# ILP Monitor: Creating Trust in the Interledger Network

Nikhil Suri

May 7, 2019

## 1 Abstract

Interledger is a payments network that provides scalability for blockchains and interoperability between existing ledgers, whether centralized or decentralized. While the network is live and currently processes payments, Interledger is still in its infancy phases as developers and ecosystem companies continue to concretely define how its theoretical protocols can be practically implemented. A critical piece of development which the Interledger community currently lacks is visibility into what is actually being created and how it is growing and being used. This paper introduces a prototype of ILP Monitor: a network monitoring system for Interledger. A discussion of methodology and results is followed by a survey of the current state of Interledger and possible areas of further research and development of the monitoring platform.

## 2 Introduction

Bitcoin [14] was the first successful foray into online and decentralized payment networks. Since then, the novel ”blockchain” solution that Bitcoin provided has served as a foundational way to establish trust in inherently trustless sytems. While many other digital currencies offering their own unique properties and advantages have followed suit, there has also been much work in applying blockchain to other areas to establish systems of online trust among unknown peers. A notable example of such a system is Ethereum [3], which allows users to publish ”smart-contracts” in addition to peer-to-peer transactions on the blockchain. These can be used for applications ranging from governance to trading in decentralized markets.

Despite the exciting possibilities that blockchain technology brings, however, it remains at the edge of mainstream use. Some of the current problems preventing its more widespread usage have to do with the fact that it is not very user friendly and independently interacting with the technology often requires advanced programming knowledge. The biggest problem, however, is that a majority of the public, which includes investors, governments, companies, and potential users, simply does not trust the technology. Without applications to monitor and display data on these public systems, there is no visibility into the wider network. This can be very problematic, as decentralized ledgers are subject to various attacks which vary from cutting off or shutting down nodes to undermining the validity of the blockchain itself. Indeed, if it were not for public websites such as Blockchain.info, users would have no way to measure the size of or quickly address issues on blockchain networks. It seems that truly trustless systems are still vulnerable and introducing public methods of establishing trust in networks can be very helpful for growth.

The Interledger network is only a few years old and, while many developers and ecosystem companies are building products to increase its usability and user-friendliness, its inner-workings have remained fairly opaque as there are few methods for potential users and investors to critically evaluate its problems and metrics such such as size and reliability. There are many benefits to monitoring the network. First, greater visibility would make the network more tangible to new developers or companies that are considering contributing to or building applications on Interledger. Second, it provides connector operators with much better insight into how they could benefit the network. For example, if a connector operator wants to provide Bitcoin liquidity, seeing how much network traffic is in XRP will give them a clear idea of how much value they are providing. Third, seeing a working network validates Interledger within the existing community of decentralized technology. Fourth, it can help both companies and users of the network better evaluate which connectors to peer with. The public data which network monitoring provides will also allow problems to be solved in real time and the development of smarter routing algorithms.

The Interledger community is still quite small, so I was able to interact with many members such as developers and leaders of ecosystem companies while working on this project. To gain a better understanding of the network and meet those involved, I attended the Interledger Summit in San Francisco. Through posts on the Interledger forum and participation in community calls, I was able to generate discussion about and support for this project. I am the leader of the Interledger network monitoring working group, which is focused on building this sort of system. As a result of these efforts, there is considerable excitement from the community about this project and interest in continued development and support for tools to monitor the Interledger open network.

## 3 Background

The important background to understand this work and the nature of my contributions is the Interledger protocol itself, the current users of the network, and the kinds of statistics that I want to collect on the network.

### 3.1 Interledger

Interledger was first introduced through a whitepaper written by Stefan Thomas and Evan Schwartz in 2015 [18]. The current specification of the protocol, ILPv4 [6], is a vast simplification of the original protocol. It also is a more accurate model of the internet protocol stack and optimized for sending large volumes of small sized packets of value.

There are three kinds of actors on the Interledger network: senders, receivers, and connectors [5]. As their names imply, senders send money to receivers in many ”packets” of small amounts. Money is routed from a sender to a receiver through connectors. To establish routes in the network, connectors peer with each other. All payments are made using payment channels between connectors and connectors run software which maintains accounts for all of their peers. Similar to the Lightning network for Bitcoin [16] on-chain settlement is deferred. Importantly, connectors can also provide currency interoperability by acting as exchanges. This allows payments that are essentially currency agnostic to be made using the Interledger network. Running connectors in the network is meant to be a competitive business (similar to modern-day internet service providers) where connectors can make money from spreads, flat fees per packet, subscription fees for higher payment bandwidths (defined in section 3.3.3), or value added services such as APIs and analysis products.

