Resilient Clock Synchronization over Path-Aware Wide-Area Networks

Marco Pioppini*, Marc Frei*, Sven Meier†, Kevin Schärer†, Thomas Schaub†, Marc Wyss*, and Adrian Perrig*

*Network Security Group, Department of Computer Science, ETH Zurich, Switzerland
Email: {marco.pioppini, marc.frei, marc.wyss, adrian.perrig}@inf.ethz.ch
†NetTimeLogic GmbH, Zurich, Switzerland
Email: {sven.meier, kevin.schaerer, thomas.schaub}@nettimelogic.com

***Abstract*—Across critical infrastructure sectors, wide-area clock synchronization over dedicated or leased-line networks is becoming increasingly constrained as infrastructures grow more distributed and demands for flexibility and cost efficiency rise. At the same time, relying solely on the public Internet raises concerns around resilience, security, synchronization accuracy, and digital sovereignty.**

To bridge this gap, we propose *path-aware wide-area networks* as a substrate for resilient clock synchronization in critical infrastructures, combining the flexibility and cost efficiency of the public Internet with the control and security guarantees of dedicated networks. By maintaining multiple paths between a client and its reference time servers, our proposed system can tolerate faults and congestion affecting a fraction of the paths. At a higher level, a multi-source clock-steering layer combines these network-derived references with local sources such as GNSS receivers, reducing dependence on any single timing source. We present the system design and implementation, built on the SCION path-aware Internet architecture, along with early pilot deployments.

I. Introduction

Clock synchronization for critical infrastructure is becoming increasingly challenging as infrastructures grow more distributed and demands for resilience, flexibility, and cost efficiency increase.

With IT and OT networks converging, the dedicated and leased-line networks that historically offered simplicity, control, and determinism are reaching their limits. Field-proven technologies, which supported critical operations for decades, are approaching end of life, leaving operators with a shrinking support ecosystem and rising costs. Meanwhile, the proliferation of distributed assets, from renewable energy resources to remote substations and branch offices, creates scaling demands that rigid point-to-point provisioning cannot meet economically and with sufficient flexibility.

The public Internet offers the cost-effectiveness and flexibility required in modern infrastructure but operates to a significant extent as a black box: end hosts cannot influence routing, forwarding happens with little to no information about individual traffic characteristics, and paths are often asymmetric and unstable. Routes may be misconfigured, and failover or repair after link or equipment failures can be slow. These properties are fundamentally at odds with accurate and dependable time distribution. At the same time, evolving geopolitical threat assessments and growing demands for digital sovereignty further call into question reliance on shared infrastructures without stronger control guarantees.

TABLE I
Qualitative comparison between connectivity options.

	Dedicated Networks	**SCION**	**Public Internet**
Control	✔	✔	✗
Flexibility	✗	✔	✔
Security	✔	✔	✗
Low Cost	✗	✔	✔

Beyond these operational limitations, clock synchronization over open networks is inherently exposed to adversarial threats. On-path attackers can manipulate time by influencing traffic between clients and servers, and compromising even a small fraction of network elements along the synchronization path can induce difficult-to-detect time shifts. Routing attacks, particularly through route hijacking, can divert synchronization traffic and enable such manipulation. Even cryptographic authentication and integrity protection fundamentally cannot prevent delay attacks when operating over networks outside the end host's control. And even in the absence of malicious on-path entities, protecting long-haul synchronization against volumetric DDoS attacks launched by off-path attackers remains an active research area rather than a solved problem [1].

Path-aware networking architectures such as SCION [2] occupy a promising middle ground between existing solutions (see Table I), combining Internet-like flexibility and cost-effectiveness with explicit path control, multipath communication, and strong routing security, and offer properties directly relevant to wide-area time distribution: communication can be constrained to trusted routes, path diversity can contribute to fault tolerance and DDoS resilience, and path transparency can improve timing predictability. To leverage these properties, we present a system for clock synchronization over path-aware wide-area networks with real-world pilot deployments and support in custom-designed production-grade hardware.

II. SCION

This section gives an overview of the SCION Internet architecture [2], a core component of our design.

