# Toward Global Latency Transparency

Cyrill Krähenbühl, Seyedali Tabaeiaghdaei, Simon Scherrer, Matthias Frei, Adrian Perrig
*Department of Computer Science, ETH Zürich, Switzerland*

***Abstract*—A recent advance in networking is the deployment of path-aware multipath network architectures, where network endpoints are given multiple network paths to send their data on. In this work, we tackle the challenge of selecting paths for latency-sensitive applications. Even today's path-aware networks, which are much smaller than the current Internet, already offer dozens and in several cases over a hundred paths to a given destination, making it impractical to measure all path latencies to find the lowest latency path. Furthermore, for short flows, performing latency measurements may not provide benefits as the flow may finish before completing the measurements. To overcome these issues, we argue that endpoints should be provided with a latency estimate *before* sending any packets, enabling latency-aware path choice for the first packet sent. As we cannot predict the end-to-end latency due to dynamically changing queuing delays, we measure and disseminate the *propagation* latency, enabling novel use cases and solving concrete problems in current network protocols. We present the Global Latency Information Dissemination System (GLIDS), which is a step toward global latency transparency through the dissemination of propagation latency information.**

## I. Introduction

With the deployment of path-aware network (PAN) architectures such as SCION [1], new challenges and opportunities arise for latency-sensitive applications such as video conferencing, online gaming, holographic communication, or the tactile Internet [2]. In contrast to today's Internet, which only offers a single, typically cost-optimized, path, a PAN can offer multiple path options, providing advantages to latency-sensitive applications since endpoints can select the shortest-latency path. In the current deployment of SCION, we witness that a large number of distinct paths are available, with many destinations having over 100 different paths. With the steady expansion of the commercial SCION network, we expect this number to further increase.

For latency-sensitive applications, a research challenge thus emerges on which path to use. The path length in terms of AS hops is unfortunately a weak predictor of the end-to-end latency [3]. Path probing with active measurements would waste the benefit in the case of dozens, or even hundreds, of available paths. In particular, single-packet request-response protocols, such as DNS, would not permit any probing for achieving low-latency operation.

In this work, we explore an approach that adds *propagation* latency information to PAN path information, enabling an endpoint to compute an estimate for the end-to-end propagation latency. The reasons for disseminating the propagation latency instead of attempting to predict the experienced end-to-end latency, which also comprises the transmission, processing, and queuing latency, are as follows. Transmission and processing latencies are typically negligible compared to propagation and queuing, and often exhibit little variance. Since the amount and nature of cross traffic is usually unpredictable, an accurate estimation of the queuing latency is not possible in most cases. On the other hand, propagation latency is a useful metric, as it enables the computation of the experienced queuing latency, is typically static, and can be measured without relying on a model. Finally, as we will outline below, for use cases such as efficient path probing and improving the fairness of congestion control algorithms, knowledge of the propagation latency provides significant benefits.

We propose the Global Latency Information Dissemination System (GLIDS) for estimating propagation latency information of end-to-end paths in an inter-domain setting. We study the research problem of how to achieve high accuracy for latency estimates while minimizing the network overhead. Furthermore, since GLIDS is based on SCION, where partial paths in the form of path segments are combined into end-to-end paths, we ensure that end-to-end propagation latency estimation is possible for all path segment combinations.

One goal of GLIDS is to find the lowest latency path in a multipath network with a large number paths. For long-lived latency-critical flows, a sender will (continuously) probe available paths and switch to the lowest end-to-end latency path available. Knowledge of the propagation latency gained through GLIDS provides the sender with a clear order and cutoff latency for path probing instead of relying on heuristics which may not find the lowest experienced latency path.

For short-lived flows that finish in a single round trip, e.g., consisting of a single request and response packet pair or where all data fits into the initial congestion window, performing latency measurements to decide where to send a packet may take as long as, or even longer than, the flow itself and thus negate any benefits of the measurement. Even for longer-lived flows, knowledge of the propagation latency gained from GLIDS is valuable to make an informed path selection when sending the *first* packet.

Another use of GLIDS is to improve delay-based congestion control algorithms (CCAs), which adjust a sender's congestion window based on the measured latency inflation as a signal of queuing on intermediate nodes. BBR is a delay-based CCA which has shown to be unfair to competing loss-based CUBIC CCA flows, due to overestimating the propagation latency [4], [5]. By providing a propagation latency estimate, GLIDS enables BBR to infer the current latency inflation and overcome this unfairness.