The Interledger stack is modeled after that of the Internet in that it has different layers of protocols which all work together to transfer packets, which in the case of Interledger represent value instead of arbitrary data. The highest layer on the protocol stack is the ”application” layer. My project uses application-level protocols extensively. The Ping protocol establishes a method of sending pings between connectors on the network. The Simple Payment Setup Protocol provides a secure method of establishing a shared secret between a sender and receiver on the network (which is then used to validate packets of value that they exchange).

Currently, money is routed through the network by connectors using the shortest path between a sender and receiver. If multiple paths exist between the two endpoints, then an arbitrary tie-breaking algorithm is used. By default, the standard Interledger connector open-source software [20] sets exchange rates according to the data from the CoinMarketCap API and enables connector operators to add a spread percentage on top. However, connectors have complete control over their rates and do not have to set them in that way.

## 3.2 Users of Interledger

The main users of Interledger currently appear to be developers and researchers working on independent projects (such as this one) and ecosystem companies.

Most of the companies building on Interledger are startups banking on the hope that usage will increase from the nascent to, eventually, the mainstream. Anyone who wishes to operate on the Interledger network needs to be running a connector. These companies are likely already collecting statistics on their own connectors and have varied interest in an open network monitoring and data collection tool. Strata Labs, for example, maintains infrastructure for the Interledger network by running connectors for other companies or individuals. Open network information could be very useful to them as they make decisions regarding resource allocation. The Flare Network [17] uses Interledger for node remuneration. To finalize events, the Flare Network uses extensive routing and could benefit from a monitoring service which tracks the reliability of nodes in the network to enable better routing algorithms. Coil, headed by Interledger co-founder Stefan Thomas, is creating a service for micropayments using Interledger. They are a client facing company and work directly with online content creators and consumers. They have interest in supporting the continued development of a monitoring service that can help establish trust and visibility into Interledger as they work to bring it into mainstream usage.

## 3.3 Relevant Statistics

With an open network there are, of course, many interesting statistics that can be collected. The statistics I have chosen to focus on initially collecting are latency, exchange rates, payment bandwidth, and routing.

### 3.3.1 Latency

Network latency measures the length of time from the start of packet transmission from the sender to the end of packet reception at the receiver [9]. For my project I measure round-trip network latency, which is the sum of the one-way latencies from the source to destination and vice versa. The simplest metric offered by pinging other connectors on the network is the ability to see which connectors are live. More complex measurements can be made by sending multiple pings to a peer on the network and calculating statistics such as packet loss percentage and the minimum, maximum, average, and standard deviation times. Periodically pinging other connectors on the network is helpful in determining which connectors could be better to route money through and offers visibility into how reliable certain connectors are.

### 3.3.2 Exchange Rates

The effect of measuring and displaying exchange rates of various connectors is helpful for both users of the system and connector operators in the Interledger network. With published exchange rates, connectors will be able to establish trust with the users who peer with them to send payments and those that consistently publish reliable exchange rates will gain recognition for their services. Users will be able to ensure that they are not being charged unreasonable prices and can also determine which connector provides rates that are best for them.

#### 3.3.3 Payment Bandwidth

Similar to the notion of ”bandwidth” on the internet, payment bandwidth is the rate of transfer of packets of value. Connectors are free to choose how much payment bandwidth they provide to the network and could even offer different levels of tiers which users could pay for to gain higher liquidity. Users may or may not require higher payment bandwidths based on the nature of the payments that they are sending at a particular time. Displaying this information for users will provide a way for them to determine which connectors they should use to send a payment. The ability to view the payment bandwidth of other connectors will also likely foster a competitive environment in which connectors will seek to continually improve their services to be more responsive to user demands.

#### 3.3.4 Routing

At the simplest level, displaying routing information is a way for users and other connectors to see what routes between senders and receivers even exist in the network. Beyond this, routing in the network can be vastly improved when armed with data about the network. Connectors can determine where profit can be made by providing routes that do not yet exist. With a combination of knowledge about the above three statistics (latency, exchange rates, and payment bandwidth), users can also determine which routes would be best for them in different situations. Routing in the network is currently price-insensitive with an arbitrary tie-breaking mechanism. A public source of network data will enable much more intelligent routing and therefore better access to liquidity and fast payments.

## 4 Related Work

The Interledger community has undertaken several previous efforts to provide network monitoring solutions and data collection systems. Many of these projects have either stagnated or do not offer a holistic view into the network. I hope to solve both of these problems. I have discovered through extensive interaction with the Interledger community (made up of developers, ecosystem companies, connector providers, and users) that people would be willing to pay for a centralized location which can provide actors on the network with the best possible public statistics such as routing information, exchange rates, latency, and payment bandwidth. Descriptions of several of these previous efforts are below along with short overviews of code or ideas that I adapted from them.

