

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

S. Thomas
E. Schwartz
A. Hope-Bailie
Ripple
July 08, 2016

The Interledger Protocol
draft-thomas-interledger-00

Abstract

This document specifies the Interledger Protocol (ILP). It draws heavily from the definition of the Internet Protocol (IP) defined in [RFC0791]. The interledger protocol is the culmination of more than a decade of research in decentralized payment protocols. This work was started in 2004 by Ryan Fugger, augmented by the development of Bitcoin in 2008 and has involved numerous contributors since then.

Feedback

This specification is a part of the Interledger Project [1] work. Feedback related to this specification should be sent to public-interledger@w3.org [2].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction (#intro)	2
1.1. Scope	3
1.2. Definitions	3
1.2.1. Transfer	3
1.2.2. Ledger	3
1.2.3. Connector	3
1.2.4. Payment	3
1.3. Basic Concepts	3
1.4. Operation	4
2. Overview	4
2.1. Relation to Other Protocols	4
2.2. Model of Operation	5
2.2.1. Without Holds ("Optimistic Mode")	5
2.3. Function Description	8
2.3.1. Addressing	9
2.3.2. Connectors	9
2.4. Specification	9
2.4.1. ILP Header Format	9
2.5. Holds Without Native Ledger Support	11
3. References	12
3.1. Normative References	12
3.2. Informative References	12
3.3. URIs	12
Appendix A. Security Considerations	12
Appendix B. IANA Considerations	12
Authors' Addresses	12

1. Introduction (#intro)

Payment networks today are siloed and disconnected. Payments are relatively easy within one country or if the sender and recipient have accounts on the same network or ledger. However, sending from one ledger to another is often impossible. Where connections do exist, they are manual, slow, or expensive.

The Interledger Protocol provides for routing payments across different digital asset ledgers while isolating senders and receivers from the risk of intermediary failures. Secure multi-hop payments and automatic routing enables a global network of networks for different types of value that can connect any sender with any receiver.

1.1. Scope

The interledger protocol is intentionally limited in scope to provide the functions necessary to deliver a payment from a source to a destination over an interconnected system of ledgers. It includes minimal requirements for underlying ledgers and it does not include public key infrastructure, identity, liquidity management, or other services commonly found in payment protocols.

1.2. Definitions

1.2.1. Transfer

Change in ownership of some asset

1.2.2. Ledger

System which records transfers

1.2.3. Connector

System which relays transfers between two ledgers

1.2.4. Payment

An exchange of assets involving one or more transfers on different ledgers

1.3. Basic Concepts

On the Interledger there are two roles. A ledger is a system of accounts, with balances, and the role of the ledger is to record transfers which change the balances of the accounts on the ledger. A connector is a host holding a balance on two or more ledgers. Connectors trade a debit against their balance on one ledger for a credit against their balance on another as a means of facilitating the payment between the two ledgers.

1.4. Operation

The central functions of the interledger protocol are addressing hosts and securing payments across different ledgers.

Each host sending and receiving interledger payments has an interledger module that uses the addresses in the interledger header to transmit interledger payments toward their destinations. Interledger modules share common rules for interpreting addresses. The modules (especially in connectors) also have procedures for making routing decisions and other functions.

The interledger protocol uses transfer holds to ensure that senders' funds are either delivered to the destination account or returned to the sender's account. This mechanism is described in greater detail in the Section 2 and the Interledger Whitepaper [3].

The interledger protocol treats each interledger payment as an independent entity unrelated to any other interledger payment. There are no connections or channels (virtual or otherwise).

Interledger payments do not carry a dedicated time-to-live or remaining-hops field. Instead, the amount field acts as an implicit time-to-live: Each time the payment is forwarded, the forwarding connector will take some fee out of the inbound amount. Once a connector recognizes that the inbound amount is worth less (though not necessarily numerically smaller) than the destination amount in the ILP header, it will refuse to forward the payment.

