

# SPON: Enabling Resilient Inter-Ledgers Payments with an Intrusion-Tolerant Overlay

Lucian Trestioreanu  
*Interdisciplinary Centre for Security, Reliability and Trust*  
*University of Luxembourg*  
Luxembourg, Luxembourg  
lucian.trestioreanu@uni.lu

Cristina Nita-Rotaru  
*Khoury College of Computer Sciences*  
*Northeastern University*  
Boston, USA  
c.nitarotaru@northeastern.edu

Aanchal Malhotra  
*Xpring*  
*Northeastern University*  
Boston, USA  
amalhotra@ripple.com

Radu State  
*Interdisciplinary Centre for Security, Reliability and Trust*  
*University of Luxembourg*  
Luxembourg, Luxembourg  
radu.state@uni.lu

**Abstract**—Payment systems are a critical component of everyday life in our society. While in many situations payments are still slow, opaque, siloed, expensive or even fail, users expect them to be fast, transparent, cheap, reliable and global. Recent technologies such as distributed ledgers create opportunities for near-real-time, cheaper and more transparent payments. However, in order to achieve a global payment system, payments should be possible not only within one ledger, but also across different ledgers and geographies.

In this paper we propose Secure Payments with Overlay Networks (SPON), a service that enables global payments across multiple ledgers by combining the transaction exchange provided by the Interledger protocol with an intrusion-tolerant overlay of relay nodes to achieve (1) improved payment latency, (2) fault-tolerance to benign failures such as node failures and network partitions, and (3) resilience to BGP hijacking attacks. We discuss the design goals and present an implementation based on the Interledger protocol and Spines overlay network. We analyze the resilience of SPON and demonstrate through experimental evaluation that it is able to improve payment latency, recover from path outages, withstand network partition attacks, and disseminate payments fairly across multiple ledgers. We also show how SPON can be deployed to make the communication between different ledgers resilient to BGP hijacking attacks.

**Index Terms**—Performance, Interledger, Redundancy, Spines Overlay, Networks, ILP, blockchain

## I. INTRODUCTION

Recent technologies such as distributed ledgers (DLT) create opportunities for near-real-time, cheaper, global, and more transparent payments. Examples include Hyperledger Fabric [1], R3 Corda [2], Quorum [3], Stellar [4], Overledger [5], OpenChain [6], or private ETH configurations. Central banks are experimenting with DLT: Project Stella between EU and Japan, Jasper and Udin between Canada and Singapore, Project Khokha in South Africa, Emerald at Royal Bank of Scotland, UPI in India, and experimentations at the Central Bank of Brasil are just a few examples.

Recent developments in protocols allow payments to be initiated on one ledger and cross *multiple ledgers* until reaching the final payee, creating a unique opportunity for open and global payment systems. One such solution proposed to perform payments across different ledgers is the Interledger protocol (ILP) [7]. ILP is an application-layer solution and thus, it is not designed to address network level issues such as

optimizing for network latency, or resilience to network level attacks, degradations and failures. When deployed over the Internet, ILP can suffer from service degradations like lossy links, network failures and routing mis-directions of benign or malicious nature. Payment systems are critical systems and thus it is desirable they have similar levels of resiliency and security encountered in cyber-physical systems or SCADA [8] networks.

One approach to address the performance, resilience, and security issues is to use an overlay of relay nodes. These relay nodes are not part of the distributed ledger's nodes and their only goal is to relay communication between ledgers. Such an overlay of relays can leverage redundancy in the IP network and deploy customized protocols to provide desired security, latency performance, and resilience to failure and attacks.

Previous work used relays to solve some of these individual problems. For example SABRE [9] was proposed to address BGP routing attacks against Bitcoin (BTC); SABRE relies on the BGP path selection to ensure through the placement of a few nodes (<10) that most BTC nodes will not be partitioned by a BGP hijacking approach. This is achieved by relaying all the traffic through this very small set of relays that must be equipped with sophisticated hardware to sustain the high load of the BTC network. Changes in BGP peering relationships and costs will impact the correct functioning of SABRE. SABRE also relies on the fact that many BTC clients are within a very small number of ASes, and as such, scaling it for inter-ledger communication in order to cover clients spread across many different locations may not be a straightforward task. SABRE does not employ custom protocols to improve performance. Finally, SABRE requires that relay nodes do not get compromised and follow the protocol correctly. Example solutions focused on performance are Falcon [10] and Fibre [11] which both use relays for fast dissemination of BTC blocks, and BloXroute [12] which also uses relays for fast dissemination of blocks for several ledgers. All of them focus on blocks and not payments, are vulnerable to routing attacks and, as SABRE, assume that the relay nodes are not compromised and follow the protocol correctly.

In this paper we show how a global payment system enabling payments between different ledgers can be designed

and deployed over the public Internet using ILP and Spines [13] intrusion-tolerant overlay network. ILP facilitates the interoperability of any payment systems across different ledgers, while Spines serves as secure and trusted transport backbone for ILP communication. We assume that clients conducting payments within the same ledger are handled by internal ledger-specific protocols (e.g. BTC), and *we focus on inter-ledgers communication*. Our Secure Payments with Overlay Networks (SPON) system provides (1) improved payment latency between ledgers, (2) fault-tolerance to benign failures such as node failures and network partitions, and (3) resilience to BGP attacks. While intra-ledger protocols typically consider that any ledger node can be compromised (e.g. BTC nodes), previous work using relays to connect ledgers did not assume that relay nodes between ledgers can also be compromised and not forward payments or that the relay network itself can be subject to BGP routing attacks.

We implemented SPON and investigated how well it achieves its goals. *We consider 3 network topologies*: the first is a synthetic topology allowing to investigate different capabilities of SPON; the second is based on a real-life deployment [13] with nodes spanning East Asia, North America and Europe and allows to evaluate SPON’s performance in a realistic scenario, and the third was used in [14] to show the impact of eclipse attacks conducted by partitioning the network using BGP hijacking and we use it to show how SPON can be deployed to address such attacks.

We summarize our findings as following:

- We showed that SPON improves the payments latency over a *baseline* system not using the overlay. Benefits become higher as network loss increases, because the customized overlay protocols recover the lost packets from nodes closer to the recipient instead of recovering it from the sender.
- We showed that even under extreme scenarios such as a network meltdown SPON was able to continue forwarding payments by rerouting around the failures, while the baseline system could not complete the payments.
- We use the network topology in [14] to show the impact of eclipse attacks conducted by partitioning the network using BGP hijacking, as a demonstrative example on how SPON should be deployed to address such attacks.