### 4.1 Connector.land

”Connector.land” [19] was the domain name for a website created by a group of developers working on Interledger or for Interledger ecosystem companies. The aim of this project was, somewhat similar to mine, to create a public location which actors on the Interledger network could use to see which connectors were live. To display this data, Connector.land requires a live connector. This connector can either be running locally or have its routing tables port-forwarded to the machine local to the Connector.land web server. The web server then uses the connector’s routing tables to find and ping peers in the network to determine if they are live or not. The Connector.land website is not longer live and its development halted about a year prior to the writing of this paper. The method of data collection used by Connector.land was the foundation for the method of data collection that I have implemented in my service. For my project, I adapted parts of the Connector.land backend code such as the web server and routing table query.

### 4.2 ILP Ping

ILP Ping [11] is a recent contribution to the Interledger community created by a developer named Martin Lowinski. It is a simple command-line tool that utilizes the Interledger Ping Protocol by allowing a user to test the latency of any connector by sending repeated pings and displaying relevant statistics, such as packet loss percentage and minimum/maximum/average/standard deviation times. For my project, I adapted this code for sending repeated pings and calculations of ping statistics.

### 4.3 Moneyd GUI

Moneyd GUI [21] is a graphical extension for moneyd, which is currently one of the fastest ways to access the Interledger payments network. Moneyd GUI is a website that acts as a more user-friendly interface for a local running connector. It displays information such as such as transaction history and routes to peers. It also has functionality for sending and receiving payments. Moneyd GUI has many useful features that can be adapted for this project. However, its main disadvantage (for the purposes of my project) is that is only offers a private view into a local connector. The aim of my project is to provide a public view (using connectors that I run) into the entire Interledger network.

### 4.4 ILP Markets and Monitoring Agent

ILP Markets [12] and the corresponding monitoring agent [13] were created by a developer under the moniker ”N3TC4T”. The monitoring agent, written in python, is a daemon that, similarly to Connector.land, uses the routing tables and various other endpoints of a local running connector to gather data on routing and accounts. While the website displays some useful data, its major drawback is that it cannot fully take advantage of most of the open source software for interacting with the Interledger which is written in JavaScript. Using this open source software is how, in my project, I have been able to prepare and send ping packets to connectors that I am peered with and will be able to programmatically send payments through connectors to discover exchange rates and payment bandwidth information.

## 5 Methods

### 5.1 Approach

To create ILP Monitor I primarily built atop the Connector.land project (detailed in section 4.1). To gain a holistic view into the network and access to an array of routes, I worked on peering with as many connectors on the Interledger livenet as possible. I extended the functionality of the Connector.land project by adding features to its backend and rewriting its frontend, deployed it, and peered with more connectors. Due to various problems with specifications of the Interledger protocols and peering with other connectors on the open network, I was unable to collect and display all the kinds of statistics that I aimed to gather. However, I have theoretically formulated how the implementation could be extended along with protocol specifications that the community is currently working on to include these statistics as well.

### 5.2 Data Collection

#### 5.2.1 Latency

To collect latency statistics I utilized the Interledger Echo protocol, which is still in development and has only been informally documented [2], and standard metrics such as package loss percentage and minimum/maximum/average/standard deviation round trip times (in milliseconds). In addition to round trip times, there is significant interest from the community in measuring one-way trip latency. While round trip time is a useful metric for determining which connectors are live and overall connection time, one-way trip time more closely mirrors the amount of time it may take for a small packet of value to be routed through the network. However, developing the specification for a true ”ping” protocol is still an active area of work in the Interledger community. While an RFC (Request for Comment) has been published [8], none of the current Interledger connector implementations support a one-way ping protocol.

#### 5.2.2 Exchange Rates and Payment Bandwidth

Exchange rates and payment bandwidth information can be collected by sending test payments through my peers in the network. Collecting this information requires setting up a sender and receiver with my connector and routing different sized payments through different peers that I am connected to. Connectors may provide services for different levels of payment bandwidth and may charge different exchange rates based on how much money is being routed through them. These rates can also change over time. Therefore, to most accurately measure exchange rates and payment bandwidth for connectors, I will need to periodically send different amounts of test payments (small, medium, and large sized) from the senders and receivers that I set up over the network. Since connectors can make money from charging flat fees per packet, access to different levels of payment bandwidth, or spreads on currency conversion, there has been interest from the community in turning this aspect of the monitoring platform into a paid service.