Compared to the current Internet, the key ingredient en-

ISBN 978-3-903176-63-8 © 2024 IFIP

abling GLIDS in the SCION path-aware Internet architecture is the stability of disseminated paths. GLIDS leverages this stability to directly use the latency information disseminated through the control-plane for path selection and thus provides propagation latency transparency. An extended version of this paper, including a background section on SCION, extended related work, and details on the experiment setup, can be found on arXiv [6]. The main contributions of this work are:

- Design of the scalable system GLIDS for estimating propagation latency of end-to-end paths in SCION.
- Evaluating GLIDS through the SCIONLab testbed, emulation in a Mininet-based BBR testbed, and simulations on real-world topologies.

## II. Motivation and Challenges

In this section, we highlight issues prevalent in existing latency prediction systems, motivate the need of GLIDS through three concrete use cases, and discuss design challenges.

### A. Prior Work on Latency Prediction

Latency prediction for intra- and inter-domain networks has been extensively studied in the past. Latency prediction approaches can be separated into deterministic approaches that distribute aggregated *measured latencies*, and model-based approaches that use statistical *latency models* to infer latency based on network delay monitoring between various vantage points in the network [7], [8], [9]. Due to challenges such as asymmetric routing, unpredictable paths, dynamic load balancing, and lack of transparency on the actually used inter-domain forwarding paths, recent advancements in latency prediction have been focused on model-based approaches.

*1) **Intra-domain Latency Prediction**:* In the space of intra-domain networking, there has been much work on low-latency networking [10] and latency prediction (often in the context of data centers) [11], [12], [13]. This shows the usefulness of latency prediction in a controlled environment to enhance application performance, reduce overhead, and potentially minimize cost in terms of hardware or computational resources.

*2) **Inter-Domain Latency Prediction**:* In the space of inter-domain networking, low-latency networking and latency prediction is hindered by a multitude of factors. Since forwarding devices are not controlled by a single entity, they may behave unexpectedly. Additionally, external factors, such as varying queuing delay caused by background traffic, impact the prediction accuracy. Moreover, the lack of transparency in today's Internet reveals little about the used network paths, further increasing the uncertainty of any form of latency predictions. Nevertheless, there has been extensive research in latency prediction in inter-domain networking, demonstrating the desire for latency prediction [7], [14], [15], [8], [9].

### B. Use Cases

With the emergence of inter-domain path-aware network architectures, some of the issues prevalent in latency prediction can be addressed. Path-aware networks enable path transparency, which allows endpoints to know the network paths of their traffic, and consequently make use of latency-related information about these paths. Furthermore, path-aware *multipath* networks provide multiple (partially) disjoint network paths for an endpoint to choose from. Endpoints can thus locally optimize paths based on a metric, e.g., latency. We argue that recent developments in inter-domain path-aware networks allow users to accurately *estimate* propagation delay through concrete measurements, enabling innovative and novel use cases. Concretely, we present three use cases.

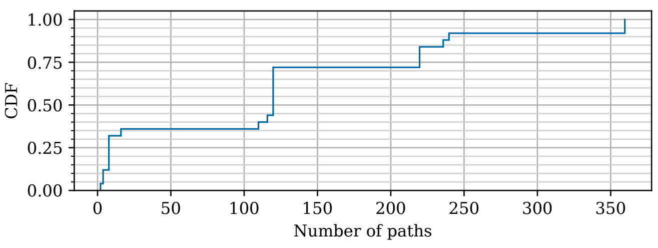

Fig. 1: The number of distinct paths to all reachable ASes in the commercial SCION network from a single vantage point.

*1) **Delay-Based Congestion Control**:* The first use case is providing propagation delay information to delay-based congestion control algorithms to accurately derive the current latency inflation on a path and infer the level of congestion. The knowledge of propagation delay is thus critical for the competition of the delay-sensitive BBR algorithm with the traditional loss-based CUBIC algorithm. In this competition, BBR has been found to be unfair towards CUBIC in terms of capacity sharing, because the sending rate of the BBR flows is directly proportional to their estimate of the path propagation delay. Crucially, BBR flows overestimate this propagation delay as a result of queuing caused by competing CUBIC flows, and therefore maintain an excessively large sending rate [4], [5]. This unfairness can be eliminated with explicit knowledge of path propagation delay, as offered by GLIDS.

*2) **(First-Packet) Latency Estimate**:* The next two use cases illustrate the importance of accurate latency information in path-aware networks that offer multiple paths to choose from. For short-lived connections, e.g., query-response protocols such as DNS, it is essential that an RTT estimate is available *before* sending the first packet. There is no benefit for an endpoint to actively measure the latency if all relevant data can be transmitted in the first reply, i.e., the reply may arrive before the measurements are finished.

