THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

An Exploration of the Libra Testnet

by

CAMERON JAMES HARDER-HUTTON

School of Information Technology and Electrical Engineering,
The University of Queensland.

Submitted for the degree of
Bachelor of Engineering (Honours)
in the division of Software Engineering

22nd June, 2020

Cameron Harder-Hutton
23 Moriah St
Boondall, QLD 4034
Tel. 0400 879 112

June 22, 2020

Prof Amin Abbosh
Head of School (Acting)
School of Information Technology and Electrical Engineering
The University of Queensland
St Lucia, QLD 4072

Dear Professor Abbosh,

In accordance with the requirements of the degree of Bachelor of Engineering (Honours) in the division of Software Engineering, I present the following thesis entitled "An Exploration of the Libra Testnet". This work was performed under the supervision of Dr Marius Portmann.

I declare that the work submitted in this thesis is my own, except as acknowledged in the text and footnotes, and has not been previously submitted for a degree at The University of Queensland or any other institution.

Yours sincerely,

CAMERON HARDER-HUTTON.

Acknowledgments

I would like to acknowledge my supervisor Dr Marius Portmann, for his help and guidance on undertaking this project. I would also like to thank him for offering this project in the first place, allowing me to do my thesis around a subject I am genuinely passionate about - blockchain.

I would also like to acknowledge the (originally many, now few) developers working on open source projects related to Libra. I am excited for the code from this project - and for myself - to become a part of that group.

Abstract

Libra is a blockchain-based payments platform created and backed by Facebook and their subsidiary company, Novi. On June 18, 2019 the Libra testnet was launched, allowing developers to interact with the platform. This thesis is an exploration of such testnet. It investigates the data which is available to be queried on the testnet and what can be learnt from it. The way in which the testnet has changed over the first 12 months of its development is also explored.

Libra uses a leader-based Byzantine Fault-Tolerant consensus protocol. The role of the leader in leader-based consensus protocols is to receive votes from other nodes in the network, and propagate messages detailing the success/failure of the voting for that round. Despite claiming to be random, the main contribution of this thesis is the discovery that Libra uses a reputation based algorithm for the selection of a round leader in consensus. Validator nodes - the name given to those whose computers participate in the consensus - were less frequently chosen as round leader when they had a greater latency of communication with the rest of the network, causing their votes to be counted less often.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Significance

In 2020, it is hard to see a future in which blockchain technology does not disrupt several industries. In 2019, Venezuelans were able to use Bitcoin and Litecoin as a hedge against hyperinflation as a result of government corruption [1], [2]. Louis Vuitton are able to use blockchain to help customers verify the authenticity of their designer goods [3]. And, healthcare institutions are using blockchain for contact tracing and secure medical records [4], [5].

Blockchain, however, is still very far from widespread consumer adoption as a trustworthy and reliable technology [6]. This is where the advent of Libra makes the space interesting.

Libra was announced on 18 June, 2019, as a new cryptocurrency being developed by Facebook [7]. Libra intends to be a global payment system, and to bring banking to the unbanked populations of the world [8]. Unlike many cryptocurrencies, it intends to be a stable currency [9]. Since April 2020, this definition has expanded to Libra becoming a global payment platform - incorporating many stable cryptocurrencies. This project is a significant validation of blockchain technology to the greater public, and it will be even more significant once it is officially released.

Many people have criticised Libra, stating that it is not a true cryptocurrency, needs regulation [10] or that it poses a threat to the very nature of global finance [11]. There is however, one absolute truth: Facebook has over 2.5 billion monthly active users [12], so regardless of whether you believe Libra is an abhorrent misrepresentation of Satoshi's vision or not, if it can find its legs, it *will* show consumers that there are faster, easier and cheaper ways to send money than using a bank [13].

1.2 Project Scope

The project proposal entailed two parts:

1. The development of a 'blockchain explorer' for the Libra testnet, similar to those made for other blockchains [14], [15], and

2. The development of a custom Libra Node, within a local Libra testnet. This would facilitate running simulations, threshold and stress testing, and observe network commmunication at a low level.

However, as the project proceeded, the second goal was dropped in favour of a more thorough investigation of the first. There were two reasons for this:

1. The endpoints through which information could be gathered from the Libra testnet received breaking changes too frequently, each time requiring effort to update data collection methods.