The structure of the paper is as follows. Section II discusses challenges for global payment systems and how to overcome them by using overlays. Section III presents our SPON design and implementation, Section IV presents our experimental results. We place our project in the context of related work in V and finally, we conclude in Section VI.

## II. BACKGROUND

### A. Payments Across Different Distributed Ledgers with ILP

One interoperable solution proposed to support payments across different ledgers is the ILP protocol. We consider version 4 of ILP<sup>1</sup>, or ILPv4. Its main usage consists in multi

ledger payments, enabled by a set of connectors. To stream payments, the ILP stack provides STREAM, an additional transport protocol which breaks large payments in packets of smaller value.

The ILP ecosystem comprises of multiple software components. *Ledgers* keep records of users accounts and balances, either in fiat or crypto-currencies. *Connectors* are the transaction intermediaries and hold multiple wallets on different ledgers, such that they can perform currency exchange, and forward packets on behalf of their customers, while receiving a fee. Finally, *Applications* run by end-users to perform transactions; examples include *Moneyd*, or *Switch* by Kava Labs.

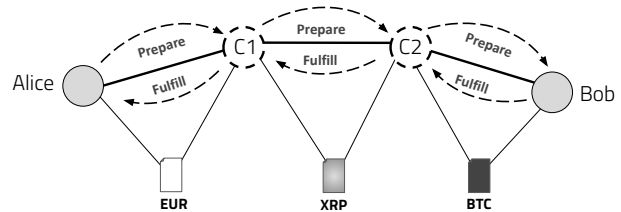


Fig. 1: Payment with ILP. C1 and C2 are ILP connector nodes.

Figure 1 shows how ILP facilitates payments. Consider customers *Alice* and *Bob*, where *Alice* has an account in Euro and wants to pay *Bob*, who has an account in BTC. Connector *C1* has an account in Euro, and an account in XRP, while Connector *C2* has an account in XRP and an account in BTC. C1 and C2 are peered together, i.e. they negotiated also a business relationship. ILP allows Alice to create a payment request in Bob’s favor, which will travel from her to C1, C2 and then to Bob. Upon receiving the payment, Bob will send back on the same path a receipt, which will finally reach Alice. The receipt assures all parties that the payment was successful and they settle their balances. As it travels between connectors, the value changed wallets and currencies.

### B. Limitations of ILP Payment Systems over the Internet

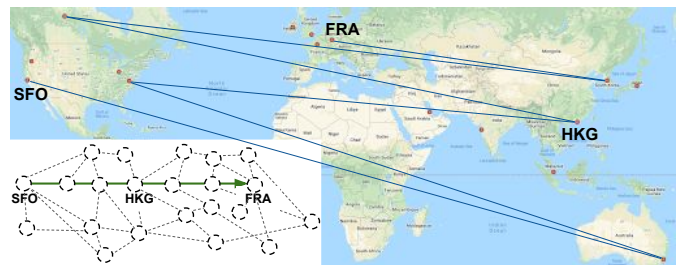


Fig. 2: Example ILP payment routing (lower left thumbnail) and actual geographical location of corresponding ILP nodes.

To facilitate the discussion about some of the limitations of current payment systems designs we present an example in Figure 2. The lower left thumbnail shows a possible example of an ILP network, where the nodes are ILP connectors. As ILP nodes may freely form links on the ILP network,

<sup>1</sup><https://interledger.org/rfcs/0027-interledger-protocol-4/>

according to reasons like regulatory, legal, business and trust relationships, the network is not constructed based on latency or attack resilience criteria. So, according to current ILP payment routing algorithm, a payment from San Francisco (SFO) to Frankfurt (FRA) could be routed along the green path also including Hong Kong (HKG). The physical locations of these ILP nodes along the payment path highlighted in green could be spread all around the world, resulting in high end-to-end latency and increased vulnerability of the payment system to lossy data paths, faults and attacks.

In this paper we focus on network level limitations of ILP payment systems. We identify three such limitations: (1) resilience to lossy paths, (2) resilience to network faults and partitions, (3) resilience to DoS such as route hijacking.

*Lossy paths* can be problematic especially in case of streaming payments, in which one single payment can be spawned over multiple smaller payments. This is encountered in pay-as-you go for torrent like distribution services [15], which can not afford packet losses even if the per packet level payment amount is tiny. Many underbanked communities [16] experience the downsides of digital, financial divides and even in developed economies some rural communities have to face mediocre Internet connectivity.

*Path failures and network partitions.* Network resilience is an important factor to consider since network enabled systems can be partitioned by intentional actions (censorship) or non-intentional (faults) accidents. The consequences for both are the same: outage, delays and degraded performance which impact the availability of the service. Payment systems should be capable to rapidly detect failures and react accordingly.

*BGP hijacking attacks.* BGP routing attacks against ILP could have a serious impact such as: partition the payment network and create a situation similar to a DoS, which can result in revenue loss for ILP nodes and their customers (open attack), delay all/chosen packets, while attacker's packets would be forwarded at normal rate (covert attack), hairpin drop packets from/to a certain ILP node/endpoint (covert attack), or at will, attacker can be the only one able to send/receive ILP transactions in/from both partitions. The attacker can also divert, store, map and analyse the traffic: get geo-location information of ILP providers/customers, gather/infer information about payments volumes per ILP node (average value carried by an ILPv4 packet at the attack moment is  $x$  XRP).

### III. SPON DESIGN AND IMPLEMENTATION

In this section we describe SPON, our proposal for resilient global payment systems over Internet. We first describe the design goals for our system, then describe the attacker model, and give a description of the design and implementation.

#### A. Design Goals and High-level Approach

Our main goal is to design a global payment system that supports payments across different ledgers *while achieving*: improved performance (latency), improved service availability (fault-tolerance), and security guarantees, including resilience to routing attacks. We assume clients conducting payments

within the same ledger are handled by ledger-specific protocols. While these internal protocols can also benefit from additional improvements, *our focus is on connecting different ledgers and not on services within a ledger.* We use ILP to facilitate the exchanges across different currencies and ledgers. However, ILP is not meant to optimize network communication and address fault-tolerance to network failures or BGP attacks. With our goals in mind, we would like our service connecting multiple ledgers to have the following properties:

- G1** Improved payment latency: Our design should leverage the redundancy in the underlying IP network to take advantage of links offering better connectivity, by using customized routing protocols.
- G2** Resilience to lossy paths: Our design should be resilient to lossy communication links across ledgers and as such improve the client network's resilience to lossy links.
- G3** Resilience to path failure and node crashes: The design should increase payment service availability by increasing data flow availability through providing a system resilient to network path failures and relay node crashes.
- G4** Resilience to BGP routing attacks: Our design, also deployment dependent, should provide resilience to routing attacks like *Coremelt* and *Crossfire* [17], [18].