*3) **Efficient Probing**:* Finally, we argue that a propagation latency estimate is *necessary* to efficiently probe a multipath network for low-latency paths. Consider an endpoint establishing and maintaining a long-lived low-latency connection. If only a few network paths are available, the endpoint could simply continuously measure the paths and switch to a lower latency path if available. However, this approach does not scale if the network offers hundreds of potential paths, i.e., in a massive multipath network, as this would incur substantial processing, time, and bandwidth overhead. The current commercial SCION network provides a concrete example of such a scenario. Figure 1 depicts the number of different paths we can observe from ETH Zürich to 30 ASes located in 5 different ISDs. Although the SCION network is small compared to the Internet, the median number of paths to each destination is over 100, and we expect this number to further increase with the expansion of the commercial SCION Internet. The endpoint could heuristically select and probe a subset of paths but this does not guarantee finding latency-optimal paths. We discuss a concrete algorithm for efficient path probing based on GLIDS in Section III-C.

### C. Challenges

In this section, we list the most important challenges encountered in designing a latency transparency system.

*1)* ***Variable Delays:*** One major challenge in latency estimation is to correctly model all variable delays in the network [7]. The dominating variable delay is typically queuing delay due to cross traffic filling up a packet queue on the path. The packet processing and transmission delays are usually negligible in comparison to the propagation and queuing delays, but may be included in a latency measurement or calculated based on the link bandwidth. Since the queuing delay cannot be predicted, our work focuses on the predictable part of the end-to-end latency, in particular the propagation delay.

*2)* ***Disclosing Internal Topology:*** While entities can enhance latency estimates by revealing detailed information about their internal topologies, this might reveal sensitive information to a competitor. It is thus imperative that participating entities can decide how much information is revealed.

*3)* ***Load Balancing and Variable Routes:*** Internet traffic is rerouted for various reasons, e.g., economic impact, SLAs, and bottlenecks. This can happen for inter-domain paths or for intra-domain paths, e.g., due to traffic engineering or failing links. Accurate latency estimation is challenging under these circumstances and requires regularly updated measurements.

## III. System Design

We propose a latency transparency system that measures and distributes the propagation latency of inter-domain paths at Internet scale and present an efficient path probing algorithm for multipath networks. In designing such a system, we make use of the facts that latency is an additive metric, and any inter-domain path can be split into several *intra-domain* paths and *inter-domain* links connecting them. The propagation latency of an inter-domain path can thus be computed by accumulating the propagation latencies of all intra-domain paths and inter-domain links. Therefore, we divide GLIDS into two subsystems: (1) the *measurement system* that accurately measures the latency of intra-domain paths and inter-domain links, and (2) the *dissemination system* that globally disseminates latency information.

### A. Performing Latency Measurements

In GLIDS, we focus on measuring propagation delay, instead of modeling queuing delay. Note that depending on the measurement method, the propagation delay measurement may include the processing and transmission delay, but in practice, they are typically negligible in comparison to propagation and queuing delay. Hence, GLIDS requires a participating AS to be able to measure the propagation latency of intra-domain paths between their own border routers and of the links connecting them to neighboring ASes' border routers as shown in Figure 2.

There exist a wide variety of ways to measure latency, which can generally be separated into three groups [16]: (1) traditional network measurements [17], [18], (2) SDN-based measurements [19], [20], and (3) telemetry-based measurements [21], [22], each with different tradeoffs. Since we are interested in the *one-way propagation* latency, the network operator must ensure that queuing delay is factored out, e.g., via packet prioritization or exclusion of "packet queue time", and that potential path asymmetry is taken into consideration, e.g., using one-way latency measurements with synchronized clocks instead of RTT measurements. Additionally, multiple paths with varying latency may exist between border routers due to redundancy, load balancing, or traffic engineering, and paths may change over time. The network operator should measure all possible paths and re-measure changed paths to revoke the out-of-date path segment and re-disseminate it. Optionally, the network operator may also enhance the measurement with the used measurement methodology and a level of measurement (un-)certainty, e.g., assigning a higher level of certainty if a more sophisticated latency measurement approach is used.

### B. Disseminating Latency Information

To calculate the latency of an inter-domain path, the endpoints must be provided with the latencies of its constituent intra-domain paths and inter-domain links. GLIDS achieves this in a scalable fashion through path exploration and dissemination. In the exploration phase, each AS on an inter-domain path encodes the latency information of its AS hop in the forwarded path segment. In the dissemination phase, the endpoint retrieves the path segments with the included latency information. Note that GLIDS uses an opt-in approach and an AS can choose to disclose latency information with appropriate granularity based on the desired level of topology secrecy. The privacy aspects are discussed in more detail in Section V-A.