2. Interesting information regarding validator voting and round leadership was found which warranted further investigation.

As such, the scope of this project is to be able to query and collect data from the Libra testnet. It is the responsibility of technically capable individuals to conduct due dilligence of platforms such as Libra, since they are likely to affect the daily lives of many people in the not too distant future. With this in mind, the further goal of this project is to improve the security and user experience of Libra's end users, by keeping the blockchain accountable for the promises it makes.

1.3 Outline

Chapter 2 contains a detailed, technical explanation of the Libra blockchain: Section 2.1 details Key Concepts, section 2.2 explains the Consensus protocol and section 2.3 describesthe state of the testnet.

Chapter 3 explores related works, with section 3.1 detailing similar work done on other blockchains and section 3.2 looking at other works done on the Libra testnet.

Chapter 4 details the methodology followed in this project. Section 4.1 describes how data was collected, section 4.2 describes how data was stored and section 4.3 describes the assoicated web application.

Chapter 5 presents the data and findings. Section 5.1 presents data related to transactions, section 5.2 presents data related validator voting patterns and round leadership selection. Section 5.3 discusses the implications of these findings, as well as a way for the Libra developers to address the found concerns.

Chapter 6 conludes this thesis. Section 6.1 summarises the findings and section 6.2 describes future work.

Chapter 2

The Libra Blockchain

On June 18 2019, Facebook announced 'Libra' [7]. At this time, the project was just intended to be a digital currency and payment platform. Libra was branded as being a global, stable cryptocurrency - called Libra Coin - to facilitate the unbanked population of the world. [16] Since then, Libra has pivoted goals. Libra is now a payments platform for the aforementioned Libra Coin, and global currency stablecoins. Libra Coin itself is intended to be a stablecoin backed by the Libra Reserve - a basket of cash, cash equivalents and short-term government securities [9].

Libra is a permissioned blockchain, meaning that appoval must be given by a governing body in order to participate in validating transactions on the Libra network. Such governing body is called the Libra Association, an independent membership organisation in Geneva, Switzerland - of which Facebook remains a member. The original white paper intended for Libra to transition to a permissionless blockchain - like Bitcoin and Ethereum - in five years. However, the April 2020 revision of the Libra white paper has removed this intention, citing regulatory dissatisfaction [9], [17]. According to documents released by the Libra Association [18], an organisation must meet the folowing criteria to become a member:

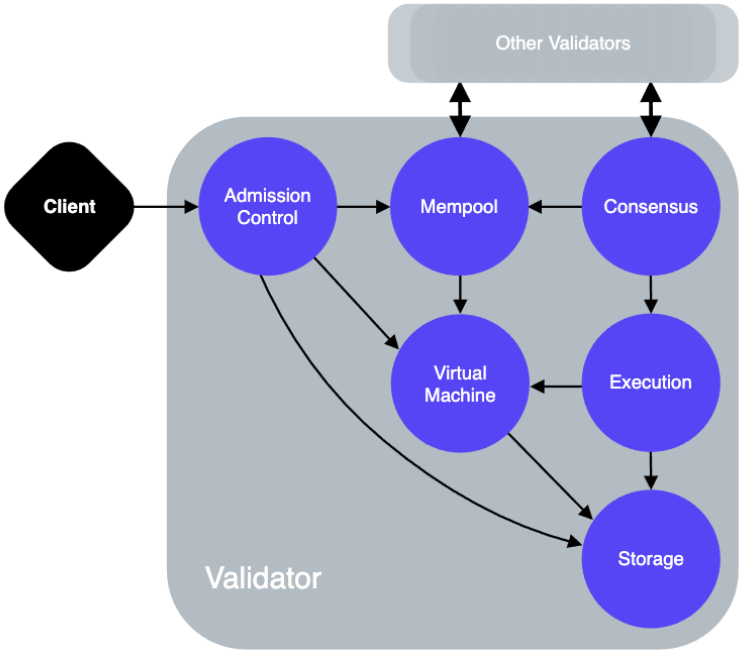
- Be able to run either a self-hosted or cloud-hosted Validator Node,
- Market value greater than $1 billion USD *or* Customer balances greater tha $500 million USD,
- Reach more than 20 million people a year in more than one country,
- Be recognised as a top-100 industry leader.