**Approach.** These goals can be achieved by changing an existing payment-exchange protocol like ILP to add the desired performance, fault-tolerance, and attack resilience. However, we argue that a separation between the payment-exchange and the communication functionalities provides more flexibility in ILP node placement and modularized development. For example, the ledger pre-post processing functionality is better placed closer to the ledger; also, because they manipulate user value and data, the placement of ILP nodes in different geographical areas may involve different legal restrictions, licensing, regulations. A compromised ILP node is more dangerous than a compromised overlay node performing a simple forwarding because the forwarding nodes do not need visibility into the payments to perform network-level forwarding. Thus, our approach is to separate ledger processing from the forwarding functionality, to maximize performance and resilience to attacks, while accommodating legal restrictions. The data forwarding layer can be an overlay of *relay nodes* that implement customized routing algorithms for better latency, routing around failures and with BGP attack resilience. The ILP payment exchange connectors use the overlay of relays to communicate with each other.

Figure 3 shows how communication flows between ILP nodes Alice and Bob, through ILP and the overlay of relay nodes (Alice and Bob are not end-users but full ILP nodes): Each ILP node is connected to at least one overlay relay node. Each overlay relay node is connected to multiple Internet Service Providers (ISP) / Internet Exchange Points (IXP) / Autonomous Systems (AS). At ILP level, a payment originated from Alice for Bob, is routed through the "ILP connector" in the middle. However at data packet level, the 2 hops (Alice <-> Connector and Connector <-> Bob, are routed through

redundant paths on the overlay network (thick arrows on the middle layer of Figure 3). Further, each overlay link benefits from disjoint, redundant paths at Internet level below.

**Need for intrusion-tolerant overlays.** Overlay networks can improve latency because they can reduce re-transmissions [19], [20] and can provide resilience to benign faults by routing around them. However, the introduction of the overlay of relay nodes in the system design changes the trust model. First, the overlay itself is susceptible to compromises since a software node is easier to compromise than a hardware router. Compromised overlay nodes can significantly impact the system performance as a whole, or target specific connectors or ledgers and discriminate against some clients conducting payments. Second, the nature of the overlay requires different payment streams to share the same logical structure which can allow some clients to create denial of service against competitor clients conducting payments through the same link(s) on the relay network. Such overlays need to be centrally managed to prevent topology related attack. We set the following goals for our overlay of relays:

- O1** Resilience to attacks from compromised forwarding relays: We want to prevent compromised relay nodes from being able to divert or stop traffic.
- O2** Resilience to denial-of-service from malicious clients: In the presence of the overlay, payment flows from different competitor clients can potentially compete to each other at networking level to the point where one can generate a targeted denial of service for the other by saturating the link(s). We would like all payment flows to be treated fairly by the relay nodes, i.e. all payment streams receive the same share of available network bandwidth.

### B. Threat Model

We assume that the overlay of relay nodes is centrally managed and communication between relay nodes is authenticated using Public Key Infrastructure (PKI), where the system administrator and each overlay node has a public/private key pair and knows all the other public keys. The overlay topology is known by all of the overlay nodes, and changes can be made only by the system administrator.

We also assume that overlay relay nodes can be compromised. A compromised node can exhibit Byzantine behavior such as arbitrary dropping, delaying, or incorrect forwarding of packets. We assume that overlay nodes have sufficient computational resources to keep up with processing incoming messages, but bounded buffers for message storing.

We do not assume a specific bound on the number of compromised relays in the overlay network. Instead we assume that the adversary cannot partition the sender from the receiver, i.e. there is a path from the sender to receiver where all relays are not controlled by the adversary.

We assume attackers have large amounts of network bandwidth, memory and computation, such as those required by large-scale DDoS attacks as those in [17], [18].

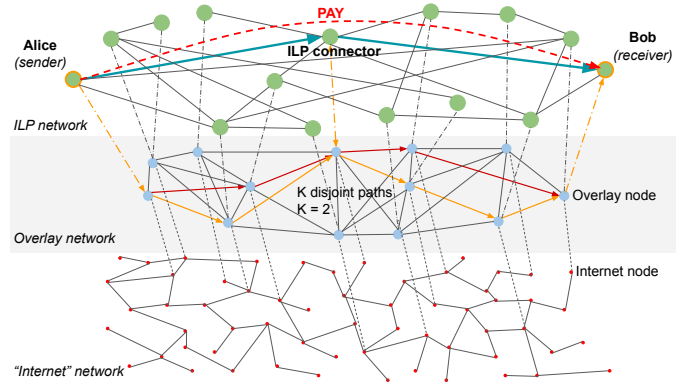


Fig. 3: Communication mapping for Ledgers, overlay, Internet.

### C. SPON Design and Implementation

We implemented SPON using ILP and the Spines overlay. Below, we first give a description of aspects of ILP and Spines relevant to our design, then describe our system, SPON.

The ILP environment consists of a stack of protocols:

- *Bilateral Transfer Protocol* (BTP), responsible of establishing a link between two peers.
- ILP itself, ensuring the value transfer across ledgers. The ILP packet offers a *data* field of size 32k, where different information and sub-protocols can be encapsulated.
- *Streaming Transport for the Realtime Exchange of Assets and Messages* protocol (STREAM), implementing the concept of streaming value (money) and data over ILP (encapsulated in ILP packets). This concept offers a series of advantages over sending a transaction in full.
- *Simple Payment Setup Protocol* (SPSP) ensuring the exchange of credentials required to establish a STREAM payment, which for specific reasons works over HTTP.

Spines is an open source overlay network [20], [21] that provides availability, resiliency, and timed-delivery, achieved by making use of multi-homing at multiple ISPs and deploying the nodes in strategically located datacenters (connectivity). The nodes are centrally managed and resilient overlay routing such as multiple disjoint paths and flooding [13]-p6 are used to ensure resilience to forwarding attacks. Buffer management like round robin is used to ensure that each node evenly processes packets per sender in case of priority sending, or per flow (sender-receiver pairs) in case of reliable sending.