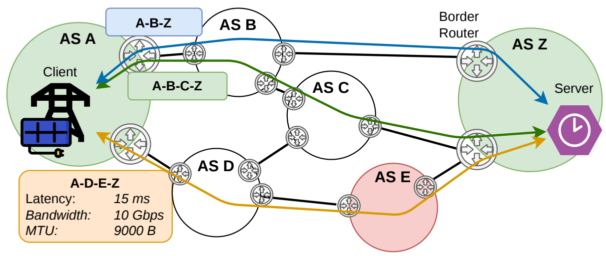

Fig. 1. SCION multipathing example. The client in AS A obtains several symmetric metadata-annotated paths (represented by the arrows) to the clock source in AS Z. The client learns that one path traverses AS E, an untrusted network, and avoids this path.

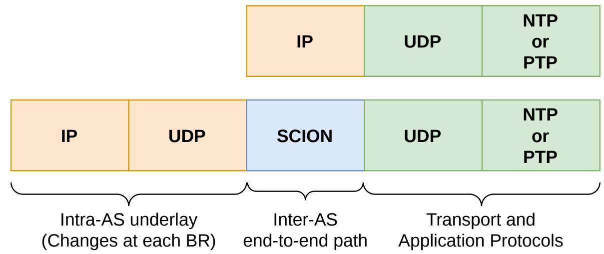

Fig. 2. Headers in a plain IP packet (top) and in a SCION packet (bottom). In IP, forwarding decisions are based solely on the destination address, whereas SCION packets explicitly encode the end-to-end forwarding path in the SCION header. The SCION header is preceded by underlay headers used for intra-AS routing of SCION packets (between end hosts and border routers, or between pairs of border routers).

The public Internet consists of many interconnected but independently operated networks. These networks are referred to as Autonomous Systems (AS). As it would be infeasible for each AS to connect to every other AS directly, most data packets sent over the Internet traverse multiple ASes on the way to their destination. Often, multiple AS-level paths exist between different ASes, yet in the public Internet only one is used. SCION provides a way to discover these *inter-AS* paths and expose them to end hosts, allowing end hosts to choose over which path to send their traffic. *Intra-AS* routing, i.e., the internal connectivity of each AS, is not covered by SCION; as in the public Internet, each AS handles this independently.

A. Overview

An example of a SCION topology is shown in Figure 1. Each SCION AS is connected to one or more neighbor ASes via inter-AS links. An AS contains *end hosts* (devices which send or receive traffic, e.g., Network Time Protocol (NTP) [3] or Precision Time Protocol (PTP) [4] servers and clients) and *border routers* (devices which forward traffic from or to outside the AS). The ASes periodically run a path discovery process to find available paths between ASes in the network. SCION discovers paths at the AS-interface level, i.e., a SCION path includes the ingress and egress points (referred to as *interfaces*) of each traversed AS. The paths discovered in this process are cryptographically protected and can contain additional metadata, such as the geographical coordinates of each hop, or the expected network latency along the path.

An end host wishing to send traffic to a destination in another AS obtains end-to-end paths to the destination AS from a local path service. It may then send traffic over one or several of the available paths, potentially using the path metadata to inform its choice, e.g., to restrict its choice to paths within a geographical area, or to avoid paths that it knows to have bad performance. In the example in Figure 1, the NTP client in AS A obtains three paths to the server in the remote AS Z. The client learns from the path information that one path traverses the untrusted AS E and avoids that path.

To send application data, the end host encapsulates each data packet into a SCION packet. An example SCION packet in comparison to an equivalent plain IP packet is shown in Figure 2. The two outermost headers of a SCION packet contain intra-AS forwarding information and are replaced by each border router the packet traverses. This intra-AS underlay preserves compatibility with existing routing setups. The underlay headers are followed by the SCION header, which contains the SCION path and end host addresses, followed by the transport and application headers, and the payload. Note that, because the forwarding path is specified in each packet's header, end hosts can select the path for each packet individually, and thus can switch paths at any time. The end host sends its SCION packets to its local border router. From there, packets are forwarded between border routers according to the specified path until they reach the border router of the destination AS, which forwards them to the destination end host. When the destination receives a SCION packet, it can extract the packet's path from the SCION header, reverse it, and use it to send its reply over the same path used by the sender, making SCION paths symmetric.