2.1 Key Concepts

2.1.1 Nodes

Validator Node is the term given to a full Libra node which participates in consensus [19]. At launch, Libra intends to have 100 validator nodes from a diverse selection of organisations and geographic locations [9], [16].

Figure 2.1: *The components of a Libra Validator Node [20]*



The six components in Figure 2.1 serve the following purposes:

- **Admission Control:** External interface for client queries. Newly submitted transactions and queries about the state of the ledger or an account are received here,
- **Mempool:** Contains transactions awaiting execution, updates to this component are share amongst validators,
- **Consensus:** The component responsible for executing the consensus protocol, see 2.2,
- **Virtual Machine:** Responsible for speculatively executing the transaction scripts to validate them see 2.1.4
- **Execution:** Responsible for executing transactions committed to the blockchain,

- **Storage:** The ledger itself, see 2.1.3.

There are two other types of participants in the Libra network: *Full nodes* and *Clients*.

Full nodes fulfil a role similar to the Admission Control component of a validator node, by answering ledger queries from clients. Full nodes also keep local storage of the ledger [21]. At present, it is not possible to run a full node connected to the official Libra testnet.

Clients refers to any computer or wallet submitting transactions or querying the ledger state [21].

2.1.2 Accounts & Wallets

The term account and wallets are used interchangeably in Libra. Accounts are simply an address with which some Move Resources (see 2.1.4) are associated [20], [22].

2.1.3 Ledger

The underlying data storage method of Validator and Full nodes in the Libra network is a RocksDB database [23]. A RocksDB instance contains a versioned database of transactions. Each version is a single transaction, identified by a monotonically increasing unsigned 64-bit integer starting from the Genesis: version 0 [19], [20], [22]. Versions are stored in a Merkle Tree, with each version existing as its own leaf node [22].

2.1.4 Move Language

Move is the programming language in which Libra transactions and (eventually) smart contracts are written. Move is an executable bytecode langauge - meaning that it is very fast and efficient to transport, store, and validate. *Types* in Move are called Resources. Resources cannot be copied, only moved in memory. Only the Libra Association is able to create or destory Move Resources [24].

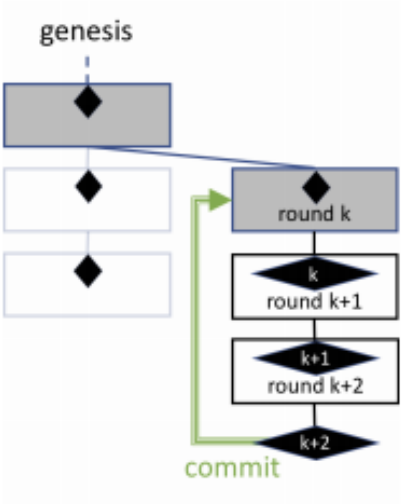
2.2 Consensus Protocol

Libra implements a leader-based Byzantine Fault Tolerant (BFT) consensus protocol called LibraBFT [22]. LibraBFT is covered in detail by an article authored by the Libra Association entitled "State Machine Replication in the Libra Blockchain" [19]. The following is a high level summary of that article. Voting is organised into *rounds*

and each round has a randomly chosen leader. The leader is sometimes referred to at the *block proposer*, as the leader proposes a block of transactions, and validators vote if that block is valid. Once *2f + 1* votes are received, the block is certified.

When a leader proposes a new block, it must extend from a certified block, or a WriteSet Transaction. A certified block k is not committed to the ledger until a 3-chain of blocks extending from k is certified. This is better illustrated in Figure 2.2.

Figure 2.2: *Proposed blocks and consensus rounds [19]*



2.3 Testnet

The Libra testnet allows for creating addresses, minting Libra coins (LBR) and making peer to peer transactions. It is also possible to deploy a local version of the Libra network.

Transactions committed to the ledger can be queried by version, or by association with a particular address. Unfortunately, that is the only information available. Since Libra is a permissioned blockchain, it is not possible to connect directly to the validator nodes, only through the specified API endpoints (section 2.3.2). So, data on how validator nodes communicate and propagate messages cannot be recorded.