We show the architecture of SPON in Figure 4. There are 3 network layers: the base internet layer, the Spines overlay, and the ILP network, each featuring their own addressing schemes and protocols. Each ILP node connects to a Spines node using the stack illustrated in Figure 4. The connector applications connect through a tunnel, agnostic of the overlay below. An *adapter* application makes the connection to the *spines\_socket* exposed by the Spines node, and sends it the different parameters to use in order to forward data. We use the *Priority Messaging (PRI)* and *Reliable Messaging (REL)* communication services, shown and explained in Table I.

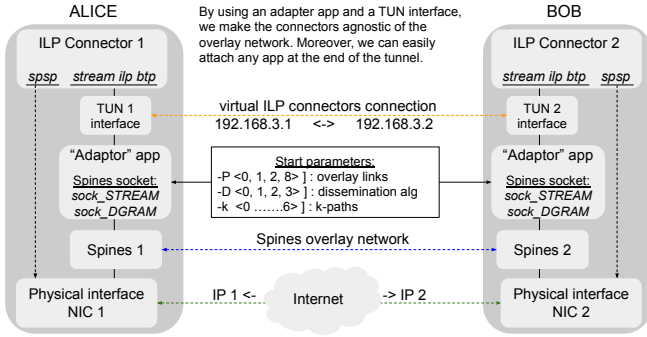


Fig. 4: SPON Architecture.

One advantage of SPON is that the service can be selected per ILP packet, because Spines provides its reliable or priority services on a per packet basis. Our design exposes this functionality to ILP payments and other ILP tools such as ILP-ping. As such, for example, the risk of *fulfillment failure* specific to ILP, could now be alleviated by prioritizing the *fulfilling* over the *prepare* packets<sup>2</sup>. As needed, any ILP related flow can be *prioritized* or sent *reliably*, for example routing updates or SPSP data could use the reliable protocol.

Because the connectors are agnostic of the overlay below, our design also allows for a *partial deployment*, where some connectors choose to join the network and others do not. This involves the existence of some *bridge* connectors, having connections both outside and inside SPON.

TABLE I: SPON services (via Spines).

| Service               | Details                                                                                                                                                               |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PRIORITY (PRI)</b> | Source-based routing with timeliness guarantees, i.e. packets are sent based on their priority, each node forwards packets fairly across all sources.                 |
| <b>RELIABLE (REL)</b> | Source-based routing with reliability guarantees, i.e. packets are sent with end-to-end reliably, each node forwards packets fairly across all sender-receiver pairs. |

#### IV. EXPERIMENTAL RESULTS

In this section we describe the evaluation of SPON. We seek to answer the following questions:

- Q1** What are the latency improvements of SPON when compared with an approach that does not use relays?
- Q2** How does SPON react to more severe network events such as network meltdowns?
- Q3** How does SPON handle denial of service attacks where some clients try to overload the links with payments?
- Q4** How does SPON react to severe network events such as route misdirections and BGP hijacking attacks?

##### A. Methodology

We conduct our experiments using Mininet to better control the network topology, links and their properties. We used the "reference" ILP connector<sup>3</sup> and a private XRP ledger.

<sup>2</sup><https://interledger.org/rfcs/0018-connector-risk-mitigations/>

<sup>3</sup><https://github.com/interledgergerjs/ilp-connector>

*Topologies.* We used 3 topologies for our evaluations, and a fourth to demonstrate BGP resilience. The first, referred as *Chain Topology* (Figure 5) is a demonstrative topology allowing to investigate different path capabilities of our overlay. The second, referred as *Global Topology* (Figure 6) is a real-life topology spanning the Internet and obtained from [22] which allows to demonstrate the performance and resilience of SPON in a more realistic scenario. Link latencies were obtained from specialized websites<sup>4</sup>. Third setting, shown in Figure 12 helps answer Q3, while Q4 is discussed using Figure 14.

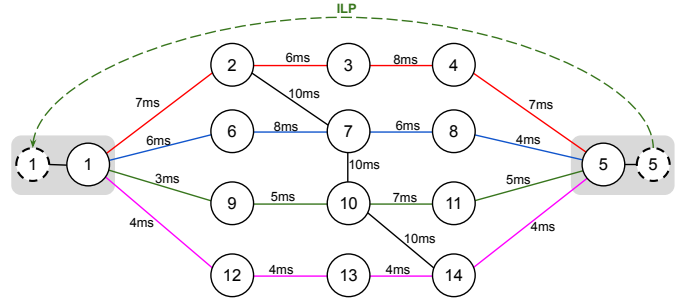


Fig. 5: Chain Topology.

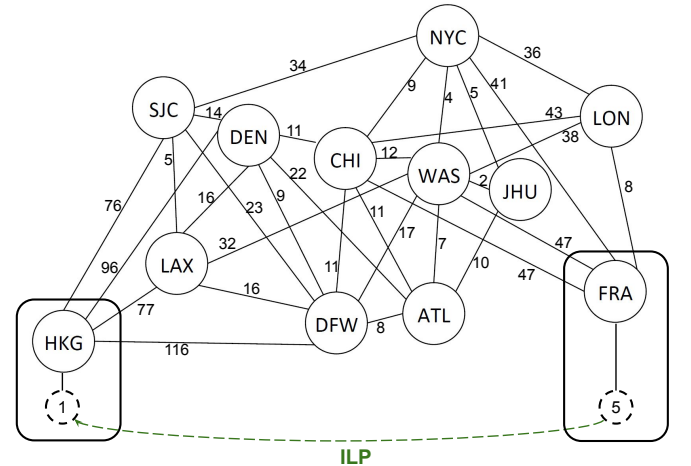


Fig. 6: Global Topology.

*Systems.* We compare the following configurations:

- Baseline: payments are sent via ILP nodes without SPON.
- Priority (PRI): payments use SPON configured with source-based routing and timeliness delivery [13].
- Reliable (REL): payments use SPON configured with source-based routing and reliable delivery [13].

For both Priority and Reliable settings, we evaluated Flooding (FLD) and k-path as communication mechanisms. **Q1** and **Q2** are answered by comparing the *Baseline* with SPON's behavior in *PRI* and *REL* mode.

<sup>4</sup><https://ipnetwork.windstream.net/>, <https://wondernetwork.com/pings>

*Metrics.* We use *Round Trip Time on ILP* ( $RTT_{ILP}$ ) reported by the ILP Ping tool<sup>5</sup> to evaluate the communication between ledgers via SPON. For larger payments which are broken into a number of ILP packets and sent via *STREAM*, we use *Payment Latency* as the total time to complete a payment.

