Housekeeping

- Road Map of 2018 - More meet ups!
- Thank you to our sponsor
  - Vectr Ventures
- Message from the Linux Foundation
Agenda

● Brief Introduction to Interledger protocol
● About Hyperledger Quilt
● Interledger Deep Dive
● Demo
● How to Contribute To Interledger
● How to Contribute to Quilt
● Q&A
What is a Quilt?

This is what Quilt looks like! From a Google Search of “Quilt”
What is Hyperledger Quilt?

Quite Simply...

Quilt is an implementation of the Interledger protocol in Java.

Leading to More Questions...

- What is the Interledger Protocol?
- Why is a Java Implementation required?
What is Interledger? What is Interledger relationship to Quilt?

What is Interledger?

Interledger is a protocol (similar to HTTP).

It is a protocol for payments across payment systems.

Interledger enables secure transfers between ledgers and allows anyone with accounts on two ledgers to create a connection between them.

A protocol for internetworking different payment networks.

Is a Network of Networks

What is Quilt?

Quilt is an implementation of such protocol (similar to how Tomcat, Jetty, … are implementations of HTTP/s).

SOURCE: https://interledger.org/interledger.pdf
Interledger Deep Dive
Interledger Protocol Suite: What is Interledger Make up of?

Collapsed Architecture View

Expanded Architecture View
Interledger Architecture

Interledger provides for secure payments across multiple assets on different ledgers.

The architecture consists of a conceptual model for interledger payments, a mechanism for securing payments, and a suite of protocols that implement this design.

1. Secure Payments

2. Multiple Assets

3. Different Ledgers
The Interledger Protocol (ILP) is the core of the Interledger protocol suite.

Colloquially the whole Interledger stack is sometimes referred to as "ILP".

*Technically, however, the Interledger Protocol is only one layer in the stack.*
Interledger Model (1/3) - Overview

The Interledger Model consists of:

1. Sender
2. Connectors
3. Ledgers
4. Receiver
A Connector is a system that facilitates payments across different ledgers. Connectors generate revenue from Interledger payments while accepting some risk.

Connectors are described as separate logical systems even though the same entity may operate a ledger and a connector.

A connector receives a local transfer on one ledger in exchange for making another local transfer on a different ledger. A single interledger payment may include multiple connectors and may traverse any number of ledgers.

If the ledgers represent different assets, the connectors set the exchange rate between the transfers. Connectors may generate revenue from the difference in value between incoming and outgoing transfers. Senders may request quotes from multiple connectors to determine the best price before sending a payment.

Connectors *peer* with one another to exchange information used to determine the best route for a payment.
Ledgers are stateful systems that track the ownership of assets. Ledgers contain buckets of assets known as accounts and record transfers between them.

Each account has a *balance*, which is the amount of the ledger's assets the account holds. Account balances may be positive or negative, representing assets or liabilities.

A ledger only tracks a single asset, which may be a currency, stock, commodity, etc. One entity that maintains accounts denominated in multiple assets is described as having multiple ledgers.

A ledger may be operated by a single entity, as in the case of a bank, or it may be decentralized, as in the case of a blockchain or distributed ledger.
Interledger Payment Method

```json
{
  address: "g.crypto.btc.1BvBMSEY...",
  amount: "10",
  condition: "aGh2n8gV29ybGQhCg...",
  data: "SGVsbg8gV29ybGQhCg...
}
```
The Interledger Protocol Suite

The Interledger protocol suite may be used to transact across any ledgers and connectors, whether they are public or private. There is no single network that all parties must connect to to use these protocols. It is decentralised in its very nature.

The Interledger is a conceptual network made up of independent and diverse ledgers linked by connectors. Each account "on the Interledger" is part of a particular ledger, but they may transact with others by sending interledger payments through different ledgers and connectors. Like the Internet, the Interledger is not a single network but is comprised of multiple interconnected networks.
Interledger Security - Conditional Transfers

Interledger uses **conditional transfers** to secure payments across multiple hops and even through untrusted connectors.

Senders are guaranteed cryptographic proof that the receiver got the payment or their money back, no matter how many ledgers and connectors are in between.

Connectors take some risk, but this risk can be managed and is primarily based upon the connector's chosen ledgers and direct peers.
Conditional Transfers: Prepared -> Executed or Reject

Each local transfer is first prepared and then either executed or rejected.

When a transfer is prepared, the ledger puts the funds of the source account on hold with a cryptographic condition and timeout. If the condition is fulfilled before the timeout, the transfer is executed and the funds are transferred. If the timeout is reached, the transfer expires and the ledger returns the funds to the source account automatically.
How does the Condition Look Like?

Inspired by the Lightning Network (http://lightning.network), Interledger uses the digest of SHA-256 hash function as the condition for transfers.