2. Overview

2.1. Relation to Other Protocols

This protocol is called on by hosts through higher level protocol modules in an interledger environment. Interledger protocol modules call on local ledger protocols to carry the interledger payment to the next connector or destination account. In this context a ledger may be a small ledger owned by an individual or organization or a large public ledger such as Bitcoin.

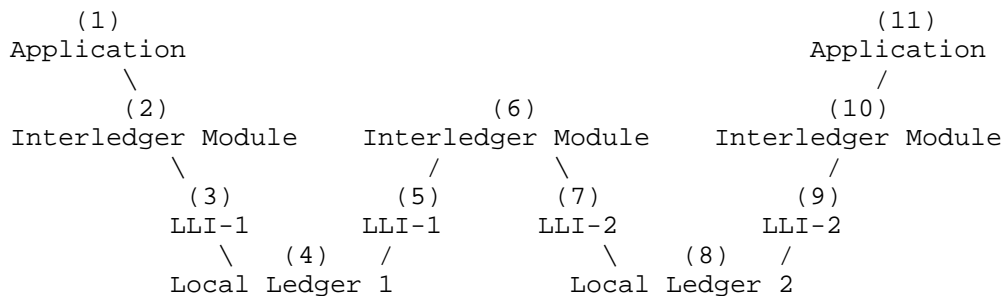
For example, a Simple Payment Setup Protocol (SPSP) [4] module would call the interledger module with the address and other parameters in the interledger packet to send a payment. The interledger module would send a transfer to the next connector or destination account along with the interledger packet and according to the parameters given. The transfer and interledger packet would be received by the next host's interledger module and handled by each successive connector and finally the destination's SPSP module.

2.2. Model of Operation

2.2.1. Without Holds ("Optimistic Mode")

The protocol MAY be used without the security provided by holds - sometimes referred to as "Optimistic Mode". The model of operation for transmitting funds from one application to another without holds is illustrated by the following scenario:

We suppose the source and destination have accounts on different ledgers connected by a single connector.



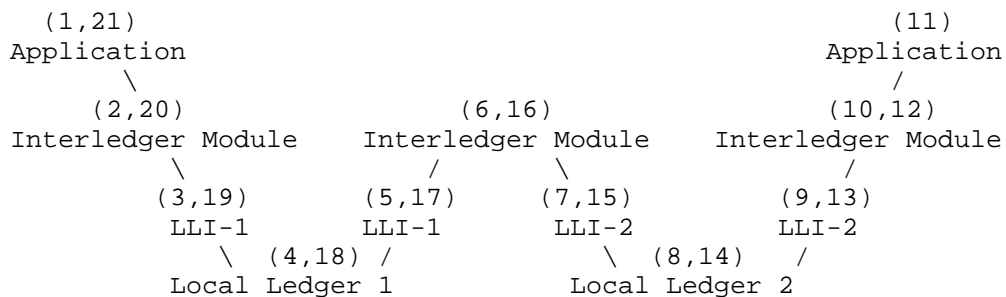
1. The sending application chooses an amount and calls on its local interledger module to send that amount as a payment and passes the destination address and other parameters as arguments of the call.
2. The interledger module prepares an ILP packet and attaches the data to it. The interledger module determines a destination account on the local ledger for this interledger address. In this case it is the account of a connector. It passes the chosen amount and the local destination account to the local ledger interface.
3. The local ledger interface creates a local ledger transfer, then authorizes this transfer on the local ledger.
4. The ledger executes the transfer and notifies the connector.
5. The connector host's local ledger interface receives the notification and passes it to the interledger module.
6. The connector's interledger module extracts the ILP packet from the notification and determines from the interledger address that the payment is to be forwarded to another account in a second ledger. The interledger module converts the amount according to its locally available liquidity and determines the

local account on the other ledger corresponding to the destination host. It calls on the local ledger interface for the destination ledger to send the transfer, which includes the same ILP packet.