B. Critical Infrastructure and Clock Synchronization

Several properties make SCION an ideal substrate for critical infrastructure in general, and for the specific task of clock synchronization in particular. Most importantly, SCION offers applications control and transparency over how their traffic is forwarded. Because end hosts learn the available paths and mark each sent packet with the exact path it must take, they can, for example, confine their data traffic to trusted geographical regions. Moreover, applications can send traffic over multiple paths at their discretion, gaining redundancy, and the cryptographic protections in SCION prevent routing attacks such as route hijacking.

Like Internet connectivity, SCION connectivity can be obtained from existing service providers without the expenses and overheads associated with dedicated networks. Connectivity via multiple providers is also possible, providing redundancy and flexibility. At the same time, despite relying on shared infrastructure, SCION offers a level of security and control that otherwise requires dedicated setups such as SD-WAN [5]. Unlike such solutions, however, SCION does not create a dependency on a single vendor or provider. SCION is an open architecture shepherded by an independent organization [6] and is currently undergoing specification through

IETF drafts [7], [8]. This allows different vendors to build compatible implementations, reducing the risk of vendor lock-in. Besides the global, open SCION Internet, SCION has seen adoption in several critical infrastructure sectors: finance and electronic payments, healthcare, and energy and utilities [9].

For clock synchronization in particular, SCION's path diversity, path stability and path symmetry are directly beneficial. As a multipath architecture, SCION exposes the path diversity that remains unused in the public Internet: At the time of writing, we were able to find paths to 99 SCION ASes across the world from our vantage point at ETH Zurich, with a median of 204 distinct available paths to each AS. Within Switzerland only, we counted 59 ASes with a median and maximum path count of 328 and 4184, respectively. This path diversity is not only useful for redundancy, but it can also improve clock synchronization accuracy [10]. Moreover, SCION provides path stability, meaning that forwarding paths do not change unexpectedly. Unlike in the public Internet, where paths can change without warning (e.g., due to configuration changes), packets in SCION consistently follow their explicitly selected paths. When used with SCION's path reversal mechanism, these paths are also symmetric, an advantage for delay measurements based on round-trip times.

Finally, SCION also supports several extensions to its functionality. For example, FABRID [11] enables ASes to expose intra-AS path properties and allows end hosts to select ones that satisfy their policies, such as traversing only devices with hardware support for PTP timestamps. Another extension, Hummingbird [12], enables end-to-end bandwidth reservations. Such reservations provide quality-of-service guarantees and thwart denial-of-service attacks.

III. Clockwire Design and Implementation

Clockwire is our implementation of a complete wide-area clock synchronization system designed to make optimal use of SCION's multipath capabilities. While initially developed in a research context, the project is progressively evolving toward production quality. Multiple pilot deployments are currently under way, and the lessons learned from them are continuously informing further development. An initial open-source release is planned for late 2026, to be followed by further refinement in collaboration with industry partners. The goal is to enable full-scale production deployments.

A. Service Model and Protocol Support

Clockwire follows a traditional and well-established time-service model, with a daemon that can operate as a primary time server, an intermediate server, or a client steering a local clock. In each of these roles, Clockwire can integrate and combine multiple local and remote clock sources. Networked reference time sources can be accessed over Layer 2, IP (using a single path), or SCION (using multiple paths). Likewise, Clockwire can serve time over both IP and SCION.

At the application layer, Clockwire implements a subset of NTP, with the goal of eventual compliance with NTP version 5 [13]. This results in a dual-stack design supporting both NTP/UDP/IP and NTP/UDP/SCION. In addition, we are following the ongoing IEEE standardization of the Client-Server Precision Time Protocol (CSPTP) [14] with an early implementation in Clockwire.[1]

Message authentication is provided via the Network Time Security Protocol (NTS) [15] for NTP over both IP and SCION, in client and server roles. NTS support for CSPTP is also planned, following the ongoing discussions on NTS for CSPTP within the IEEE 1588 Security Working Group.