## B. Performance

1) *Chain topology:* As illustrated in Figure 5, we use two ILP nodes (5 and 1) acting as sender and receiver, to send 100 ILP ping packets at a rate of 1 packet/s, using the ILP-PING tool. The baseline ( $RTT_{ILP}$ ) is 32ms and equivalates the two *connectors* paired directly on the fastest path from the figure.

*ILP RTT.* To evaluate latency under loss, we introduce variable loss of 2, 5, 10% on link S12-S13, chosen because it's on the fastest topology path, so it has high chances to have a visible impact on results, illustrated in Figures 7b, 7c, 7d. Solid grey bars represent baseline averages, grey striped bars represent *Priority* messaging with flooding (FLD), 1 or 2 paths [13]-p6, and dark grey bars represent *Reliable* messaging with FLD, 1 or 2 paths. While not shown experimentally, we appreciate that introducing loss on slower paths (9-10, 6-7, 2-3) would advantage SPON by enabling it to use the fastest path at full capability. We isolate Spines' processing overhead by setting loss to 0; as shown in Figure 7a, SPON does fare a little bit worse than the baseline (5% or 6s in our case). This overhead however is small and does not prevent SPON from performing better than the baseline in realistic situations with loss: at 2% loss, Figure 7b shows that SPON already offers an advantage of 10% latency over the baseline when working in FLD mode. As loss increases, SPON's advantage increases, and at 5% loss the gain over same baseline is 33%, as depicted in Figure 7c. The error bars also point that the service is more stable under loss, if using SPON.

*Payment latency.* We evaluate ILP payment latency under similar scenarios with network loss. On the topology in Figure 5 we sent 20 ILP *STREAM* payments. The amount per ILP payment was 100000 drops (1 drop = 0.000001 XRP)<sup>6</sup>; each *STREAM* packet was 100 drops. Thus, for each payment we sent 1000 ILP *STREAM* micro-transactions. We used *Priority* and *Reliable* messaging with FLD ( $k=0$ ), 1 and 2 paths ( $k=1,2$ ). The loss was set again on link S12-S13. In Figures 8a,8b,8c,8d we compare the time taken to complete the transactions over SPON, with the baseline: under ideal conditions (loss 0), payment latency over SPON is a little bit larger than over the baseline (under 5%, or 2s in this case), while at 2% loss, SPON already offers a gain of 10% (5s) in FLD mode. At 5% loss, all SPON modes show 15-33% gains.

2) *Global topology:* To demonstrate the behavior in a more realistic scenario, we repeat the experiments above on the Global topology; inspired from [22], it offers increased link redundancy while using well-chosen real-world, global locations spanning US, EU and Asia. Each circle represents an overlay node deployed on our Mininet testbed. As baseline, we

sent *STREAM* ILP payments between two *connectors* paired directly over a single link with delay 148ms - equivalent to the fastest path from Figure 6. On the global topology, the *connectors* were attached to the overlay nodes FRA and HKG, and sent a total of 16 ILP payments directly through the *STREAM* protocol (no SPSP). The total transaction amount was 100000 drops per ILP payment, and each *STREAM* packet was 500 drops (200 *STREAM* micro-transactions). The *loss* was introduced between HKG and SJC because the link belongs to multiple low latency (possible) paths, and as such, with chances to impact multiple possible flows.

The results in Figure 9a,9b,9c,9d show that in ideal conditions, except for sending on 1 path, SPON adds only 1.5% to the total payment duration, compared to baseline; at 2% loss, SPON offers a gain of 5%; while at 5%, the gain is 16%.

In summary, in all scenarios we experimented with, the additional processing introduced by SPON and identified at loss 0 was small, and the payment system offered better performance under a link loss of 2, 5, 10%.

## C. Resilience to Network Melting.

1) *Chain topology:* We want to see how an ILP payment sent over SPON behaves when all but one path fail. We set all links to loss 0. Because the baseline would obviously fail in this scenario, we can only assess how SPON's performance would compare with a functional baseline. So concerning baseline, we send a payment between 2 *connectors* paired over a link of 20ms latency - equivalent to the remaining path 1-9-10-11-5 from Figure 5, if all other paths fail.

We send an ILP payment of 100000 drops, and packet size 10 drops. Thus for each payment we sent 10000 ILP micro-transactions, for a total *STREAM* duration of 480s. While the *STREAM* is sent, we take down the communication of the overlay nodes 2, 7, 14 using *IPtables* on the respective machines, at a 40s interval, in a 5-count cycle. This procedure completely melts and brings back every 40s, all the possible paths but the green one (nodes 1-9-10-11-5) from Figure 5.

In Figures 10a, 10b, 10c we plot individual ILP packet latencies. We observe that, if one of the currently active transmission paths is the actual path to remain unaffected by the network melt, then the system can offer optimal protection against the melt starting even from 2-paths; on 1-path, the minimal drawback comes due to rerouting time to a better path after the network becomes available again. Even when all but one path vanish, SPON service continues reliably, with no packets lost during the experiment.

Concerning the total duration of payments sent over the baseline versus SPON, even when the latter was subjected to the severe path flipping above, it still performed slightly better than the baseline (3%), as shown in Figure 10d. This is because the baseline is able to send only on the 20ms link, while at times, SPON can also use the fastest path of 16ms.

2) *Global topology.* Through our two *connectors* attached to the Spines nodes FRA and HKG, we sent a payment of 80000 drops, and packet size 50 drops (1600 ILP micro-payments), during a total time of 500s. While the *STREAM* is sent, we

<sup>5</sup><https://github.com/martinlowinski/ilp-ping>

<sup>6</sup><https://xrpl.org/xrp.html>, accessed August 2021

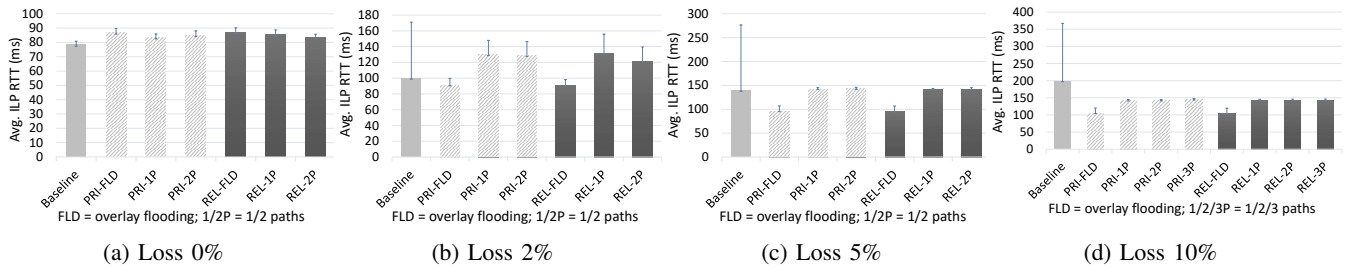


Fig. 7: Average ILP ping RTT on the Chain topology in a network loss scenario, Priority (PRI) or Reliable (REL) messaging.