The fulfillment is a valid 32-byte preimage for the hash specified when the transfer was prepared. Ledgers are responsible for validating fulfillments. [Transport Layer](#transport-layer) protocols are used by the sender and receiver to generate the condition for a particular payment.
Crypto-Conditions

```json
{
  ...,
  condition: "aGh2n8gV29ybGQhCg...",
  data: "SGVsbG8gV29ybGQhCg..."
}
```

SHA-256 hash of the fulfillment (base64url encoded)

cryptographically bound to payment data (provides integrity protection)
Payment Flow (1/2)

In Interledger payments, all component transfers are prepared before any are executed. No funds are transferred, so none can be lost if a connector fails or attempts to redirect the payment.

Interledger: Two-Phase Execution Secures Multi-Hop Transfers

REFERENCES
S. Thomas and E. Schwartz, A Protocol for Interledger Payments, 2015
Payment Flow (2/2)

When the receiver is notified of funds on hold for them, they submit the fulfillment of the cryptographic condition to claim their funds. Each connector uses the same fulfillment to claim their incoming transfer.

The timeout of each successive transfer is shorter than the previous one, giving each connector a window of time to deliver the fulfillment even if their outgoing transfer was executed at the last possible moment.
Interledger Protocol Suite

1. Ledger Layer
2. Interledger Layer
3. Transport Layer
4. Application Layer
Ledger Layer

In order to facilitate transfers between accounts, ledgers must implement some API or protocol. This is called the ledger layer. There is a wide variety of ledger layer protocols, corresponding to the many different types of ledger.
The Interledger layer is responsible for facilitating payments across multiple ledgers. It is comprised of two key components that are used together: the Interledger Protocol (ILP) and the Interledger Quoting Protocol (ILQP).
Transport Layer

Transport layer protocols are end-to-end protocols used by the senders and receivers of Interledger payments to determine the payment condition and other details. The guarantees afforded to the sender vary depending on the type of transport protocol used.
Application Layer

The application layer is the top layer of the Interledger protocol suite.

Protocols on this layer are responsible for:

1. Destination account discovery
2. Destination amount negotiation
3. Transport protocol selection and communication of associated details, such as the shared secret or condition
4. Additional details to be communicated in ILP packet data
Interledger Protocol Suite

1. Ledger Layer
2. Interledger Layer
3. Transport Layer
4. Application Layer
Crypto Conditions

Known implementations of Crypto Conditions:

- JavaScript: [https://github.com/interledgerjs/five-bells-condition](https://github.com/interledgerjs/five-bells-condition)
- Java: [https://github.com/hyperledger/quilt/tree/master/crypto-conditions](https://github.com/hyperledger/quilt/tree/master/crypto-conditions)
- Python: [https://github.com/bigchaindb/cryptoconditions](https://github.com/bigchaindb/cryptoconditions)
- Go:
  - [https://github.com/go-interledger/cryptoconditions](https://github.com/go-interledger/cryptoconditions)
  - [https://github.com/jtremback/crypto-conditions](https://github.com/jtremback/crypto-conditions)
- Haskell: [https://github.com/libscott/cryptoconditions-hs](https://github.com/libscott/cryptoconditions-hs)
- C: [https://github.com/libscott/libcryptoconditions](https://github.com/libscott/libcryptoconditions)

To create a transaction that transfers an asset to new owners, one must fulfill the asset's current conditions (crypto-conditions).

The most basic kinds of conditions are: A hashlock condition: One can fulfill a hashlock condition by providing the correct “preimage” (similar to a password or secret phrase)

Interledger Protocol (1/2)

The purpose of the interledger protocol is to enable hosts to route payments through an interconnected set of ledgers.

How is it achieved?

This is done by passing the payments from one interledger module to another until the destination is reached. The interledger modules reside in hosts and connectors in the interledger system. The payments are routed from one interledger module to another through individual ledgers based on the interpretation of an interledger address. Thus, a central component of the interledger protocol is the interledger address.
Interledger Protocol (2/2)

When routing payments with relatively large amounts, the connectors and the intermediary ledgers they choose in the routing process may not be trusted. Holds provided by underlying ledgers MAY be used to protect the sender and receivers from this risk.

In this case, the ILP packet contains a cryptographic condition and expiration date.

What are holds?

The protocol uses transfer holds to ensure a sender's funds are delivered to the destination or returned to the sender's account.
Interledger Relation to Ripple

In the Ripple case, for example, the interledger module would call on a local ledger module which would create a Ripple transaction with the interledger packet attached to transmit to the Ripple Consensus Ledger.