The latency information of each AS hop consists of the propagation latency of the intra-domain path between the ingress and the egress border routers (e.g., SF to NJ in Figure 2) as well as the propagation latency of the inter-domain links between the border routers of the neighboring ASes (e.g., SF to LA). Due to possibly asymmetric latencies, the AS encodes the latency in both directions. If the exact intra-domain path each packet can take is not predictable, e.g., due to load-balancing or backup routes, the AS must disseminate the minimum propagation latency of all possible routes—which is necessary for efficient path probing, see Section II-B. In addition to this minimum latency, an AS

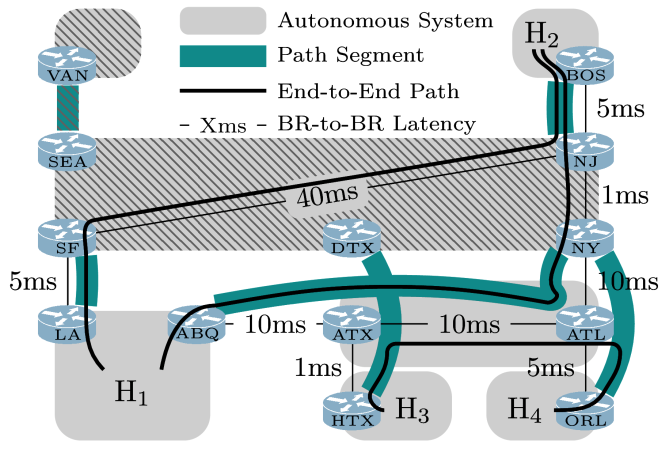

Fig. 2: Topology where four hosts ($H_{1-4}$) combine path segments to end-to-end paths and leverage GLIDS to select paths. Diagonal lines indicate core ASes and core path segments.

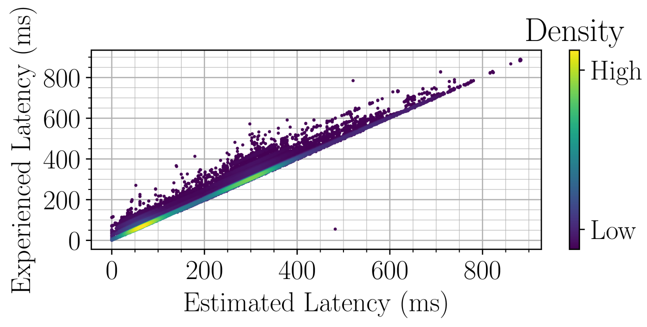

Fig. 3: Difference between the estimated and the experienced propagation latency in SCIONLab.

can optionally provide more information, such as the maximum propagation latency among these paths or a latency distribution. Alternatively, the network operator could make such intra-domain paths (including their latency) accessible via FABRID policies [23] and allow endpoints to explicitly select a specific intra-domain path.

*1) **Disseminating Latency in Hierarchical Architectures:*** In hierarchical routing architectures like SCION, path segments need to be combined to construct full inter-domain paths. Hence, not only do we need to add the latency information in each path segment, but we also need to ensure that endpoints can reconstruct the latency information of all possible segment combinations. Figure 2 shows an example where the lack of intra-AS latency information between SF and NJ would cause $H_1$ to incorrectly select the path through LA (via SF, NJ, BOS) to communicate with $H_2$ even though the alternative path through ABQ (via ATX, ATL, NY, NJ, BOS) has a smaller propagation latency. Hence, GLIDS must disseminate the latency information between all border router pairs of ASes at the path segment junctions. To satisfy this requirement of SCION's hierarchical routing with two levels, i.e., core and intra-ISD routing, we propose two different information dissemination mechanisms, one for each level.

In core routing, where the topology is densely connected and routing messages are flooded to a subset of neighbors, we need a mechanism with small overhead per routing message to achieve scalability. To that end, each AS only encodes the latency for the intra- and inter-domain path that the routing message traversed. This is possible since core-path segments are always combined with intra-ISD-path segments, which will contain the necessary latency information to interfaces that are not traversed by the core-path segment.

In intra-ISD routing, where routing messages are only forwarded from providers to customers and the topology is typically sparse, more latency information can be added to routing messages without introducing significant overhead. A trivial approach is the addition of latency information between the egress interface of the routing message and all other interfaces. However, this approach wastes network bandwidth, since some latency information will be duplicated, i.e., two combined path segments will both contain the latency information between two interfaces. To prevent this, only the latency information between the egress interface and interfaces which have a lower AS-local identifier is added. This ensures that any two intra-ISD-path segments, or any core and any intra-ISD-path segment can be combined. For example, in Figure 2, hosts $H_3$ and $H_4$ need the intra-AS latency between ATX and ATL even though neither segment contains this link.

*2) **Scalability:*** GLIDS achieves scalability using three mechanisms. First, it disseminates the minimum propagation latency of a path, which does not change frequently, and thus does not require frequent re-dissemination. Second, the existing path-segment establishment and dissemination process is used to disseminate the latency information without introducing additional messages. Third, it leverages the observation that core path segments are always combined in their entirety with intra-ISD path segments, thus no additional intra-AS latency information needs to be added to the core path segments, but only to the intra-ISD path segments.

