Meeting notes

Date
26 Feb 2020

Hyperledger is committed to creating a safe and welcoming community for all. For more information please visit our Code of Conduct: Hyperledger Code of Conduct.

Call Details
Join from PC, Mac, Linux, iOS or Android: https://zoom.us/my/hyperledger.community.backup
Or iPhone one-tap:
US: +16465588656,4034983298# or +16699006833,4034983298#
Or Telephone:
Dial(for higher quality, dial a number based on your current location): US: +1 646 558 8656 or +1 669 900 6833 or +1 855 880 1246 (Toll Free) or +1 877 369 0926 (Toll Free)
Meeting ID: 403 498 3298
International numbers available: https://zoom.us/u/bAaJoyznP

Attendees
- David Fuelling
- George Roman
- Ian Simpson
- Neil Hartner
- Noah Kramer

Agenda
1. Introductions
2. Discuss open Quilt issues & PRs
   a. Release 1.3.0
      i. Improve PaymentPointer parsing (#441)
      ii. Add Denomination to send money result (#442)
      iii. Send Correct Sender Address in STREAM Payment (fixes #445)
      iv. Fix Array allocation in STREAM sender (#446)
      v. Support standard NIST-recommended AuthTag ByteOrdering in STREAM Encryption Service (#447)
      vi. Clarify Length-prefix contract (#448)
      vii. Make ConnectionNewAddress Frame's address optional (#459)
      viii. Add typeData field to InterledgerPacket (#461)
      ix. Send close frame when all is said and done on a send money request (#464)
   b. Discuss STREAM sender improvements found in https://github.com/interledger-rs/interledger-rs/pull/635 and task this out.
   c. Discuss & Prioritize 1.4 release items (https://github.com/hyperledger/quilt/projects/9)
3. Q&A, misc issues

Goals
- Stakeholder sync-up
## Discussion items

<table>
<thead>
<tr>
<th>Time</th>
<th>Item</th>
<th>Who</th>
<th>Notes</th>
</tr>
</thead>
</table>
| 1min   | Intros                      | All         | • George working to integrate iroha and Quilt.  
• Luri Vinogradov  
• Kincaid, Neil, Noah, Ian, David: Working on ILP in Java and JS. |
| 10min  | Iroha and Quilt             |             | • Iroha is a private permissioned blockchain.  
• Iroha is in C++; Good permission model; lightweight, easy to deploy; use on mobile.  
• No smart contract -  
• Many Private networks all running same code. Intended for private enterprise blockchains (20-30 projects).  
• CBDC in Cambodia (Bakank)  
• Looking to connect/bridge different iroha networks.  
• Focused on financial transactions. |
| 2min   | Release 1.3.0               | David Fuelling |                                                                                                                                       |
| 45min  | Discuss STREAM sender improvements found in https://github.com/interledger-rs/interledger-rs/pull/635 and task this out. | All         | • Kincaid  
  ○ Sender wallet presents some max-amount that will leave your account.  
  ○ StreamSender needs to enforce that this max is the most that leaves the account.  
  ○ Question: Does the lower-level sender need knowledge around how much its delivering?  
  ○ JS StreamController interface: break sender up into as many little state machines as possible (max packets; liquidity congestion; setting the amount and tracking amount paid; pacing).  
• Neil  
  ○ Durability is aimed at knowing the state of any given stream payment.  
  ○ We could go back and retry. Or we could just do another payment.  
  ○ How do we get a fixed amount in receiver's units where things like FX/slipage could be unknown until the sender gets it.  
  ○ What should STREAMSender do if the FX rate goes against the sender and the payment cannot be completed?  
  ○ Kincaid: FX rates are that big of a deal because they don't fluctuate that frequently.  
  ○ Could happen but is unlikely.  
  ○ Bigger issue is sending packets that fail due to rounding errors, ultimately failing the payment.  
  ○ Accurately probing an FX rate is pretty hard because different-sized packets have different rounding errors. Some packets divide nicely but others don't.  
• Ideas  
  ○ Fixed-exchange rate path guarantees could be an interesting solution here.  
  ○ Many mini-state machines to control for each thing independently (e.g., amount delivered should be distinct from time).  
  ○ Try to complete a payment, but plan on failure.  
• Kincaid  
  ○ Majority of incomplete payments are caused by unimplemented features or bugs in implementations.  
  ○ Very tiny fraction of payments are caused by network errors or intermediary manipulation.  
  ○ Create a "guidance doc" for state machines and approaches to building a STREAM sender:  
  ○ Notes on error-cases, better explanation.  
  ○ Discussion around each "mini state machine"  
• Kincaid  
  ○ Opinion on best-practices is evolving. |
| 0min   | Discuss & Prioritize 1.4 release items (https://github.com/hyperledger/quilt/projects/9) | All         |                                                                                                                                       |
| 0min   | Open Discussion             | All         |                                                                                                                                       |

### Action items

- ➡️ Upload call audio to wiki and link here. David Fuelling

### Recordings