The Ripple address would be derived from the interledger address by the local ledger interface and would be the address of some account in the Ripple network, which might belong to a connector to other ledgers.
Internet Architecture

Interledger Architecture

<table>
<thead>
<tr>
<th>Application</th>
<th>HTTP SMTP NNTP NTP RTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport</td>
<td>TCP UDP</td>
</tr>
<tr>
<td>Internetwork</td>
<td>IP</td>
</tr>
<tr>
<td>Network</td>
<td>WIFI BLUETOOTH ETHERNET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application</th>
<th>SPSP HTTP-ILP PAYTORRENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport</td>
<td>SSP PRP</td>
</tr>
<tr>
<td>Interledger</td>
<td>ILP</td>
</tr>
<tr>
<td>Ledger</td>
<td>BITCOIN ETHEREUM RIPPLE</td>
</tr>
</tbody>
</table>
Demo of Interledger
Tying Everything Together
Demo Components

- **five-bells-ledger**: Used as a ledger
- **ilp-connector**: Used for routing payments across ledgers
- **ilp**: Used as a sender and receiver component
- **ilp-plugin-bells**: Plugin for talking to five-bells-ledger
- **five-bells-visualization**: UI component to visualize what is happening
How the Dashboard Looks Like

Each circle is a ledger, and the lines joining them are connectors. Click two ledgers in series to initiate a payment between them along the cheapest path.
Demo of Interledger

ILP Provider using XRP Payment Channels
How it looks like
Contributing to Quilt

A good starting point could be trying to fix any of the code-related issues described at:

https://github.com/hyperledger/quilt/issues/

If you also have experience with Hyperledger Fabric, developing an hyperledger end-point to a token-like smart-contract compatible with ILP can be another interesting project. In practice this means developing a REST API compatible with a generic ILP plugin (this Plugin API, for the JS plugin can be used as reference for a connector plugin connecting to such REST API).

This REST API will map plugin requests to internal smart-contract transactions, and that also means creating internal mechanisms to support HOLD-accounts as well as a (perhaps in golang) implementation of crypto-conditions/crypto-fulfillments.
Common Questions

Q1. Since Quilt is a Java implementation of Interledger, can one assume that the roadmap of quilt hinges on Interledger?

Answer: Yes, that's the case at this moment. (But Interledger development would always welcome feedback and improvements from Quilt)

Q2. I am still confused between Interledger and Quilt.

Answer. Interledger is a protocol (similar to HTTP). Quilt an implementation of such protocol (similar to how Tomcat, Jetty, … are implementations of HTTP/s).

Q3. I am confused with Connectors

Answer. Notice also that Interledger has the concept of connector plugins, that would be something like similar to JDBC drivers. Different java apps can use the same JDBC drivers to connect to different databases in similar ways. Following the Java “way of thinking” Quilt connector plugins could be used to connect different connectors/apps to different “ledgers”.
<table>
<thead>
<tr>
<th>S/N</th>
<th>Questions</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Who and what applications are using interledger</td>
<td>It is still in development so no applications have been as yet. Idea</td>
</tr>
<tr>
<td></td>
<td>protocol at the moment?</td>
<td>was birth in Mid to End 2016.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ripple, however, is used in banks.</td>
</tr>
<tr>
<td>2</td>
<td>What’s the roadmap of Quilt?</td>
<td>The Java implementation is in a bit of flux at the moment and nowhere</td>
</tr>
<tr>
<td></td>
<td></td>
<td>near the feature set of the Javascript.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Our goal is to get the connector to a good state and then consider</td>
</tr>
<tr>
<td></td>
<td></td>
<td>having a way to bridge between that and the existing JS plugins so</td>
</tr>
<tr>
<td></td>
<td></td>
<td>we don't need to re-implement them all.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Roadmap of Interledger can be found at:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="https://github.com/interledger/interledger/issues/">https://github.com/interledger/interledger/issues/</a></td>
</tr>
</tbody>
</table>
| 3 | When reach 1.0? What are the common ledgers which interledger is building on? | Detailed Roadmap of Interledger can be found at: https://github.com/interledger/interledger/issues/  

Last update 2 months back was by the end of Q1 (early April) to have a real-money micropayment network that connects at least XRP, Bitcoin, Ethereum, a robust open source connector implementation, a stable Interledger protocol, and compatible implementations in at least 1 or 2 languages other than JS à i.e., Hyperledger Quilt  

Refer to Screenshots in Documents |
|---|---|---|
| 4 | You are one of the maintainers of a Hyperledger project? What does it mean?  
How can you be one and what did you need to do? | I help to fix issues. Anyone can help! Check out the following.  
https://github.com/hyperledger/quilt/issues |
| 5 | Another contributor is Ripple. What do you need to interact with them? | At this point, all the focus is on Interledger.

Ripple Protocol is targeted at the banks while interledger is targeted at everyone and anyone.

ILP is designed to be open source because the vision is an Internet of Value. That means we need an equivalent of TCP/IP for that Internet. If IP was a closed technology there would be no Internet. |