*3) **Obtaining Full Inter-Domain Path Latency:*** Once provided with the path segments to construct full inter-domain paths, endpoints compute the full path latency by accumulating latencies of constituent AS hops. Note that an endpoint may only receive partial latency information if some on-path ASes do not reveal this information. In such a case, the endpoint locally decides whether to use the partial information, e.g., by assuming a propagation delay of zero for the missing latency information, or to discard the incomplete latency information.

## *C. Efficient Path Probing*

We propose an efficient path probing algorithm based on GLIDS, which works as follows: (1) sort all paths based on their propagation latencies (assuming zero latency for path segments with missing latency information), (2) measure them one-by-one (or batch-wise) and remember the smallest measured latency, and (3) stop as soon as the propagation delay of the next path is higher than the smallest measured latency. Since all following paths *must* have higher latency, we can terminate the algorithm early. A requirement is that, if a path has multiple intra- or inter-domain links with different propagation delays, the system always returns the *minimum* value. Otherwise, the algorithm might discard and thus not probe a potential candidate path. The number of probed paths depends on the distribution of latencies, e.g., a higher variance in propagation delays leads to an earlier termination.

## IV. Implementation and Evaluation

We prove the feasibility of GLIDS by implementing it on the SCIONLab testbed, evaluate the impact of propagation latency knowledge on the fairness of delay-based congestion control, and show the benefits of GLIDS's first packet latency estimation through simulation on Internet-scale topologies.

### A. Implementation and Deployment in SCIONLab

We evaluate GLIDS in the SCIONLab testbed [24]. Our implementation uses RTT measurements, i.e., the minimum measured RTT over the span of 60 s, to infer propagation latency between AS interfaces. The experienced latency is then measured by taking the median of 5 ping measurements.

Figure 3 shows that inferring propagation latency based on the RTT, which is arguably the simplest measurement approach, provides a reasonable approximation of the propagation latency since less than 5% of measurements experienced a lower latency than the advertised propagation latency. The overall experienced latency, with a median of 234 ms and a 95th percentile of 551 ms, is relatively high since the SCIONLab network partially consists of overlay links, which may lead to longer paths, and since the SCIONLab nodes are globally distributed in Europe, the United States, and East Asia. However, note that SCIONLab, as a research network, experiences little congestion. In networks carrying large amounts of data and experiencing more congestion, these results might differ substantially and require more sophisticated latency measurement systems.

In addition to the feasibility of the system, we measure the latency reduction of an endpoint that chooses the path with the lowest estimated propagation latency over the default path. Figure 4 shows that using GLIDS, 50% of endpoints reduced the latency by over 40 ms, or 25% relative to the original value. In 10% of the cases, endpoints reduced the latency by over 200 ms, or 70%. These results show that a multipath-capable network with inter-domain propagation latency estimate can indeed significantly reduce the experienced latency.

### B. Delay-Based Congestion Control

In the following, we demonstrate how information about path propagation delay can improve the deployment effects of delay-based congestion-control algorithms (CCAs). In particular, we show that the knowledge of path propagation delay improves the fairness of the delay-sensitive BBR CCA [25] toward the traditional loss-based CUBIC CCA [26].

*1) **Setup**:* For our evaluation, we emulate the competition among 10 flows on a 100 Mbps bottleneck link using Mininet [27]. Each flow is using a path with 20 ms one-way propagation delay, composed of 10 ms propagation delay on non-shared links and 10 ms propagation delay on the shared bottleneck link. The non-shared links and the shared bottleneck link are intermediated by a switch with a queue size that corresponds to 1.5 bandwidth-delay products of the network path. In this setup, we measure the capacity share obtained by all BBR flows together, depending on how many of the 10 flows adopt BBR, while the remaining flows adopt CUBIC. We also distinguish two different BBR versions, namely the official BBR version, and a variant named "informed BBR" that we implemented by adapting the BBR source code: While the official BBR attempts to estimate path propagation delays by regular probing, informed BBR works with actual propagation delays, which are known when using GLIDS.

*2) **Results**:* Figure 5 presents the results of this experiment. For all tested configurations, the BBR capacity share is strictly above the dotted gray line that marks the proportional capacity share; hence, BBR is unfair towards CUBIC in every evaluated case. However, this unfairness is particularly pronounced if fewer than 5 flows adopt the official BBR version.