Clockwire is implemented in Go for Linux-based systems and takes full advantage of software and hardware timestamping capabilities where available. For NTP this is achieved via Interleaved Modes [16], for CSPTP it is supported directly by the protocol. The software can therefore be deployed on commodity hardware platforms, including virtualized environments. Beyond this standard deployment model, we are pursuing an integrated hardware/software co-design that lays the foundation for achieving single-digit-microsecond accuracy, as required by many critical infrastructure systems. This approach enables full transparency and control across the entire end-to-end synchronization path, down to the hardware level.

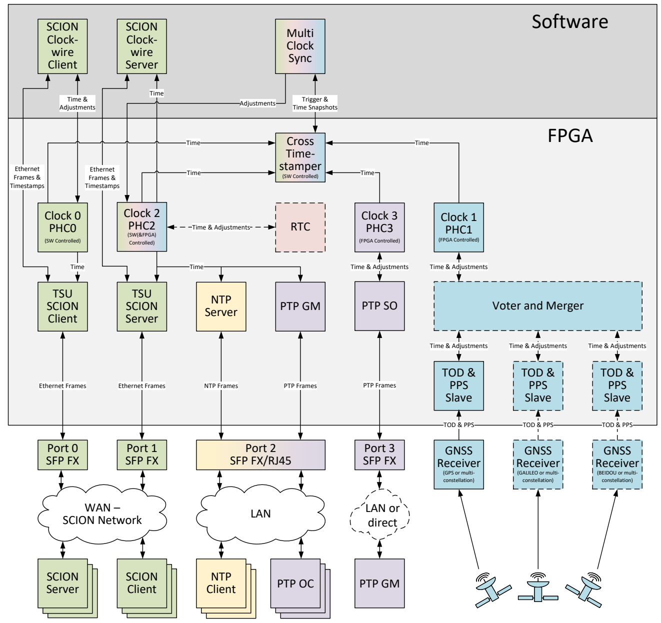

Fig. 3. Reference Design Hardware Platform.

B. Hardware Platform and Multi-Clock Synchronization

The proposed hardware platform (shown in Figure 3) is based on a Multi-Processor System-on-Chip (MPSoC) architecture that integrates programmable logic (FPGA) and multiple processing units (CPUs) on a single device. This heterogeneous architecture enables a clear functional partitioning

[1]CSPTP defines a simple and precise time-transfer protocol that reuses message formats from IEEE Std 1588 (PTP) while adopting a substantially simpler and more scalable design suitable for wide-area settings: clients can obtain timing information from any CSPTP server, and servers operate without maintaining per-client state.

between time-critical and non-time-critical tasks. Specifically, latency- and jitter-sensitive operations such as timestamping are implemented in the deterministic, hard real-time domain of the FPGA, while higher-layer functions, including protocol stack processing, are executed in software on general-purpose CPUs. A key feature of the reference design is its multi-source, multi-clock, and multi-sink time synchronization architecture. This design enables robust time generation through the combination, comparison, and validation of multiple time sources, taking into account their availability, stability, accuracy, and operational state. Each time source is associated with an independent local clock that is disciplined solely by that source. To enable interoperability between these independent clocks, a cross-timestamping mechanism is employed to establish precise temporal relationships among them. This approach offers a significant advantage: individual time sources remain isolated and do not directly influence each other during their initial synchronization phase. In contrast to conventional architectures that require early-stage selection or voting among sources, often complicated by differing update rates, this design defers source fusion (e.g., voting or merging) to a higher level. The multi-clock paradigm allows time-source fusion to be performed at arbitrary intervals, independent of the individual update rates of the sources. A designated system clock, used for time distribution, is continuously disciplined based on the ensemble of available clocks. In its simplest form, this can be realized by arithmetic averaging of all valid sources. More sophisticated methods include weighted averaging based on statistical performance metrics (e.g., jitter, drift, or variance), as well as decision mechanisms such as majority voting or trust-based ranking.