7. This local ledger interface creates a local ledger transfer and authorizes it.
8. The ledger executes the transfer and notifies the destination host.
9. The destination host's local ledger interface receives the notification and passes it to the interledger module.
10. The interledger module extracts the ILP packet and determines that the payment is for an application in this host. It passes the transfer data to the application.
11. The destination application receives the notification of incoming funds and reacts accordingly.

2.2.1.1. With Holds ("Universal Mode")

The protocol MAY be used with transfer holds to ensure a sender's funds are delivered to the destination or returned to the sender's account. The model of operation is illustrated with the following example:



1. The sending application uses a higher-level protocol to negotiate the address, an amount, and a cryptographic condition with the destination. It calls on the interledger module to send a payment with these parameters.
2. The interledger module prepares the ILP packet, chooses the account to send the local ledger transfer to, and passes them to the local ledger interface.

3. The local ledger interface creates a local ledger transfer, including the cryptographic condition, then authorizes this transfer on the local ledger.
4. The ledger puts the sender's funds on hold - it does not transfer the funds to the connector - and notifies the connector.
5. The connector host's local ledger interface receives the notification and passes it to the interledger module.
6. The connector's interledger module extracts the ILP packet and determines that it should forward the payment. The interledger module calls on the destination ledger's local ledger interface to send the second transfer, including the same condition as the sender's transfer.
7. The local ledger interface creates a local ledger transfer, including the cryptographic condition, then authorizes this transfer on the local ledger.
8. The ledger puts the connector's funds on hold - it does not transfer the funds to the destination - and notifies the destination host.
9. The destination host's local ledger interface receives the notification and passes it to the interledger module.
10. The interledger module extracts the ILP packet and determines that the payment is for an application in this host. It passes the transfer data to the application.
11. The destination application receives the notification and recognizes that funds are on hold pending the condition fulfillment. It checks the details of the incoming transfer against what was agreed upon with the sender. If checks pass, the application produces the condition fulfillment and passes it to the interledger module.
12. The destination's interledger module passes the fulfillment to the local ledger interface.
13. The local ledger interface submits the fulfillment to the ledger.
14. The destination ledger validates the fulfillment against the held transfer's condition. If the fulfillment is valid and the

transfer is not expired, the ledger executes the transfer and notifies the destination host and the connector.

15. The connector's local ledger interface receives the fulfillment notification and passes it to the interledger module.
16. The connector's interledger module receives the fulfillment and passes it to the local ledger interface corresponding to the source ledger.
17. This ledger interface submits the fulfillment to the source ledger.
18. The source ledger validates the fulfillment against the held transfer's condition. If the fulfillment is valid and the transfer is not expired, the ledger executes the transfer and notifies the connector and the sender's host.
19. The sender's local ledger interface receives the fulfillment notification and passes it to the interledger module.
20. The sender's interledger module receives the fulfillment notification and passes it to the application.
21. The sender's application receives the fulfillment notification and reacts accordingly.

2.3. Function Description

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

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.

2.3.1. Addressing

As with the [RFC0791], interledger distinguishes between names, addresses, and routes. > "A name indicates what we seek. An address indicates where it is. A route indicates how to get there. The internet protocol deals primarily with addresses. It is the task of higher level (i.e., end-to-end or application) protocols to make the mapping from names to addresses."

The interledger module translates interledger addresses to local ledger addresses. Connectors and local ledger interfaces are responsible for translating addresses into interledger routes and local routes, respectively.

Addresses are hierarchically structured strings consisting of segments delimited by the period (".") character. In order to distinguish the present address format from future or alternative versions, the protocol prefix "ilp:" MUST be used:

```
"ilp:us.bank1.bob "
```

Care must be taken in mapping interledger addresses to local ledger accounts. Examples of address mappings may be found in "Address Mappings" ((TODO)).

2.3.2. Connectors

Connectors implement the interledger protocol to forward payments between ledgers. Connectors also implement other protocols to coordinate routing and other interledger control information.