#### 5.2.3 Routing

Routing information is automatically collected when you peer with other connectors on the network. This routing information can be graphically displayed. A combination of the above data (latency, exchange rates, and payment bandwidth information) can be used to determine if a certain route or connector matches the needs of a particular user. In the future, the Interledger routing protocol can use these "preferences" of peers to create a smarter routing algorithm. To incentivize connectors to route payments in a particular way according to user preferences, they can be rewarded based on how well their routing conforms to these preferences. A service that provides visibility into the network will make this algorithm robust against connectors who try to cheat the system by holding them accountable by publicly displaying their rates. The Interledger community is also currently discussing the implementation of an auto-peering protocol [1]. The network could also create a method of smarter routing by using an auto peering protocol guided by heuristics (such as a user's payment preferences). ILP Monitor could provide the data that powers the heuristics this algorithm uses.

### 5.3 Implementation

The ILP Monitor prototype is a website that requires access to the routing tables of a running connector. It uses the routing tables to display which connectors the running connector has established routes to and latency statistics to those connectors. Its backend was implemented in pure Javascript using a Koa web server, while the frontend was implemented in Javascript/HTML/CSS using React. The code repository is open source and can be accessed here. Major code samples are shown and fully explained in Appendix A.

#### 5.3.1 Deployment

To keep a live connector running consistently with an externally accessible IP address, I am running a connector with the web server for ILP Monitor on a Google Cloud virtual machine at the following address: `http://34.74.215.62:6001/`[1]. To easily start and configure the connector and speed up the process of peering with others on the network, I am using Strata Labs' easy connector bundle [10]. While the connector is running, the server can query its endpoints to get peering and routing information, which is then used to collect and display statistics. The running connector does not have to be local to the server. As long as the connector's endpoints are accessible (which can be done via port forwarding) the server can perform its queries.

I have deployed both a livenet and testnet connector on the Google Cloud virtual machine. Currently, only one is running at a particular time. A natural extension of this project is to create functionality to

[1]Loading the data display can take up to 10-15 seconds. After querying the local running connector for its routing information, the website waits for all asynchronous pings to connectors to finish before displaying results. Fixing this requires slightly reworking the frontend design.

display both the livenet and testnet monitoring systems at the same time (on different pages of the website, for example). Showing information on the testnet can be very helpful for developers on Interledger as they seek to address problems and fully understand how their applications perform before they deploy on the main network.

#### 5.3.2 Peering

Peering on the Interledger live and test networks is currently still quite an arduous and manual process. To find other people running connectors to peer with on the network, I reached out to ecosystem companies and individuals in the community through slack channels, gitter, forum posts, emails, and phone calls. Even after establishing connection with a peer, various errors regarding authentication, payment channel creation, and mismatching asset transfer scales can slow down successfully finishing the process.

The Interledger address of my connector is `g.ns-live-gcp`. The connectors that I have currently successfully peered with and am broadcasting routing information to on the testnet and livenet are the following:
**Livenet:** `g.kava,g.kka72,g.kd1jm1,g.n3tc4t`
**Testnet:** `test.pumbaa,g.testnet.n3tc4t`

I was only able to peer with many of these connectors very recently due to errors with both the open source software and connector configurations, so at the time of the writing of this paper I had not yet fully implemented all of the data collection methods detailed in this paper.

### 5.4 Alternative Approaches and Further Considerations

There are several other approaches to this problem that I considered before settling on my current approach. At present, there are also several unsolved problems in the Interledger protocols, open source software, and implementation that must be addressed.

Another method of collecting data from connectors for public display would be to leverage the Interledger connector `ADMIN_API` that (by default) runs on port 7769 of the machine local to the connector process. One can insert code to publish statistics that are collected by the `ADMIN_API` to a public website. This would be an opt-in service for connectors on the network where they could choose which statistics to publicly share. The drawback of this method of discovering statistics in the open network is that it relies too heavily on the goodwill of unknown connectors in the network. Some connectors may not wish to share their data or could modify the software they run to submit false data. While network monitoring can have tremendous benefits, if a malicious agent were to subvert a trustworthy source of information the network could suffer drastic consequences. Other approaches of collecting and displaying data are similar to those employed in the currently existing Moneyd GUI and ILP Markets software (detailed in sections 4.3 and 4.4).

Further considerations must be made with regards to problems that still exist within the Interledger Ping protocol and the ILP Monitor implementation. The Echo protocol is still fairly new and there has been some discussion in the Interledger community for its revision or deprecation [7] due to dangers that expose cross-currency connectors [15]. Given that running cross-currency connectors is one of the main use cases for Interledger which ecosystem companies and developers are already building on top of, it seems that the community requires more concrete definitions of how a Ping protocol should be structured in the Interledger network before it should be deployed for widespread use in the main net.

