Comparing Layer-1 and Layer-2 Blockchain Protocols in Reputation-Based Networks

07-400 Milestone 6 Report

April 22, 2022

1 Major Changes

There have been no major changes since my last milestone report.

2 What I Have Accomplished Since My Last Meeting

I have extended my comparative analysis of storing 4-byte unique identifiers instead of 32-byte public keys up to 1000 accounts for all implemented queries and transactions in my ledger prototype. The setup for the experiment consisted of creating 1000 bolt specifications (and initializing the respective 1000 bolt instance accounts) such that they were uniformly distributed across 100 users. The following table that summarizes the performance results, or the time it takes to perform each task in milliseconds.

Task	4-byte UID	32-byte pubkeys
Querying bolt specs	5.024	0.069
Querying bolt instances (filtration)	5.444	17.231
Querying bolt instances (computation)	10.945	0.497
Querying user wallets	6.479	17.422
Querying issuer specs	6.107	20.066
Minting	448.342	437.060
Transferring	453.908	444.833

For tasks such as querying bolt specifications and bolt instances, holding 32-byte public keys performs a factor of 50-70 times faster than the 4-byte unique identifiers. The method for querying bolt specifications relies on a client library function that retrieves account information given the account public key. The method for querying bolt instances relies on computing the bolt instance public key from the given user and bolt spec public keys and using the prior method in order to retrieve its account information. These results can be attributed to the fact that retrieving information associated with a public key is highly

optimized.
However, as observed last week, the alternative method for querying bolt instances demonstrates that for more complex queries, having 4-byte UIDs improves performance (as well as storage costs!). This is largely because all other queries rely on filtering through all the accounts owned by our smart contract and comparing bytes of account data at specified offsets. With smaller identifiers, we have less data to compare, which means that we begin to see improvements in query performance. We see a 2-4 times performance in tasks such as querying user wallets and querying issuer specifications.
There is little to no difference in times for transactions, such as minting or transferring bolts.

3 Meeting Milestones

Yes, I have met my milestone.

4 Surprises

Although I expected that single queries would be faster than double querying for some of the tasks, I did not expect it to be faster by a factor of 50-70 times! This complicates our decision-making since the tradeoffs are so extreme. We will most likely have to consider how often we make these queries in our application.

5 Looking Ahead

For my final report, I will collect and compare measurements for cost and performance for the same queries and transactions on the existing BoLT ledger implementation and draw a comparative analysis between the two implementations.

6 Revisions to Future Milestones

No revisions to future milestones are needed at this time.

7 Resources Needed

No additional resources are needed at this time.