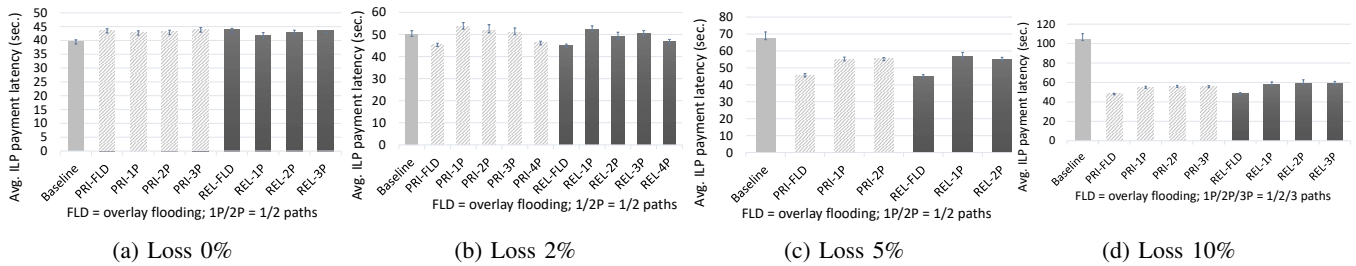


Fig. 8: Payment latency on the Chain topology in a network loss scenario, Priority (PRI) or Reliable (REL) messaging.

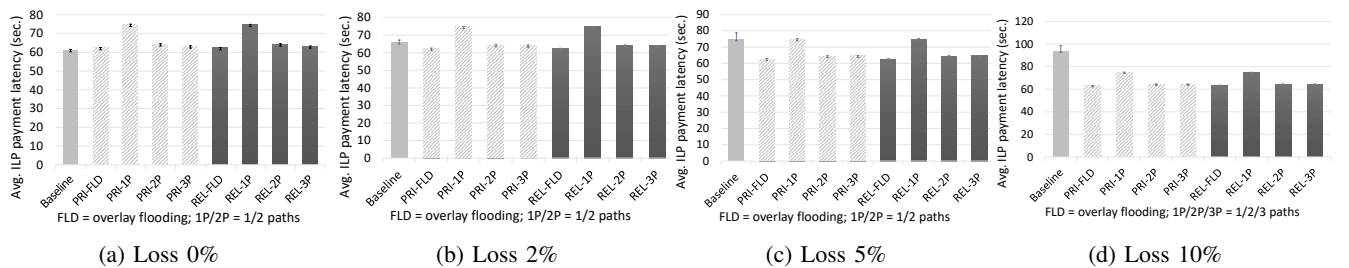


Fig. 9: Payment latency on the Global topology in a network loss scenario, Priority (PRI) or Reliable (REL) messaging.

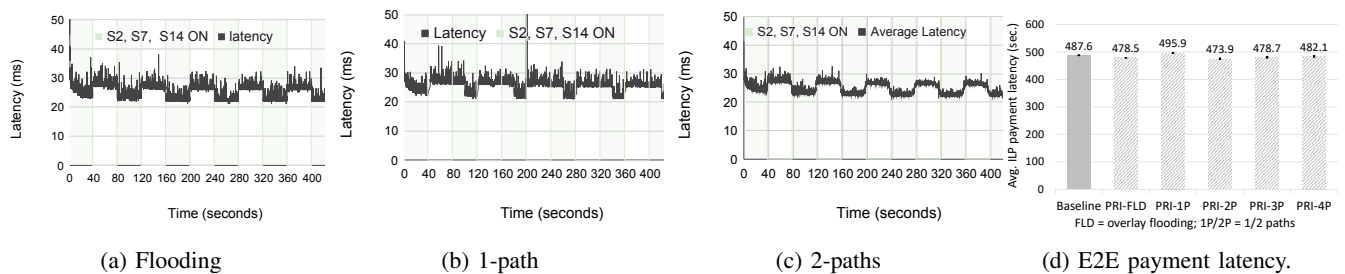


Fig. 10: Payment latency on the Chain topology in a network meltdown scenario, Priority messaging (PRI).

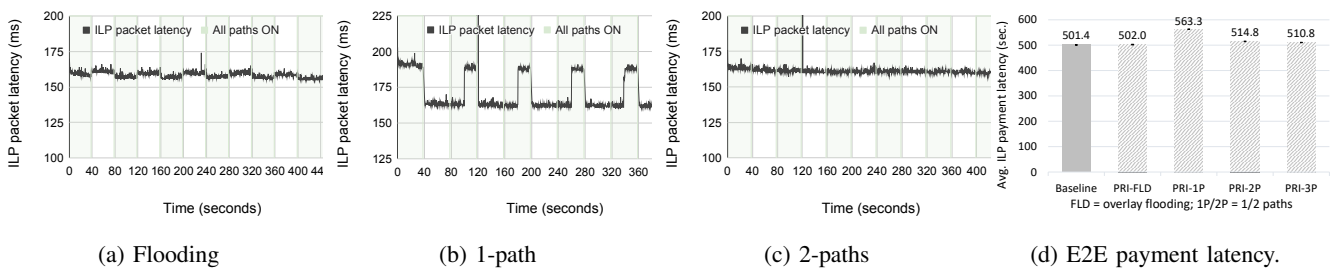


Fig. 11: Payment latency on the Global topology in a network meltdown scenario, Priority messaging (PRI).

cut the communication of nodes SJC, NYC, LON, WAS, JHU, DFW, ATL using *Iptables* on the respective machines, at a 40s interval, in a 5-count cycle. This procedure completely melts and brings back every 40s, all the possible paths but FRA-CHI-DEN-LAX-HKG from Figure 6. The baseline is two ILP *connectors* paired over a single link with delay 151ms - equivalent to the remaining path (FRA-CHI-DEN-LAX-HKG) from Figure 6, after all other paths go down. To compare the time taken to complete the transactions over the overlay versus baseline, we repeat the experiment 5 times, average the results for each case, and finally represent them in Figure 11d. The individual ILP packet latencies are obtained after unique, single runs of the experiment with Priority messaging over 1, 2, 3 paths or FLD (Figures 11a, 11b, 11c). Results for 3-path were similar to flooding and are not illustrated. We notice that in case of a complete network melt up to 1 path, SPON's service continues, while the baseline completely fails. The E2E payment latency over SPON, illustrated in Figure 11d, is similar to the baseline (502 vs 501s).