In its current form, ILP Monitor still only gives a single view into the network. For example, if my running connector is peered to another connector in America, and the connector in America is peered with a well-connected connector in Europe, the European one may appear to have more latency than the American connectors despite its position or well-connectedness in Europe. To more accurately represent connector information may require setting up virtual machines all over the world to run the ILP Monitor software and aggregate statistics collected from these instances. These aggregate statistics could which factor in the relative locations of connectors could help users better determine which connectors would be best for them to connect with on the network.

## 6 Results and Analysis

All results can be publicly viewed on the ILP Monitor website.

### 6.1 Routes

The routing tables published by my connector (see appendix B) contain routes to my direct peers as well as the routes that they broadcast to me. The path to each connector I have established a route to is displayed along with the ping statistics collected from each connector. From the data, the following conclusions can be made:

1. There are 20 public connectors that I have discovered on the network so far, 4 of which I am peered with. Right now, the most central connector appears to be `g.kd1jm1`, which direct routes to 7 peers and propagates routes (to me) to 8 peers. This connector is run by a developer from the UK named Kev King.
2. The connectors of some important ecosystem companies - Strata Labs, Kava Labs, Gatehub, Coil, and Stronghold - show very few direct routes to others, if they show up at all, in my network. This is likely because these ecosystem companies take advantage of the manual peering process to run their own private networks of connectors, which can be specially configured and holistically monitored for their own special business needs. Going forward, it seems that the Interledger community should formalize a distinction between methods that should be used to run private networks vs public networks of connectors, especially as development of an auto-peering specification continues.

### 6.2 Ping

The ILP Monitor website currently successfully pings 14 out of the 20 connectors that my connector has established routes with. Each connector is pinged 4 times. These pings are used to calculate packet loss and the other displayed statistics. As one might expect, high latency generally corresponds with more network hops to a particular connector. One can determine paths which have connectors that are relatively close together using a combination of the ratio of time to the number of hops and intuition from the naming of connector addresses. For example, the route to `g.coop` requires three network hops and has relatively low latency. This makes sense because the echo packet is routed through connectors `g.kd1jm1` and `g.kd2jm2`, both of which are run by Kev King.

It may be surprising to some to see that latency is so high. Some connectors which are three hops away take as long as 950ms to respond. Typical pings to the United States take only about 150ms. This timing difference could be a result of a couple different things. First, we must keep in mind that these "ping" packets elicit a response conforming to the Interledger echo protocol, which requires the receiver to not just send a response, but a packet of their own back to the sender as well. Second, Interledger code libraries are built atop higher level Javascript APIs while the ping tool in most computers exists at a low level. Timing could also be affected by these differing levels of abstraction.

Ping errors are due to three kinds of error messages. The errors and the connectors that they affect, in addition to some analysis as to why these errors exist, are detailed below:

1. **F00 - unknown account id:** The pings to the connector `g.kava` fail with this error message. The `F00` error message represents the result of a bad request from the sender. Sending an echo request to the

connector `g.kava` is due to the fact that at the time of the writing of this paper, Kava Labs disabled the ping (echo protocol) functionality of their connector due to concerns with using the echo protocol with cross-currency connectors. While there is a solution to this problem and an issue has been opened on the Github repository for the Interledger Connector implementation in rust, this remains an open issue.

2. **F01 - unexpected ping response:** Pings to three connectors - `g.kd1jm1`, `g.kd2jm2`, and `g.iofv` - fail with this error message. The `F01` error message represents an invalid packet. At the time of the writing of this paper, however, it is unclear whether or not this is an invalid packet from the sender (me) or the receiver (one of these connectors), since the echo protocols requires the receiver to respond with a packet of their own conforming to the echo protocol. My intuition for this error is that these three connectors do not support functionality for the echo protocol (at the time of the writing of this paper). However, this remains an open issue as I work with the corresponding connector operators to debug.

3. **T00 - packet expired:** This error is propagated after pings to the connectors `g.strata` and `g.gatehub-0`. The `T00` error message represents a temporary internal error (likely the result of a bug or unhandled error case). This implies that the most likely cause of these ping failures is that the connectors `g.strata` and `g.gatehub-0` are currently not running and undergoing updates or fixes.