2.3.1 Transaction Types

The types of transactions currently being used in the Libra testnet can be seen in [25]. there are four types:

- **BlockMetadata Transactions:** These transactions contain metadata for each block. Its return fields are detailed in 2.3.3,
- **WriteSet Transaction:** The genesis version or any updates to the validator set are WriteSet transactions.
- **User Transaction:** Any peer to peer or mint transactions. Its return fields are detailed in 2.3.3,
- **Unknown Transaction:** A blanket type for transactions (for now) that execute custom, but still valid Move scripts.

2.3.2 JSON-RPC API

The methods and return types for the JSON-RPC API can be seen here [25].

2.3.3 Types of Available Data

Fields from block metadata transaction that were used in the data collection of this project can be seen in Table 2.1. The raw version of this transaction is in Appendix B.2.1.

Table 2.1: *Relevant information returned from JSON-RPC call (Block Metadata transaction)*

Value	Type
Time Proposed	Unix Timestamp
Proposer	512 bit hex address
Transaction Type	String
Time Committed	Unix Timestamp
Transaction Version	Int
VM Status	Int

Fields from user transactions that were used in the data collection of this project can be seen in Table 2.2. The raw version of this transaction is in Appendix B.2.2.

2.3.4 Previous Revisions

The testnet originally used gRPC [26] for transporting data to clients making queries on the testnet. gRPC is a remote procedure call framework developed by Google which allows for language-agnosting types to be defined (called protobufs) and built

Table 2.2: *Relevant information returned from JSON-RPC call (User transaction)*

Value	Type
Time Proposed	Unix Timestamp
Sender	512 bit hex address
Receiver	512 bit hex address
Amount	Int
Gas Amount	Int
Transaction Type	String
Transaction Version	Int
VM Status	Int

in the end user's language of choice [27]. In May 2020, Libra decided to move towards JSON-RPC [28]. This was done in order to make Libra more easily accessible to developers by using a remote procedure call framework more popular in the cryptocurrency community.

Chapter 3

Related Works

3.1 Investigation into Other Blockchains

The following sections (3.1.1, 3.1.2, and 3.1.3) were taken and revised from the project proposal document [29].

3.1.1 Ethereum Crawler

[30] implemented a network crawler on the Ethereum network. Ethereum is a permissionless blockchain which facilitates the development of decentralised applications (dApps) through the implementation of Smart Contracts [31]. The development of dApps is also an intended features of the Libra blockchain [16].

The crawler required the bootstrapping of a full Ethereum node, communicated between peers using RPC calls and queried physical memory of the local storage to obtain information from the blockchain. The data gathered was used to visualise a number of insights from the data. Transaction throughput over time was able to be visualised with timestamps on transactions, as well as during which times the most assets were being transferred. [30] also observed 'zombie contracts' - those with no executable code in the smart contract, usually because of an error on the part of the author - and found that they increased linearly over time, implying that education surrounding verifying the correctness of a contract before deployment did not improve over time. This is an interesting discovery, considering that the Move language specification document actually quotes the difficulty in writing verifiable Ethereum Smart Contracts [24].

[30] was able to see the geographic dispersity of nodes on the Ethereum network, noting that most were centralised in western countries, Russia and China. Unfortunately, this required access to the IP addresses of nodes which isn't intended to be possible in the Libra network since it is permissioned.

3.1.2 Ripple Crawler

[32] Another network crawler was developed for the Ripple network in [32]. Ripple is another permissionless blockchain, however it was developed to be purely a payment platform. This crawler obtained information on transaction history via remote procedure calls to Ripple's public APIs and gathered the transaction history of specific addresses using Ripple's own tools for this. This information was used to build a list of all the wallets on the ledger and compute the links between them. [32] was able to classify wallets as dormant/active based on their transaction history.

[32] employed a clustering of Ripple nodes into highly connected communities and determined, unsurprisingly, that the most highly connected communities were those near 'gateway nodes', nodes that deposit Ripple currency into wallets when users deposit fiat. A clustering of public wallets was also implemented to link groups of wallets that all belonged to the same individual. In regards to wallets, implementing similar advanced analysis of transaction history on the Libra blockchain would allow computation of more insightful statistics such as the number of active or the number of unique addresses on the network.