#### D. Resilience to Denial of Service from Malicious Clients

With the aim to assess how an ILP flow sent over the overlay at maximum link capacity behaves in the presence of a second malicious flow trying to take over the channel bandwidth (BW), we attach four ILP *Connectors* (1, 2, 5 and 6) to the overlay nodes in Figure 12), and then we create two ILP flows. Connector 5 is paired with, and sends an "honest" flow to Connector 2 while Connector 6 is paired with, and sends a "malicious" flow to Connector 2. To each connector we can attach progressively, at 1s interval, up to 100 clients each sending over 8 streams. We are thus able to generate for each flow a maximum traffic of 15Mbps, and as such, on our topology, we set maximum link capacity to 15Mbps. For this experiment we set all links to loss 0 and as metric we used flow size in Mbps. The experiment is carried as follows. While the first, legitimate flow (C5 to C1) is sent at maximum capacity, we progressively increase the malicious, contending flow, trying to fill BW up to maximum channel capacity. Both flows were sent with Priority messaging over 1-path. As illustrated in Figure 13, the legitimate flow decreases progressively, but only up to its fair share of 1/2 channel capacity. Although the malicious flow tried to increase its flow and send at its maximum capacity of 15Mbps, it was not able to do so beyond its fair share of BW and hence, it could not take over the channel or stop the legitimate flow. While for the particular case of ILP we experimented with only two sources, experiments with multiple sources can be found in [13].

#### E. BGP Hijacking Attacks and Benign Route Misdirections.

BGP routing attacks have been widely explored in literature. Hijacking attacks followed by double spending on Ethereum have been discussed by [14] for private, consortium or public deployments. An experimental topology for public networks has been illustrated in [14], and we use it as a working example to show how on the same topology, SPON can defend against

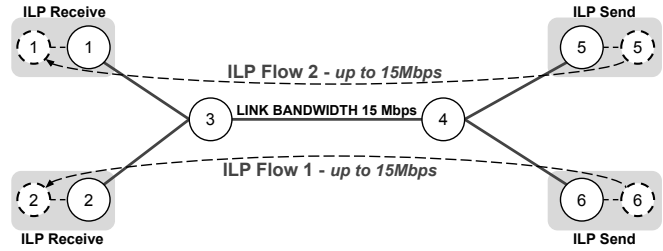


Fig. 12: Network topology for the flow fairness.

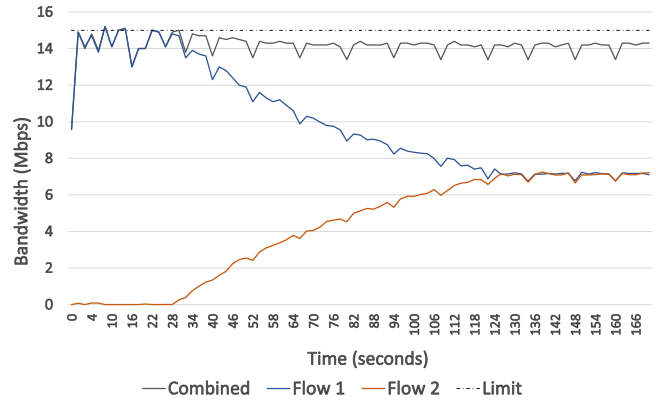


Fig. 13: Legitimate and malicious flows contending for BW.

AS-level BGP routing attacks, through a careful design of the network. As represented in Figure 14, by deploying the SPON nodes in IXPs and thus benefiting from access to say 2 or 3 ASes of interest, SPON nodes are able to ensure connectivity in spite of BGP attacks. For example, while the route between AS2 and AS4 is controlled by the adversary AS3 who partitioned AS2 from AS4, AS4 can still be reached from AS2 through SPON nodes placed appropriately in IXPs, with redundant connections to multiple ASes, and thus still being able to relay traffic for their ILP clients located in AS2 and AS4, regardless of the hijacked route.

## V. RELATED WORK

Recent efforts towards advancing the state of the art include projects like *Fibre*<sup>7</sup>, *Falcon*<sup>8</sup> [10] or *bloXroute* [12], which aim to improve blockchain transaction rate by speeding up block propagation. *Falcon* has the disadvantage that a block can be validated only after receiving all required packets. *Fibre* uses Forward Error Correction to enable nodes to reconstruct data in advance even if some parts have been lost on the way [23], while *Spines* proposes *Soft Realtime Link protocol* enabling localised retransmissions to increase packet delivery ratio [24] and protects against BGP hijacking. However, all above but SPON are vulnerable to BGP failures. *bloXroute* seeks to treat all blocks (or payments) fairly but it assumes that the overlay

<sup>7</sup><https://bitcoinfibre.org/>, accessed May 2021

<sup>8</sup><https://www.falcon-net.org/about>, accessed May 2021



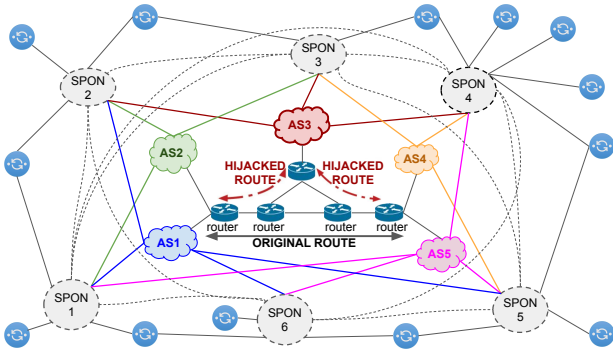


Fig. 14: BGP attack mitigation with SPON.