2.4. Specification

2.4.1. ILP Header Format

Here is a summary of the fields in the ILP header format:

Field	Type	Short Description
version	INTEGER(0..255)	ILP protocol version (currently "1")
destinationAddress	IlpAddress	Address corresponding to the destination account
destinationAmount	IlpAmount	Amount the destination account should receive, denominated in the asset of the destination ledger
condition	OCTET STRING	See [draft-thomas-crypto-conditions-00]. The condition may be included in the packet or may be transmitted through the ledger layer.
expiresAt	IlpTimestamp	Maximum expiry time of the last transfer that the recipient will accept

2.4.1.1. version

INTEGER(0..255)

The version of the Interledger Protocol being used. This document describes version "1".

2.4.1.2. destinationAddress

IlpAddress ::= SEQUENCE OF OCTET STRING

Hierarchical routing label.

2.4.1.3. destinationAmount

IlpAmount ::= SEQUENCE { mantissa INTEGER, exponent INTEGER(-128..127) }

Base 10 encoded amount.

2.4.1.4. condition

`IlpCondition ::= Condition</code>`

Crypto-condition in binary format as defined in [draft-thomas-crypto-conditions-00].

When processing a transfer carrying a condition a ledger MUST place a hold on the funds. While the funds are on hold, neither the sender nor recipient are able to access them. Upon receiving a condition fulfillment, a ledger MUST transfer the funds to the recipient if the funds are held, the fulfillment is a valid fulfillment of the transfer condition and the transfer has not yet expired. ("Universal Mode")

The condition is an optional field. If no condition is provided, the funds are immediately credited to the recipient of the transfer. ("Optimistic Mode")

2.4.1.5. expiresAt

`IlpExpiry ::= GeneralizedTime`

Ledgers MAY require that all transfers with a condition also carry an expiry timestamp. Ledgers MUST reject transfers that carry an expiry timestamp, but no condition. Ledgers MUST reject transfers whose expiry transfer time has been reached or exceeded and whose condition has not yet been fulfilled. When rejecting a transfer, the ledger MUST lift the hold and make the funds available to the sender again.

2.5. Holds Without Native Ledger Support

Not all ledgers support held transfers. In the case of a ledger that doesn't, the sender and recipient of the local ledger transfer MAY choose a commonly trusted party to carry out the hold functions. There are three options:

1. The sender MAY trust the receiver. The sender will perform a regular transfer in the first step and the receiver will perform a transfer back if the condition has not been met in time.
2. The receiver MAY trust the sender. The sender will notify the receiver about the intent to transfer. If the receiver provides a fulfillment for the condition before the expiry date, the sender will perform a regular transfer to the receiver.
3. The sender and receiver MAY appoint a mutually trusted third-party which has an account on the local ledger. The sender

performs a regular transfer into a neutral third-party account.
In the first step, funds are transferred into the account
belonging to the neutral third-party. ### Payment Channels

3. References

3.1. Normative References

[draft-thomas-crypto-conditions-00]
Thomas, S., "Crypto Conditions", July 2016.

3.2. Informative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791,
DOI 10.17487/RFC0791, September 1981,
<<http://www.rfc-editor.org/info/rfc791>>.

3.3. URIs

[1] <https://interledger.org/interledger.pdf>
[2] ../0009-simple-payment-setup-protocol/

Appendix A. Security Considerations

TODO

Appendix B. IANA Considerations

TODO

Authors' Addresses

Stefan Thomas
Ripple
300 Montgomery Street
San Francisco, CA 94104
US

Phone: -----
Email: stefan@ripple.com
URI: <http://www.ripple.com>

Evan Schwartz
Ripple
300 Montgomery Street
San Francisco, CA 94104
US

Phone: -----
Email: evan@ripple.com
URI: <http://www.ripple.com>

Adrian Hope-Bailie
Ripple
300 Montgomery Street
San Francisco, CA 94104
US

Phone: -----
Email: adrian@ripple.com
URI: <http://www.ripple.com>