3.1.3 HyperLedger Fabric Crawler

In [33], a network analysis was conducted on Hyperledger Fabric. Hyperledger is a Blockchain-as-a-Service (BaaS) platform which provides permissioned blockchain solutions for businesses [ref]. [33] deployed a new, permissioned blockchain on the Hyperledger network and ran 10 nodes locally on two separate computers, varying the 'workload' (amount transactions submitted to the network). The goal of this project was to assess the claims of Hyperledger is relation to execution time, latency and throughput. The experiment was repeated with 20 total nodes and higher workloads in order to assess the scalability of Hyperledger Fabric. The investigation verified that Hyperledger Fabric performed at its advertised levels and was consistent despite varying amounts of nodes and submitted transactions.

3.2 Explorers on the Libra Testnet

There are three active explorers for the Libra Testnet: Librabrowser.io, MoveOnLibra and LibExplorer.

3.2.1 LibraBrowser

Librabrowser.io [34] allows users to see recent transactions, search for transactions by address or version. For Block Metadata transactions Librabrowser.io displays on

the version number. For User Transactions, they display the version number, status, transaction type, sender, receiver, amount, expiration time, sequence number, gas used, maximum gas amount, public key, signature and script hash. The expose their own API, which allows developers to request information similar to that which can be queried directly from the testnet. However, they only offer information on user transactions and do not appear to store any data relating to block metadata transactions. Librabrowser.io does not display any statistics or analysis of the transactions on the blockchain. The developers behind this site used to have a project for this explorer open sourced and on GitHub but it has been removed.

3.2.2 MoveOnLibra

MoveOnLibra is the only open source development of a Libra explorer [35]. Similar to Librabrowser.io, they do not display any statistics related to engagement on the Libra testnet, however, they do not store any of the data from the testnet - instead make queries in real time to serve user requests [36]. This also makes it the only real-time explorer. MoveOnLibra also exposes their own API endpoints. The code for the MoveOnLibra Explorer can be seen at [36]. Their ability to query the testnet is based on open source projects by GitHub user 'yuan-yx' [37], [38].

3.2.3 LibExplorer

LibExplorer [39] has all the functionality of Librabrowser and MoveOnLibra, however it also displays some statistics relating to testnet engagement. The following statistics can be seen on LibExplorer, related to the current testnet revision:

- Latest version,
- Total number of transactions,
- Average number of transactions per second (TPS),
- Proportion of P2P to Mint transactions,
- Total supply of minted Libra coins,
- Total number of unique addresses.

The following statistics can be seen on a 24 hour and 72 hour timeframe, however do not appear to be functional as of the testnet's introduction of custom transaction types in May 2020:

- Number of Transactions,

- Address growth,
- Libra coin supply,
- Transaction fees,
- Average gas fee,
- TPS.

3.3 Literature on Libra

Currently, there are no technical articles regarding Libra. There are however, two papers which look at Libra from an economics perspective [10], [13].

Chapter 4

Data Collection Methodology

Due to the nature of open source development and Libra being in its infancy, the methodology for data collection was constantly evolving. Code conducting data collection often needed to be rewritten to work with the latest non-backwards compatible, breaking change. Further, data analysis and local storage needed to be rewritten to account for any changes to the data available. Information on how to deploy and run the code assoicated with this project can be seen in Appendix A.

Over the course of the project, the evolution of gathering data from the Libra Testnet went through five main phases:

1. gRPC & protobuf (June 2019 - September 2019),
2. gRPC & LCS (September 2019 - May 2020),
3. The introuction of Block Metadata (December 2019),
4. JSON-RPC and deprecation of gRPC (May 2020 - present),
5. The introduction of custom transaction types (May 2020 - present).

4.1 Querying the Testnet Ledger

Initially, the Libra testnet was only able to be queried via gRPC. This entailed downloading langauge-agnostic type definitions (called protobuf) for libra types from the Libra repository and building them in a language of choice. Python was used at this stage since its simplicitly allowed for rapid development of code to work with the constantly changing project.

On 19 August, 2019, Libra announced they would be moving away from protobuf and using Libra Canonical Serialisation (LCS) for storing and serving client queries of the testnet [40].

As mentioned previously, the gRPC API was deprecated on 11 June 2020 in favour of JSON-RPC [28]. In order to be able to submit working code as a part of this project, a new interface with the Libra Testnet had to be written.

The JSON-RPC API proved to be much simpler to work with, allowing the batching of queries for up to 1 million transactions at a time to be returned directly in JSON. This removed the necessiy to convert binary into JSON types using LCS.

