introduces a theoretical model for cyclic path asymmetry analysis that can be efficiently used for time delay attack mitigation. The derived attack detection method reliably reveals both static and incremental time delay attacks provided the existence of at least one redundant network path with symmetric delay. With PTPsec, we present a protocol that secures the latest IEEE 1588 standard against time delay attacks by incorporating our proposed mitigation techniques on top of conventional cryptographic countermeasures. This makes PTPsec fully resilient against all known attacks against PTP. The experimental results show that our approach successfully identifies fine-grained asymmetry changes with microsecond accuracy and minimal detection time in all evaluated scenarios. The authors have provided public access to their code and/or data at https://github.com/tum-esi/ptpsec.

Acknowledgment

This work has received funding from The Bavarian State Ministry for the Economy, Media, Energy and Technology, within the R&D program „Information and Communication Technology", managed by VDI/VDE Innovation + Technik GmbH. Also, this work is supported by the European Union-funded project CyberSecDome (Agreement No.: 101120779).

References

[1] A. Sargolzaei, K. Yen, and M. N. Abdelghani, "Delayed inputs attack on load frequency control in smart grid," in *ISGT 2014*. IEEE, 2014, pp. 1–5.

[2] "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)*, pp. 1–499, 2020.

[3] S. Barreto, A. Suresh, and J.-Y. Le Boudec, "Cyber-attack on packet-based time synchronization protocols: The undetectable delay box," in *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*. IEEE, 2016, Conference Proceedings.

[4] R. Annessi, J. Fabini, F. Iglesias, and T. Zseby, "Encryption is futile: Delay attacks on high-precision clock synchronization," *arXiv preprint arXiv:1811.08569*, 2018.

[5] A. Finkenzeller, T. Wakim, M. Hamad, and S. Steinhorst, "Feasible time delay attacks against the precision time protocol," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022.

[6] Q. Yang, D. An, and W. Yu, "On time desynchronization attack against ieee 1588 protocol in power grid systems," in *2013 IEEE Energytech*. IEEE, 2013, pp. 1–5.

[7] H. Li, D. Li, X. Zhang, G. Shou, Y. Hu, and Y. Liu, "A security management architecture for time synchronization towards high precision networks," *IEEE Access*, vol. 9, pp. 117 542–117 553, 2021.

[8] J. Neyer, L. Gassner, and C. Marinescu, "Redundant schemes or how to counter the delay attack on time synchronization protocols," in *2019 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. IEEE, 2019, Conference Proceedings.

[9] B. Moussa, M. Debbabi, and C. Assi, "A detection and mitigation model for ptp delay attack in a smart grid substation," in *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2015, Conference Proceedings, pp. 497–502.

[10] W. Alghamdi and M. Schukat, "Advanced methodologies to deter internal attacks in ptp time synchronization networks," in *2017 28th Irish Signals and Systems Conference (ISSC)*. IEEE, 2017, Conference Proceedings.

[11] B. Moussa, M. Kassouf, R. Hadjidj, M. Debbabi, and C. Assi, "An extension to the precision time protocol (ptp) to enable the detection of cyber attacks," *IEEE Transactions on Industrial Informatics*, 2020.

[12] M. Moradi and A. H. Jahangir, "A new delay attack detection algorithm for ptp network in power substation," *International Journal of Electrical Power & Energy Systems*, vol. 133, 2021.

[13] T. Mizrahi, "A game theoretic analysis of delay attacks against time synchronization protocols," in *2012 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication Proceedings*. IEEE, 2012, Conference Proceedings.

[14] W. Alghamdi and M. Schukat, "Precision time protocol attack strategies and their resistance to existing security extensions," *Cybersecurity*, vol. 4, no. 1, 2021.

[15] J.-H. Choi and C. Yoo, "One-way delay estimation and its application," *Computer Communications*, 2005.

[16] O. Gurewitz and M. Sidi, "Estimating one-way delays from cyclic-path delay measurements," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213)*, vol. 2. IEEE, 2001, pp. 1038–1044.

[17] O. Gurewitz, I. Cidon, and M. Sidi, "One-way delay estimation using network-wide measurements," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2710–2724, 2006.

[18] T. Böhme, F. Göring, and J. Harant, "Menger's theorem," *Journal of Graph Theory*, vol. 37, no. 1, 2001.

[19] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian journal of Mathematics*, vol. 8, pp. 399–404, 1956.

[20] L. R. Ford Jr and D. R. Fulkerson, *Flows in networks*. Princeton university press, 2015, vol. 54.

[21] M. Ullmann and M. Vogeler, "Delay attacks — implication on ntp and ptp time synchronization," in *2009 International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*. IEEE, 2009, Conference Proceedings.

[22] W. Alghamdi and M. Schukat, "Cyber attacks on precision time protocol networks—a case study," *Electronics*, vol. 9, no. 9, p. 1398, 2020.

[23] T. Mizrahi, "Slave diversity: Using multiple paths to improve the accuracy of clock synchronization protocols," in *2012 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication Proceedings*. IEEE, 2012.

[24] A. Komes and C. Marinescu, "IEEE 1588 for redundant ethernet networks," in *2012 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication Proceedings*. IEEE, 2012, Conference Proceedings.

[25] A. Shpiner, Y. Revah, and T. Mizrahi, "Multi-path time protocols," in *2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS) Proceedings*. IEEE, 2013.