For integration with Clockwire, a dedicated Hardware Timestamp Unit (TSU) has been developed, providing a timestamping resolution of 4 ns. This unit enables precise timestamping of NTP and PTP packets encapsulated within SCION headers over Ethernet. The TSU operates within the FPGA fabric, ensuring deterministic and low-latency timestamp generation. The Clockwire architecture is explicitly divided into client and server components. The server is responsible for distributing a consolidated system time, derived from multiple sources such as local PTP references, multiple Global Navigation Satellite Systems (GNSS), and remote reference time sources. This consolidated time is made available through multiple dissemination mechanisms. In addition to NTP and CSPTP over SCION, these include a hardware-based NTP server and a PTP Grandmaster (GM), enabling the system to act as a time bridge for connected local area networks. Since SCION-based synchronization over a WAN is not expected to achieve the same accuracy as GNSS or direct synchronization from a local PTP time transmitter, a dedicated local clock is synchronized independently using Clockwire. This clock may be additionally filtered before contributing to the overall time-fusion process. Its contribution can then be adaptively weighted or conditioned based on observed performance.

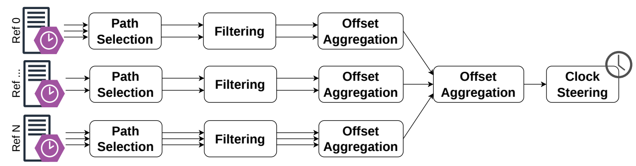

Fig. 4. Clockwire's multipath synchronization pipeline. Clockwire uses multiple redundant paths to connect to several reference time servers. Timing information from each path and server (represented by the arrows) is filtered, then aggregated and used to steer a local clock.

C. Multipath Synchronization Pipeline

The most novel part of Clockwire's software design is its multipath synchronization pipeline, shown in Figure 4, which collects, filters, and combines time-offset measurements from multiple SCION-based reference time sources. The resulting aggregate time is used to steer the dedicated local clock that maintains the Clockwire client time. In Figure 3, this clock is shown as the software-controlled FPGA block PHC0. In deployments without explicit multi-clock hardware support, it can be implemented as a Linux virtual clock running atop a shared physical hardware clock.

Path Selection. At the first pipeline stage, a working set of predefined size is selected from the potentially hundreds of paths that the SCION control plane may provide. As described in Section II-A, Clockwire end hosts regularly query path servers in their local AS to obtain paths toward a given destination. In typical network topologies, such queries may return a large number of candidate paths. Using all available paths is rarely beneficial or resource-efficient. Clients consequently select a subset for concurrent use.

The first step in path selection is policy-based screening, which rejects certain paths based on metadata, such as the geographic location or jurisdiction of traversed ASes, in order to enforce legal compliance or privacy requirements.

In a second step, so-called static path selection is performed. A subset of paths is chosen according to the heuristic ”short but diverse”. Short paths reduce the number of hops along the synchronization path at which congestion or asymmetries can affect accuracy and precision. Diversity, in turn, is the main lever for achieving resilience and protection against on-path attackers controlling parts of the network. The algorithm greedily selects up to a predefined number of paths from the candidate set while incorporating any preselected paths. If no path has yet been selected, it starts with the shortest candidate. It then iteratively chooses the path that maximizes a diversity score defined as the number of newly covered inter-AS interfaces minus penalties for path length and overlap with already covered interfaces. Ties are broken non-deterministically to protect against intelligent adversaries in the network.

Clockwire selects up to seven paths per reference time source that are then concurrently used. Once selected, paths remain active until they exceed a preconfigured number of consecutive errors or time-outs, at which point they are replaced by rerunning static path selection with the remaining active paths provided as the preselected input set. Keeping the selection relatively stable in this way helps preserve per-path filter state, thereby improving overall synchronization quality.

Beyond static path selection, the design also allows for a continuous dynamic path-selection process that actively probes the statically selected paths and chooses a preferred subset based on measured round-trip times, assuming that lower round-trip delays correspond to more symmetric path behavior. This performance-oriented technique is not yet implemented in Clockwire, since the initial release prioritizes resilience over maximum synchronization quality.