4.2 Local Data Storage

The downloaded ledger data was converted into JSON for easy storage and use later. Initially, local storage space was negligible since there were few transactions being executed on the testnet. Following the introduction of Block Metadata transactions, a new method had to be used as one day's worth of testnet data was approximately 600MB

At this point, a NoSQL database was chosen for the storage. This was decided because Libra itself uses a NoSQL database to store its ledger on disk - RocksDB [23]. RocksDB is an open source, key-value (NoSQL) database also developed and maintained by Facebook [41]. Further, NoSQL databases allow data to be organised in a less structured way, which makes development easier when the data models are constantly changing [42], [43]. Because of these two factors, it was decided that NoSQL was more appropriate that SQL for this project.

User transactions were still stored in persistent JSON on disk, as they number of these transactions was still few enough that the memory needed to store them was negligible. Block metadata transactions were not stored in memory, but were parsed to extract the block proposer and previous block voters, update statistics in the database with this information and then discarded.

4.3 Web Application

A *simple* web application was developed in React to display this information, and act as a proof-of-concept for real-time connectivity to the Libra testnet. The application is able to query recently committed transactions in real time and display statistics which are calculated and stored in the database.

Chapter 5

Results and Discussion

5.1 Transaction Data

The data presented in this section is from 28 May 2020 to 18 June 2020. There were two resets of the testnet during this time, as such, the data comes from 3 testnet revisions.

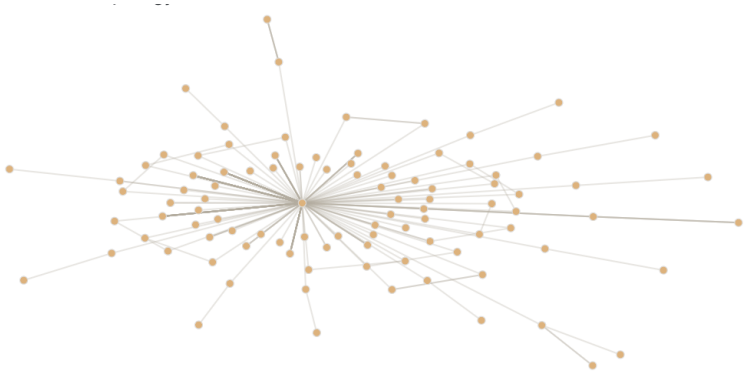
- Testnet Sample 1 (28/05/2020 - 02/06/2020) - which contains 27 million transactions,
- Testnet Sample 2 (03/06/2020 - 09/06/2020) - which contains 25 million transactions,
- Testnet Sample 3 (10/06/2020 - 18/06/2020) - which contains 35 million transactions.

Table 5.1: *Transaction data from Testnet Sample 128/05/2020 - 02/06/2020*

Total Volume LBR Minted	1,821,961,010
Total LBR Transaction Volume	15,020,134,142
Total Number Unique Addresses	62
Total Number User Transactions	125
Total Number P2P Transactions	82
Total Number Mint Transactions	43
Total Number Custom Transactions	171

Figure 5.1 shows the topology of addresses on the Libra network from this testnet revision (custom transactions removed). All coins on the network come from a single source, with very little peer to peer transactions being seen. Note: many addresses

Figure 5.1: *Address topology from Testnet Sample 1 28/05/2020 - 02/06/2020*



sent their LBR back to the minting address in this testnet, which accounts for the still large number of P2P transactions.

Table 5.2: *Transaction data from Testnet Sample 2 03/06/2020 - 09/06/2020*

Total Volume LBR Minted	949,202,440
Total LBR Transaction Volume	2,170,657,610
Total Number Unique Addresses	43
Total Number User Transactions	81
Total Number P2P Transactions	17
Total Number Mint Transactions	64
Total Number Custom Transactions	90

Table 5.3: *Transaction data from Testnet Sample 3 10/06/2020 - 18/06/2020*

Total Volume LBR Minted	671,391,071
Total LBR Transaction Volume	15,020,134,142
Total Number Unique Addresses	159
Total Number User Transactions	190
Total Number P2P Transactions	57
Total Number Mint Transactions	133
Total Number Custom Transactions	322

5.2 Voting Data

There are three sets of data presented in this section. One from prior to the deprecation of the gRPC API and two from afterwards.