Crucially, this unfairness stems from the RTT-probing behavior of the BBR flows, which try to discover the propagation delay by emptying on-path buffers with sharp contractions of their sending rate, and use the resulting measurement to adjust their congestion window [25]. While this probing behavior indeed allows discovering the propagation delay if

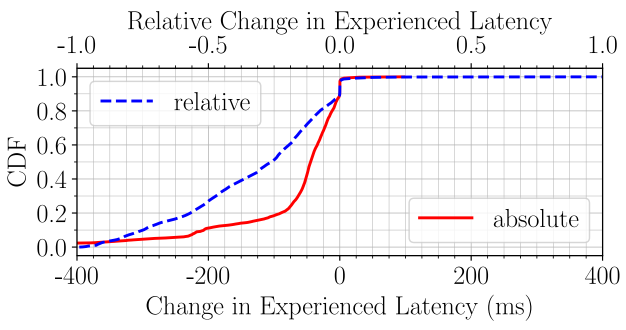

Fig. 4: Latency reduction (in relative values or ms) when selecting the path with the lowest propagation latency.

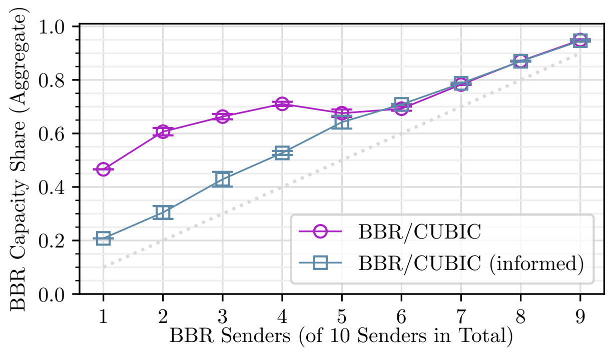

Fig. 5: Bandwidth-sharing fairness between BBR and CUBIC flows, with and without explicit propagation delay knowledge.

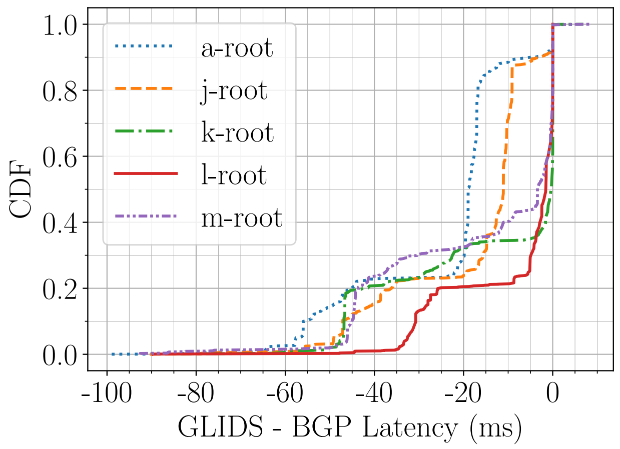

Fig. 6: Comparison of lower-bound latencies from probes to DNS root servers between GLIDS and BGP.

only BBR flows share the bottleneck link [28], competing CUBIC flows preserve buffer utilization when the BBR flows are probing, and thus inflate the propagation-delay estimate of the BBR flows [4], [5]. As a result, the BBR flows maintain an excessively large congestion window, resulting in an unfairly high sending rate. Notably, this effect arises only if the CUBIC flows are numerous enough to keep up buffer utilization in the 200 ms time window when the BBR flows are probing; in our setting, fewer than 4 CUBIC flows cannot fill the buffer during this RTT probing, and thus do not distort the BBR propagation-delay estimate.

Clearly, the propagation-delay estimate is never distorted for the *informed* BBR version, as this version learns the actual propagation delay from GLIDS. This knowledge thus eliminates the overdimensioning of the BBR congestion window, and improves the fairness of BBR towards CUBIC.

### C. *Large-scale Simulations*

In this section, we analyze the benefits of GLIDS for first packet latency estimation, in particular DNS resolution. We perform large-scale simulations on real-world Internet topologies from CAIDA [29] to compare the performance of GLIDS with BGP-based (shortest AS path) routing. For this evaluation, we simulate BGP using SimBGP [30], and SCION using a simulator based on ns-3 [31], to find inter-domain paths in the current Internet and in SCION+GLIDS. The latencies between routers are given by their great circle distance, calculated from their geographical location.

Based on these simulations, we analyze by how much GLIDS could reduce this lower-bound (great circle) latency between clients and DNS root servers. Figure 6 illustrates the difference between the simulated latency in SCION and BGP, showing that SCION's latency-aware routing and path selection can reduce the lower-bound latency to the 5 root servers in our topology for at least 40% of probes. Furthermore, this latency deflation is at least 20 ms for 20% of probes to any of the root servers. However, the latency to root servers from a negligible portion of probes can be inflated by less than 10 ms due to suboptimal choice of destination AS ingress interface. This is caused by a lack of information on the relative position of the destination root server in the destination AS.