Filtering, Aggregation, and Steering. At the filtering stage, no special techniques are applied apart from maintaining separate statistics for each path. Clockwire currently implements Poul-Henning Kamp's Ntimed filter [17] to compensate for outliers in unidirectional propagation delays, with an optional "lucky-packet" filter [18] applied in front of it. Offset aggregation then proceeds in two stages: first across all measurements obtained over multiple paths to an individual reference time source, and then across all reference time sources. Given Clockwire's focus on resilience and robustness above all else, both aggregation stages use a fault-tolerant midpoint calculation that remains unaffected by a fraction of anomalous measurements [19]. Finally, for steering the Clockwire client clock, we use a simple PI controller, deliberately relying on a well-understood and proven control algorithm to keep overall system innovation within manageable bounds.

IV. Pilot Deployments

To evaluate Clockwire in realistic network environments, we are currently working on two pilot deployments: an initial setup with SIDN Labs in the Netherlands, and a deployment with the Swiss Federal Institute of Metrology (METAS), Swiss ISPs, and Frankfurter Bankgesellschaft (Schweiz) AG (FBG) as partners.

SIDN Labs. SIDN Labs operates the Dutch Internet time service time.nl and is also an early pioneer in the deployment of SCION infrastructure. This constellation allowed us to set up a Clockwire server on a virtual machine in the Netherlands, using SIDN Labs' PTP backbone as its primary reference time source. Clockwire clients access this server via SCIERA, the research-focused segment of the emerging global SCION network. SCION connectivity from our vantage point at ETH Zurich to SIDN Labs is provided through the research network operators Switch (CH), GÉANT (EU), and SURF (NL), resulting in eight SCION paths from ETH Zurich. Although the deployment does not run entirely on production infrastructure, initial tests demonstrate offset stability within tens of microseconds (see Figure 5), while its SCION-enabled resilience to transient network issues and deliberate attacks is the subject of ongoing evaluation.

Switzerland. To explore how closely wide-area SCION networks can approach single-digit-microsecond accuracy beyond the baseline established in the SIDN Labs deployment, we use a reference time source from the METAS and Switch White Rabbit Core network [20], which disseminates the Swiss official time scale UTC(CH). A dedicated line from this core network to ETH Zurich provides the primary reference, which is now served directly from ETH Zurich into the SCION production network in Switzerland. Ongoing work with FBG in Zurich will use production-quality hardware (see Section III-B) to evaluate this METAS-based Clockwire service over more than 500 paths between FBG's AS and ETH Zurich. Preliminary evaluation within the local ETH network shows promising results (Figure 6). The longer-term goal is to establish Clockwire as a trusted wide-area timing service for critical infrastructure in Switzerland.

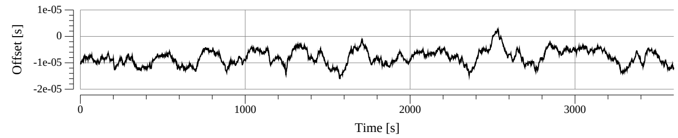

Fig. 5. Offset stability of multipath synchronization between SIDN Labs and ETH Zurich using a virtualized time server without hardware timestamping.

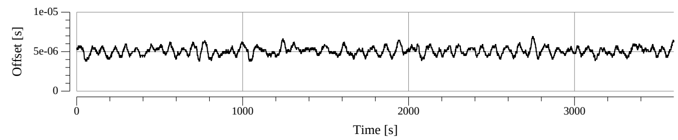

Fig. 6. Offset stability within the ETH local-area network with commodity hardware serving the METAS reference time over Clockwire.

V. Related Work

We build upon and extend several aspects of multipath clock synchronization covered in prior works.