## 7 Discussion

### 7.1 Status of the Interledger Network

Working on this project confirmed my hypothesis that the Interledger community would be interested in a monitoring service and that connectors would be willing to peer and have their information publicly displayed on a website. The Interledger network is still a very small community of developers and ecosystem companies running connectors, so this is the perfect stage of the development of Interledger to establish a strong monitoring presence over the network.

Since peering in the network is still quite a manual process, the network of connector operators is tightly knit and is utilizing multiple communication channels to enable the discussion of new features and speed up debugging errors. These platforms include the Interledger forum, slack channels, gitter, community calls, and the annual ILP Summit. Governance of the Interledger project is an important activity that primarily occurs on these platforms. The community is still small enough that consensus is often reached after discussion on the forum or community calls.

From the perspective of an independent developer working to build on Interledger, there is a very steep learning curve to working with the technology and the community is slow moving. While the forum remains fairly active (there are new posts almost every day) and people are quick to respond on messaging platforms such as slack, the open source repositories are not very active and the status of documentation is terrible. New users are almost completely reliant on reaching out to members of the community to fix issues. Although there is a store of knowledge being built up on the forum, it would be much more useful to have clear documentation on how different aspects of the stack work together to create the network.

Still in its early stages, there have been no publicly documented attacks on Interledger connectors or applications. The theme of security is present in almost all discussions as an attack could fatally cripple the network and greatly affect ecosystem companies and potential investments. A network monitoring service will increase the visibility of attacks that will likely only grow in number as the network continues to grow.

### 7.2 Creating Trust and the Role of Centralization in Decentralized Systems

Blockchain conservatives may object to this project due to the somewhat counter-intuitive notion of a ”decentralized” system that relies on a ”centralized” service. However, to dissect and reject this objection we must critically evaluate the role that centralization plays decentralized distributed networks such as Interledger and blockchains.

The nature of the massive price volatility of most cryptocurrencies should be enough to show that, currently, the public does not trust the stability or reliability of fully decentralized and anonymous technologies. With little visibility into what is actually going on in a network, users cannot have any faith that their assets will maintain their value over any period of time. In order for newcomers to trust a system, they need to be able to critically evaluate for themselves how reliable that system is through analysis of its activity, problems, actors, and functionality.

Bootstrapping and maintaining trust in a decentralized system, therefore, requires reliable systems which can accurately display public data and alerts when nodes run into problems. Currently, the best way of building such reliable systems is through centralization of monitoring (making such a service decentralized would cause us to run into many of the same problems we see in current implementations of blockchain systems). Establishing trust early on through communication with involved actors of the network such as developers and ecosystem companies and maintaining this trust through open source software and clear documentation is the best way to ensure that the benefits of a system such as ILP Monitor can be fully realized.

## 8 Conclusion

I have introduced ILP Monitor, a platform to publicly display statistics on Interledger connectors, which make up the Interledger payments network. Currently, the most relevant statistics for the Interledger community are latency, payment bandwidth, and exchange rates. These statistics can also be combined to develop better routing and auto-peering protocols for the network, which are currently in development. The initial prototype for ILP Monitor displays latency and routing data for connectors in the network. Future work involves peering with more connectors in the network, extending the implementation to show more statistics and setting up senders and receivers to route test payments through the network to discover exchange rate and payment bandwidth information, and continuing to work with the community to develop specifications of robust protocols that may also be able to leverage the connector data from ILP Monitor.

## 9 Acknowledgements

I would like to thank the CS 236R course staff - Professor Yiling Chen and Teaching Fellow Daniel Moroz - for all of their support and advice throughout the semester. I would also like to thank everyone in the Interledger community who gave me very helpful advice and critical feedback throughout the ideation and implementation of this project. I would especially like to thank David Fuelling, Dino Rodriguez, Evan Schwartz, Austin King, and Christian Ondaatje for their involved guidance, suggestions, and helpful debugging.

## References

[1] Adrian Hope Bailie. Auto-peering. https://forum.interledger.org/t/auto-peering/548. April 26, 2019.

[2] Adrian Hope Bailie. Document the ping protocol. https://forum.interledger.org/t/document-the-ping-protocol/279/7. Accessed in April 2019.

[3] Vitalik Buterin. Ethereum white paper. https://github.com/ethereum/wiki/wiki/White-Paper.

[4] Interledger Community. Connector to connector protocol. https://interledger.org/rfcs/0010-connector-to-connector-protocol/. Accessed in April 2019.

[5] Interledger Community. Interledger architecture. https://interledger.org/rfcs/0001-interledger-architecture/draft-5.html.

[6] Interledger Community. Interledger protocol v4. https://interledger.org/rfcs/0027-interledger-protocol-4/draft-6.html.