ILP nodes in light blue; overlay links between SPON nodes in dashed curvy lines; SPON connections to different ASes/ISPs in straight colored lines. Part of figure from [14].

nodes can not be compromised; it also sends audit control packets (trivial to implement in SPON at ILP level using STREAM), and together with Falcon, consider the *incentivization* of overlay operators (also straightforward to implement in SPON). SABRE [25] focuses on protecting BTC against BGP hijacking, and partially because of a low relay/client ratio, it uses software-hardware co-design to sustain high loads. It does not consider compromised relay nodes. Nebula [26] and Open Overlay [27] provide *security groups* and *access control lists*, but are not intrusion-tolerant.

## VI. CONCLUSION

We proposed SPON, an architecture for a global payment system that uses a reliable, intrusion-tolerant overlay network. SPON provides (1) improved payment latency, (2) fault-tolerance to benign failures such as node failures and network partitions, (3) resilience to routing attacks, while only incurring a small overhead. Our experimental results show that overlay networks are a viable solution towards making global payment systems a reality by increasing their service availability and improving latency.

## ACKNOWLEDGMENT

This work is supported by the Luxembourg National Research Fund through grant PRIDE15/10621687/SPsquared. In addition, we thankfully acknowledge the support from the RIPPLE University Blockchain Research Initiative (UBRI) for our research.

## REFERENCES

- [1] "An introduction to hyperledger," <https://www.hyperledger.org/learn/white-papers>, accessed: May 2021.
- [2] R. G. Brown, "The corda platform: An introduction," <https://www.corda.net/content/corda-platform-whitepaper.pdf>, accessed: May 2021.
- [3] A. Baliga, I. Subhod, P. Kamat, and S. Chatterjee, "Performance evaluation of the quorum blockchain platform," <https://arxiv.org/pdf/1809.03421.pdf>, accessed: May 2021.
- [4] "Stellar consensus protocol," <https://www.stellar.org/papers/stellar-consensus-protocol?locale=en>, accessed: May 2021.

- [5] "Quant overledger whitepaper," [https://www.quant.network/wp-content/uploads/2020/07/Quant\\_Overledger\\_Whitepaper-Sep-1.pdf](https://www.quant.network/wp-content/uploads/2020/07/Quant_Overledger_Whitepaper-Sep-1.pdf), accessed: May 2021.
- [6] "Open chain white paper," <https://docs.openfuture.io/OPEN-Chain-White-Paper.html>, accessed: May 2021.
- [7] S. Thomas and E. Schwartz. (2016) A protocol for interledger payments. [Online]. Available: <https://interledger.org/interledger.pdf>
- [8] G. Yadav and K. Paul, "Architecture and security of scada systems: A review," 2020.
- [9] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017, pp. 375–392.
- [10] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse, and E. G. Sirer, *Decentralization in Bitcoin and Ethereum Networks*. Springer Berlin Heidelberg, 2018.
- [11] M. Corallo, <https://bitcoinfibre.org/>, accessed: May 2021.
- [12] U. Klarman, S. Basu, A. Kuzmanovic, and E. G. Sirer, "bloXroute: A scalable trustless blockchain distribution network WHITEPAPER," in *IEEE Internet of Things Journal*, 2018.
- [13] D. Obenshain, T. Tantillo, A. Babay, J. Schultz, A. Newell, M. E. Hoque, Y. Amir, and C. Nita-Rotaru, "Practical intrusion-tolerant networks," in *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, June 2016, pp. 45–56.
- [14] P. Ekparinya, V. Gramoli, and G. Jourjon, "Impact of man-in-the-middle attacks on ethereum," in *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, 2018, pp. 11–20.
- [15] "Ilp torrent - the technical deep dive," <https://coil.com/p/sabinebertram/ILP-Torrent-The-technical-deep-dive/S2cdTMKby/>, accessed: May 2021.
- [16] T. Friedline, S. Naraharsetti, and A. Weaver, "Digital redlining: Poor rural communities' access to fintech and implications for financial inclusion," *Journal of Poverty*, vol. 24, no. 5-6, pp. 517–541, 2020. [Online]. Available: <https://doi.org/10.1080/10875549.2019.1695162>
- [17] A. Studer and A. Perrig, "The coremelt attack," in *Computer Security – ESORICS 2009*, M. Backes and P. Ning, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 37–52.
- [18] M. S. Kang, S. B. Lee, and V. D. Gligor, "The crossfire attack," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 127–141.
- [19] C. Danilov, "Performance and functionality in overlay networks," Ph.D. dissertation, The Johns Hopkins University, Baltimore, Sep. 2004. [Online]. Available: [http://www.dsn.jhu.edu/~yairamir/Claudiu\\_thesis.pdf](http://www.dsn.jhu.edu/~yairamir/Claudiu_thesis.pdf)
- [20] A. Babay, C. Danilov, J. Lane, M. Miskin-Amir, D. Obenshain, J. Schultz, J. Stanton, T. Tantillo, and Y. Amir, "Structured overlay networks for a new generation of internet services," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 1771–1779.
- [21] Y. Amir, C. Danilov, J. Schultz, D. Obenshain, T. Tantillo, and A. Babay. (2020, Mar.) Spines. [Online]. Available: <http://spines.org>
- [22] A. Babay, E. Wagner, M. Dinitz, and Y. Amir, "Timely, reliable, and cost-effective internet transport service using dissemination graphs," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 1–12.
- [23] W. Bi, H. Yang, and M. Zheng, "An accelerated method for message propagation in blockchain networks," *ArXiv*, vol. abs/1809.00455, 2018.
- [24] Y. Amir, C. Danilov, S. Goose, D. Hedqvist, and A. Terzis, "An overlay architecture for high-quality voip streams," *IEEE Transactions on Multimedia*, vol. 8, no. 6, pp. 1250–1262, Dec 2006.
- [25] M. Apostolaki, G. Marti, J. Müller, and L. Vanbever, "SABRE: protecting bitcoin against routing attacks," in *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019. [Online]. Available: <https://www.ndss-symposium.org/ndss-paper/sabre-protecting-bitcoin-against-routing-attacks/>
- [26] N. Brawn and R. Huber. (2019) Nebula. [Online]. Available: <https://github.com/slackhq/nebula>
- [27] A. Rodriguez-Natal, J. Paillisse, F. Coras, A. Lopez-Bresco, L. Jakob, M. Portoles-Comeras, P. Natarajan, V. Ermagan, D. Meyer, D. Farinacci, F. Maino, and A. Cabellos-Aparicio, "Programmable Overlays via OpenOverlayRouter," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 32–38, June 2017.