Most papers assume the availability of multiple communication paths, but do not focus on how this property can be obtained, especially on the public Internet, or assume techniques such as SD-WAN are used. A notable exception are multipath NTP and PTP [21], also described in RFC8039 [22], where end hosts send and receive traffic using several IP addresses and rely on intra-AS load-balancing to expose multiple intra-AS paths through the same sequence of ASes. Our work, in contrast, relies on SCION for multipath communication. SCION causes less vendor lock-in than SD-WAN and offers control even over shared infrastructure (see Section II). SCION also has several advantages compared to the aforementioned intra-AS load-balancing technique: SCION does not require multiple endpoint addresses (which is problematic, given the IP address scarcity), and gives end hosts explicit path control and better visibility over the available paths and their properties. Also, SCION exposes inter-AS paths, i.e., paths that, unlike in the load-balancing approach, traverse different sets of ASes, leading to more diverse paths and better protection against attacks, as untrusted ASes can be avoided. Net Insight's solution [23] and the upcoming Enhanced Partial Timing Support standard operate as an overlay network over existing IP infrastructure. Although this offers more control than the public Internet, overlay networks are still affected by changes in (and attacks on) the underlying network, requiring detection of and reaction to such events. In SCION, on the other hand, nodes benefit from explicit path choice, strong routing security, and several extensions (see Section II-B). Moreover, SCION's multi-operator model likely leads to more path diversity.

The "slave diversity" paper [10] highlights that multipath time synchronization is especially useful under changing network conditions and explores strategies for merging timing information received over different paths, such as averaging the received information, or preferring the path with least congestion. The use of Kalman filters has also been suggested for this purpose [24], as have Byzantine fault tolerant algorithms [25]. Because the latter ignore measurement outliers, improving robustness and constraining the influence of attackers, we use this approach in Clockwire. The detection of delay attacks done by PTPsec [26] also relies on multipath communication and on the availability of at least one symmetric path; both properties can be achieved with SCION.

VI. Conclusion

We present Clockwire, a resilient clock synchronization system built on SCION. Clockwire combines timing information from multiple reference time servers over multiple network paths and can further integrate local sources such as GNSS, thus reducing dependence on any single timing source. Clockwire clients select from dozens or even hundreds of symmetric network paths and use them concurrently over shared SCION production networks. The system can be deployed on commodity hardware or on our purpose-built hardware platform with a software-controlled cross-timestamping mechanism across independently synchronized hardware clocks. This enables a range of clock-fusion strategies with full transparency down to the hardware level.

Together, these capabilities provide a practical foundation for greater flexibility and cost-effectiveness in wide-area time distribution without compromising the indispensable properties of resilience, accuracy, and end-to-end control. Ongoing pilot deployments are advancing the system toward production readiness in real-world environments.

Acknowledgments

We thank SIDN Labs, our first pilot partner; Frankfurter Bankgesellschaft (Schweiz) AG for financial support and continued collaboration on the Clockwire testbed deployment; the Federal Institute of Metrology METAS and Switch for providing a high-precision reference time source; and ETH Zurich's Data Networks Group and Network Installations and Planning Group for support in setting up the Clockwire node at ETH Zurich. This work was supported by ETH Zurich, the Zurich Information Security and Privacy Center (ZISC), armasuisse Science and Technology, and the Werner Siemens Stiftung (WSS) Centre for Cyber Trust at ETH Zurich. AI-based tools were used to check and improve the spelling, grammar, and style of this text.

References

[1] M. Wyss, M. Frei, J. Kwon, and A. Perrig, "Low-cost and Robust Global Time Synchronization," in *IEEE S&P*, 2025.

[2] L. Chuat, M. Legner, D. Basin, D. Hausheer, S. Hitz, P. Müller, and A. Perrig, *The Complete Guide to SCION. From Design Principles to Formal Verification*. Springer International Publishing AG, 2022, https://link.springer.com/book/10.1007/978-3-031-05288-0.

[3] J. Martin, J. Burbank, W. Kasch, and P. D. L. Mills, "Network Time Protocol Version 4: Protocol and Algorithms Specification," IETF, RFC 5905, 2010.

[4] IEEE Standards Association, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," IEEE Std 1588-2019, 2020.

[5] M. Wyss, R. Meier, L. Romá, C. Krähenbuhl, A. Perrig, and V. Lenders, "On building secure wide-area networks over public internet service providers," in *CyCon*, 2024.

[6] SCION Association, https://www.scion.org/association.