[7] David Fuelling. Deprecate the echo protocol and replace with one-way ping? https://forum.interledger.org/t/deprecate-the-echo-protocol-and-replace-with-one-way-ping/520. Accessed in April 2019.

[8] David Fuelling. Interledger ping protocol. https://github.com/interledger/rfcs/blob/df-ping-protocol/0000-ilp-ping-protocol/0000-ilp-ping-protocol.md. Accessed in April 2019.

[9] Rony Kay. Pragmatic network latency engineering: Fundamental facts and analysis. Technical report, cPacket Networks, 2009.

[10] Strata Labs. Easy connector bundle. https://github.com/d1no007/easy-connector-bundle. Accessed in April 2019.

[11] Martin Lowinski. Ilp ping. https://github.com/martinlowinski/ilp-ping. Accessed in April 2019.

[12] N3TC4T. Ilp markets. https://github.com/N3TC4T/ilp-monitoring. Accessed in April 2019.

[13] N3TC4T. Ilp monitoring agent. https://github.com/N3TC4T/ilp-monitoring-agent. Accessed in April 2019.

[14] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008.

[15] Kincaid O'Neil. Dangers of the echo protocol for cross currency connectors. https://forum.interledger.org/t/deprecate-the-echo-protocol-and-replace-with-one-way-ping/520/3?u=kincaid. Accessed in April 2019.

[16] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. https://lightning.network/lightning-network-paper.pdf, January 14, 2016.

[17] Sean Rowan. Flare network forum post. https://forum.interledger.org/t/flare-a-turing-complete-federated-byzantine-agreement-network/384. March 13, 2019.

[18] Stefan Thomas and Evan Schwartz. A protocol for interledger payments. https://interledger.org/interledger.pdf, 2015.

[19] Various. Connector.land. https://github.com/interledger/connector.land. Accessed in April 2019.

[20] Various. Ilp connector. https://github.com/interledgerjs/ilp-connector. Accessed in April 2019.

[21] Various. Moneyd gui. https://github.com/interledgerjs/moneyd-gui. Accessed in April 2019.

## A Code Samples

The main extension I made to the Connector.land backend code was functionality to measure ping statistics. Below is the main ping functionality which measures the round-trip time for a single ILP packet to be routed through the network.

```
1  async ping (destination) {
2    const fulfillment = crypto.randomBytes(32)
3    const condition = crypto.createHash('sha256').update(fulfillment).digest()
4    const { clientAddress } = await ILDCP.fetch(this.plugin.sendData.bind(this.plugin))
5    this.conditionMap.set(condition.toString('base64'), fulfillment)
6    const writer = new Writer()
7    writer.write(Buffer.from('ECHOECHOECHOECHO', 'ascii'))
8    writer.writeUInt8(0)
9    writer.writeVarOctetString(Buffer.from(clientAddress, 'ascii'))
10   const start = process.hrtime();
11   const result = await this.plugin.sendData(IlpPacket.serializeIlpPrepare({
12     destination,
13     amount: '1',
14     executionCondition: condition,
15     expiresAt: new Date(Date.now() + 30000),
16     data: writer.getBuffer()
17   }))
18   const diff = process.hrtime(start);
19   const latency = diff[0] * 1000 + diff[1] / 1000000;
20   const parsedPacket = IlpPacket.deserializeIlpPacket(result);
21   return { parsedPacket, latency };
22 }
```

This code is an implementation of the the Interledger echo protocol. Conditions/fulfillments are needed to successfully send payments between senders and receivers. Protocols such as the simple payment setup protocol (SPSP) provide a method of establishing a shared fulfillment between a sender a receiver for sending money. Lines 2 and 3 in this code snippet create a random fulfillment and its corresponding condition (which, given the data written to the packet in lines 7 through 9, connectors are configured to accept based on the echo protocol). Since this code is being run by the ILP Monitor backend, which is not connected to the Interledger network and instead uses the local running connector as a parent to connect to the network, it needs a temporary address. This address is generated in line 4. Line 11 is where the echo request is actually sent out to a specific destination and timed for how long that destination takes to respond. The call to `process.hrtime(start)` in line 18 measures the interval since line 10 and returns a high resolution result of `[seconds, nanoseconds]`. This is why the latency computation in line 19 involves the combination of two separate values.