## V. Discussion

This section focuses on the practical aspects of achieving latency transparency through GLIDS. In particular, we focus on GLIDS's incremental deployment model, potential hurdles for adoption, and its privacy aspects. Furthermore, we discuss the veracity of the disseminated latency information.

### A. *Deployment Model*

A crucial aspect of any new Internet protocol is deployability [32], i.e., to ensure compatibility with existing and legacy systems in a partial deployment while providing clear incentives for early adopters. Ideally, existing latency measurement mechanisms and hardware can be reused for internal latency measurements, allowing GLIDS to be deployed at low cost.

GLIDS can be incrementally deployed by having only a subset of ASes include latency information in their routing messages. While this does not yet provide full propagation latency transparency for all Internet users, as long as ASes that are topologically close to the communicating endpoints offer GLIDS, partial latency information inferred by endpoints can be used for latency-based path optimization. Hence, two endpoints purchasing Internet connectivity from Tier 2 ISPs can achieve propagation latency transparency if their ISPs support GLIDS and are customers of the same GLIDS-enabled Tier 1 AS. Even if no Tier 1 AS supports GLIDS, ISPs can leverage peering and transit links to other ISPs to locally achieve propagation latency transparency.

The incentive of an early GLIDS adopter is to attract network traffic by providing propagation latency information. Note that even if the early adopter does not offer better performance than its competitors, simply the presence of the latency information may lead to endpoints sending traffic on these paths to benefit from the various use cases presented in Section II-B. This holds true especially for delay-based congestion control and first-packet latency estimation, while the usefulness of efficient probing increases together with the adoption rate since more paths can be pruned.

Another important aspect of GLIDS is privacy, since the participating ASes reveal propagation latencies of internal paths. We argue that privacy is often not a concern in GLIDS since it typically does not reveal more sensitive information, such as the exact configuration of internal routers and links, compared to the existing SCION paths. Furthermore, each AS can decide not to reveal certain (privacy-sensitive) latencies. Finally, even in today's Internet, propagation latencies can often be inferred through measurements between different vantage points (although typically with a lower precision).

### B. *Veracity of Latency Information*

One important aspect of GLIDS is ensuring the veracity of the disseminated latency information. A dishonest AS may disseminate wrong information for a financial gain. Artificially increasing the disseminated propagation latencies is typically less problematic since users will send their latency-sensitive traffic on alternative paths. Artificially decreasing the disseminated propagation latencies may attract users to send their latency-sensitive traffic on a sub-optimal path. In GLIDS, the veracity of latency information is protected by including a timestamped signature from the AS that provides the latency information in the respective routing message. Hence, GLIDS provides non-repudiation to ensure that a misbehaving AS can be punished while preventing framing attacks. Note that while this provides the necessary building blocks to implement a system to detect misbehaving ASes, the full design of such a system is beyond the scope of this paper.

## VI. Conclusion

GLIDS offers exciting new possibilities for inter-domain networks and moves us closer to the prospect of global latency transparency. We show that path-aware networks solve current network problems but also highlight novel challenges encountered in such networks and how to overcome them.

## VII. Acknowledgments

We thank the anonymous reviewers for their insighful feedback and suggestions. We gratefully acknowledge support for this project from the Werner Siemens Stiftung (WSS) Centre for Cyber Trust at ETH Zurich, from armasuisse Science and Technology, and from the European Union's Horizon 2020 research and innovation programme under grant agreements No 825310 and 825322. Finally, we thank Marc Wyss, Giacomo Giuliari, and Marc Frei for their valuable feedback.

## References

[1] C. Krähenbühl, S. Tabaeiaghdaei, C. Gloor, J. Kwon, A. Perrig, D. Hausheer, and D. Roos, "Deployment and scalability of an inter-domain multi-path routing infrastructure," in *Proceedings of the International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2021.

[2] F. G. on Technologies for Network 2030, "Representative use cases and key network requirements for network 2030," tech. rep., International Telecommunication Union (ITU), Jan. 2020.

[3] A. Fei, G. Pei, R. Liu, and L. Zhang, "Measurements on delay and hop-count of the internet," Sept. 1998.

[4] M. Hock, R. Bless, and M. Zitterbart, "Experimental evaluation of BBR congestion control," in *Proceedings of the IEEE Conference on Network Protocols (ICNP)*, pp. 1–10, 2017.

[5] A. Mishra, W. H. Tiu, and B. Leong, "Are we heading towards a BBR-dominant internet?," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, pp. 538–550, 2022.

[6] C. Krähenbühl, S. Tabaeiaghdaei, S. Scherrer, M. Frei, and A. Perrig, "GLIDS: A global latency information dissemination system," 2024. arXiv:2405.04319 [cs.NI].