[7] C. de Kater, N. Rustignoli, and S. Hitz, "SCION Control Plane," Internet Engineering Task Force, Internet-Draft draft-dekater-scion-controlplane-17, Apr. 2026, https://datatracker.ietf.org/doc/draft-dekater-scion-controlplane/17/ (Work in Progress).

[8] C. de Kater, N. Rustignoli, J.-C. Hugly, and S. Hitz, "SCION Data Plane," Internet Engineering Task Force, Internet-Draft draft-dekater-scion-dataplane-14, Apr. 2026, https://datatracker.ietf.org/doc/draft-dekater-scion-dataplane/14/ (Work in Progress).

[9] SCION Association, "SCION for Critical Infrastructure," 2025, https://www.scion.org/wp-content/uploads/2025/03/Fact-sheet-SCION-for-criticalinfrastructure_March2025.pdf.

[10] T. Mizrahi, "Slave diversity: Using multiple paths to improve the accuracy of clock synchronization protocols," in *ISPCS*, 2012.

[11] C. Krähenbühl, M. Wyss, D. Basin, V. Lenders, A. Perrig, and M. Strohmeier, "FABRID: Flexible Attestation-Based Routing for Inter-Domain networks," in *USENIX Security*, 2023.

[12] K. Wüst, G. Giuliari, M. Legner, J.-P. Smith, M. Wyss, J. Bachmann, J. Garcia-Pardo, and A. Perrig, "Hummingbird: Fast, flexible, and fair inter-domain bandwidth reservations," in *ACM SIGCOMM*, 2025.

[13] M. Lichvar and T. Mizrahi, "Network Time Protocol Version 5," Internet Engineering Task Force, Internet-Draft draft-ietf-ntp-ntpv5-08, Mar. 2026, https://datatracker.ietf.org/doc/draft-ietf-ntp-ntpv5/08/ (Work in Progress).

[14] IEEE Standards Association, "Standard for Client Server Precision Clock Synchronization Protocol," https://standards.ieee.org/ieee/1588.1/11644.

[15] D. F. Franke, D. Sibold, K. Teichel, M. Dansarie, and R. Sundblad, "Network Time Security for the Network Time Protocol," IETF, RFC 8915, 2020.

[16] M. Lichvar and A. Malhotra, "NTP Interleaved Modes," IETF, RFC 9769, 2025.

[17] Poul-Henning Kamp, https://github.com/bsdphk/ntimed.

[18] I. Hadžić and D. R. Morgan, "On packet selection criteria for clock recovery," in *ISPCS*, 2009.

[19] O. Deutsch, N. R. Schiff, D. Dolev, and M. Schapira, "Preventing (network) time travel with Chronos," in *NDSS*, 2018.

[20] A. Jallageas, D. Husmann, J. Morel, F. Mauchle, R. Mathis, L. Nagy, and A. Jaquier, "Long-haul dissemination of the Swiss national time-scale UTC(CH) in a telecommunication fiber network via White Rabbit," in *Journal of Optical Communications and Networking*, 2025.

[21] A. Shpiner, Y. Revah, and T. Mizrahi, "Multi-path time protocols," in *ISPCS*, 2013.

[22] A. Shpiner, R. Tse, C. Schelp, and T. Mizrahi, "Multipath Time Synchronization," IETF, RFC 8039, Dec. 2016.

[23] Net Insight, https://netinsight.net/sync/sync-solution/.

[24] G. Giorgi and C. Narduzzi, "Kalman filtering for multi-path network synchronization," in *ISPCS*, 2014.

[25] E. Kyriakakis, K. Tange, N. Reusch, E. O. Zaballa, X. Fafoutis, M. Schoeberl, and N. Dragoni, "Fault-tolerant clock synchronization using precise time protocol multi-domain aggregation," in *ISORC*, 2021.

[26] A. Finkenzeller, O. Butowski, E. Regnath, M. Hamad, and S. Steinhorst, "PTPsec: Securing the Precision Time Protocol Against Time Delay Attacks Using Cyclic Path Asymmetry Analysis," in *IEEE INFOCOM*, 2024.