## B Connector Routing Tables

Running connectors publish their routing tables at the endpoint `<hostname>:7769/routing`. The two main routing tables are the `localRoutingTable` and `forwardRoutingTable`. The `localRoutingTable` is used to determine where the connector should route packets that have come into it. The `forwardRoutingTable` contains entries that are forwarded to other connectors as part of the Interledger connector-to-connector protocol [4] (which is the primary way that routes are shared between connectors)[2]. To view the routing tables of my connector running in Google Cloud, I port-forwarded port 7769 over SSH from the cloud server to my local machine. Below is the most recent local routing table published by my connector running in the Google Cloud virtual machine. The names prefixed by ”g.” are ILP addresses which represent different connectors on the network. The ”nextHop” fields contain the account names I have configured for my peers on the network (for example, the account ”kavaLive” corresponds to the connector ”g.kava”).

```
"localRoutingTable":{
   "g.ns-live-gcp":{"nextHop":"","path":""},
   "g.ns-live-gcp.local":{"nextHop":"local","path":""},
   "g.ns-live-gcp.xrpServer":{"nextHop":"xrpServer","path":""},
   "g.kava":{"nextHop":"kavaLive","path":"g.kava"},
   "g.kd1jm1":{"nextHop":"kevkingLive1","path":"g.kd1jm1"},
   "g.n3tc4t":{"nextHop":"n3tc4t","path":"g.n3tc4t"},
   "g.aurora":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.aurora"},
   "g.valuework":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.aurora g.valuework"},
   "g.iofv":{"nextHop":"n3tc4t","path":"g.n3tc4t g.iofv"},
   "g.dora-gt":{"nextHop":"n3tc4t","path":"g.n3tc4t g.iofv g.dora-gt"},
   "g.ilpocean":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.aurora g.ilpocean"},
   "g.RallyDev1":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.RallyDev1"},
   "g.john":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.john"},
   "g.kka72":{"nextHop":"kka72Live","path":"g.kka72"},
   "g.vertigo":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.vertigo"},
   "g.zero":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.vertigo g.zero"},
   "g.hopebailie":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.vertigo g.hopebailie"},
   "g.coop":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.kd2jm2 g.coop"},
   "g.kd2jm2":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.kd2jm2"},
   "g.strata":{"nextHop":"kka72Live","path":"g.kka72 g.kd1jm1 g.aurora g.strata"},
   "g.gatehub-0":{"nextHop":"kka72Live","path":"g.kka72 g.kd1jm1 g.aurora g.strata
       g.gatehub-0"},
   "g.aurora-ilsp2":{"nextHop":"kevkingLive1","path":"g.kd1jm1 g.aurora
       g.aurora-ilsp2"}
}
```

Below is the forward routing table published by my connector, which is very similar to the local routing table:

```
"forwardingRoutingTable":{
   "g.ns-live-gcp":{"nextHop":"","path":"g.ns-live-gcp"},
   "g.kava":{"nextHop":"kavaLive","path":"g.ns-live-gcp g.kava"},
   "g.kd1jm1":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1"},
   "g.n3tc4t":{"nextHop":"n3tc4t","path":"g.ns-live-gcp g.n3tc4t"},
   "g.aurora":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.aurora"},
   "g.valuework":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.aurora
       g.valuework"},
   "g.iofv":{"nextHop":"n3tc4t","path":"g.ns-live-gcp g.n3tc4t g.iofv"},
   "g.dora-gt":{"nextHop":"n3tc4t","path":"g.ns-live-gcp g.n3tc4t g.iofv g.dora-gt"},
   "g.ilpocean":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.aurora
       g.ilpocean"},
   "g.RallyDev1":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1
```

[2]The specification of the CCP protocol, however, has been deprecated and is not very well documented. There is currently discussion in the community regarding better methods of automatically discovering connectors to peer with. One potential solution was Connector.land, but that service is no longer running. ILP Monitor is currently very well positioned to provide this service.

```
    g.RallyDev1"},
  "g.john":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.john"},
  "g.kka72":{"nextHop":"kka72Live","path":"g.ns-live-gcp g.kka72"},
  "g.vertigo":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.vertigo"},
  "g.zero":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.vertigo
    g.zero"},
  "g.hopebailie":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.vertigo
    g.hopebailie"},
  "g.coop":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.kd2jm2
    g.coop"},
  "g.kd2jm2":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.kd2jm2"},
  "g.strata":{"nextHop":"kka72Live","path":"g.ns-live-gcp g.kka72 g.kd1jm1 g.aurora
    g.strata"},
  "g.gatehub-0":{"nextHop":"kka72Live","path":"g.ns-live-gcp g.kka72 g.kd1jm1
    g.aurora g.strata g.gatehub-0"},
  "g.aurora-ilsp2":{"nextHop":"kevkingLive1","path":"g.ns-live-gcp g.kd1jm1 g.aurora
    g.aurora-ilsp2"}
}
```