[7] F. Tabatabaeimehr, M. Ruiz, C.-Y. Liu, X. Chen, R. Proietti, S. J. B. Yoo, and L. Velasco, "Cooperative learning for disaggregated delay modeling in multidomain networks," *IEEE Transactions on Network and Service Management*, vol. 18, pp. 3633–3646, Sept. 2021.

[8] D. Perdices, D. Muelas, I. Prieto, L. de Pedro, and J. E. L. de Vergara, "On the modeling of multi-point RTT passive measurements for network delay monitoring," *IEEE Transactions on Network and Service Management*, vol. 16, pp. 1157–1169, Sept. 2019.

[9] F. Krasniqi, J. Elias, J. Leguay, and A. E. C. Redondi, "End-to-end delay prediction based on traffic matrix sampling," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, July 2020.

[10] W. Li, J. Liu, S. Wang, T. Zhang, S. Zou, J. Hu, W. Jiang, and J. Huang, "Survey on traffic management in data center network: From link layer to application layer," *IEEE Access*, vol. 9, pp. 38427–38456, 2021.

[11] E. H. Bouzidi, D.-H. Luong, A. Outtagarts, A. Hebbar, and R. Langar, "Online-based learning for predictive network latency in software-defined networks," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018.

[12] S. Alesawi, M. Nguyen, H. Che, and A. Singhal, "Tail latency prediction for datacenter applications in consolidated environments," in *Proceedings of the International Conference on Computing, Networking and Communications (ICNC)*, Feb. 2019.

[13] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen, Z.-W. Lin, and V. Kurien, "Pingmesh: A large-scale system for data center network latency measurement and analysis," in *Proceedings of the ACM SIGCOMM Conference*, Aug. 2015.

[14] A. S. Khatouni, F. Soro, and D. Giordano, "A machine learning application for latency prediction in operational 4G networks," in *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 71–74, 2019.

[15] B. Liu, D. Niu, Z. Li, and H. V. Zhao, "Network latency prediction for personal devices: Distance-feature decomposition from 3D sampling," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, Apr. 2015.

[16] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu, and N. Li, "In-band network telemetry: A survey," *Computer Networks*, vol. 186, Feb. 2021.

[17] R. Sommer and A. Feldmann, "NetFlow: Information loss or win?," in *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, 2002.

[18] J. Quittek, T. Zseby, B. Claise, and S. Zander, "Requirements for IP Flow Information Export (IPFIX)," RFC 3917, IETF, Oct. 2004.

[19] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[20] C. Yu, C. Lumezanu, A. Sharma, Q. Xu, G. Jiang, and H. V. Madhyastha, "Software-defined latency monitoring in data center networks," in *Passive and Active Measurement (PAM)*, pp. 360–372, 2015.

[21] C. Kim, A. Sivaraman, N. P. K. Katta, A. Bas, A. A. Dixit, and L. J. Wobker, "In-band network telemetry via programmable dataplanes," 2015.

[22] G. Fioccola, A. Capello, M. Cociglio, L. Castaldelli, M. Chen, L. Zheng, G. Mirsky, and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring," RFC 8321, IETF, Jan. 2018.

[23] C. Krähenbühl, M. Wyss, D. Basin, V. Lenders, A. Perrig, and M. Strohmeier, "FABRID: Flexible attestation-based routing for inter-domain networks," in *Proceedings of the USENIX Security Symposium*, Aug. 2023.

[24] J. Kwon, J. A. García-Pardo, M. Legner, F. Wirz, M. Frei, D. Hausheer, and A. Perrig, "SCIONLab: A next-generation Internet testbed," in *Proceedings of the IEEE Conference on Network Protocols (ICNP)*, 2020.

[25] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Communications of the ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[26] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, 2008.

[27] B. Lantz, N. Handigol, B. Heller, and V. Jeyakumar, "Introduction to Mininet." https://github.com/mininet/mininet/wiki/Introduction-to-Mininet, 2021.

[28] D. Scholz, B. Jaeger, L. Schwaighofer, D. Raumer, F. Geyer, and G. Carle, "Towards a deeper understanding of TCP BBR congestion control," in *Proceedings of the IFIP Networking Conference*, 2018.

[29] Center for Applied Internet Data Analysis (CAIDA), "AS relationships with geographic annotations," 2021.

[30] J. L. Sobrinho, F. Le, and L. Vanbever, "SimBGP." https://github.com/network-aggregation/dragon_simulator, June 2014.

[31] nsnam, "ns-3." https://www.nsnam.org/, 2021.

[32] D. Thaler, "Planning for Protocol Adoption and Subsequent Transitions," RFC 8170, IETF, May 2017.