The first data set contain information about the number of times each validator was selected as a leader, and the number of votes that validator was able to cast. The latter data sets only contain information about the number of times each validator was selected as leader, since the JSON-RPC API does not contain information about individual validator votes.

Note that the validator addresses have been shortened to their first four digits for simplicity. The full addresses can be seen in Appendix B.1.1, B.1.2 and B.1.3, respectively.

5.2.1 Testnet Sample 1: 28/05/2020 - 02/06/2020

The raw validator voting data from this version of the testnet can be seen in Table 5.4. From Figure 5.2, it is apparent that all validators except *65fe* were selected as round leader (block proposer) an approximately equal number of times. Further, Figure 5.3 shows that validator *65fe* also had the fewest number of votes counted during consensus.

Table 5.4: *Validator voting data from Testnet 28/05/2020 - 02/06/2020*

Validator Address	Number of Blocks Proposed	Number of Votes Counted
0e5c...	4,659,743	26,406,710
4c9e...	4,725,392	26,561,253
65fe...	3,718,246	19,473,768
b1aa...	4,711,154	26,497,856
dbd8...	4,703,143	26,179,222
fe5d...	4,481,968	24,801,950

5.2.2 Testnet Sample 2: 03/06/2020 - 09/06/2020

The raw validator voting data from this version of the testnet can be seen in Table 5.5. From Figure 5.4, it is apparent that all validators except *987e* were selected as round leader (block proposer) an approximately equal number of times.

Figure 5.2: *Pie chart displaying block proposer proportions from Testnet 28/05/2020 - 02/06/2020*

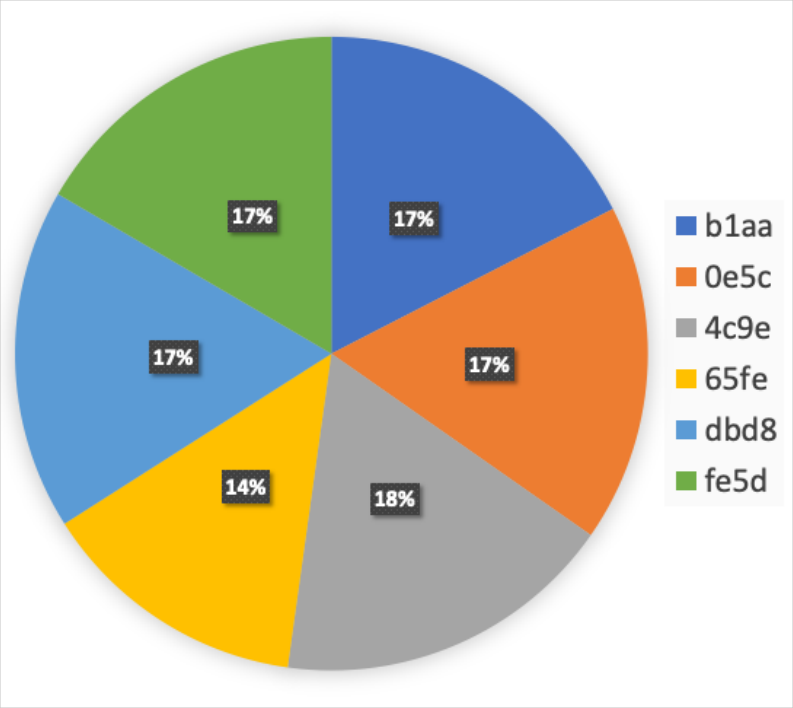


Table 5.5: *Validator voting data from Testnet 03/06/2020 - 09/06/2020*

Validator Address	Number of Blocks Proposed
987e...	2,690,592
7bbc...	4,267,926
65fe...	4,263,774
b1aa...	4,169,442
2d4c...	4,346,877
4500...	4,259,389

5.2.3 Testnet Sample 3: 10/06/2020 - 18/06/2020

The raw validator voting data from this version of the testnet can be seen in Table 5.6. From Figure 5.5, it is apparent that all validators except *65fe* were selected as round leader (block proposer) an approximately equal number of times.

Figure 5.3: *Number of votes counted per validator from Testnet 28/05/2020 - 02/06/2020*

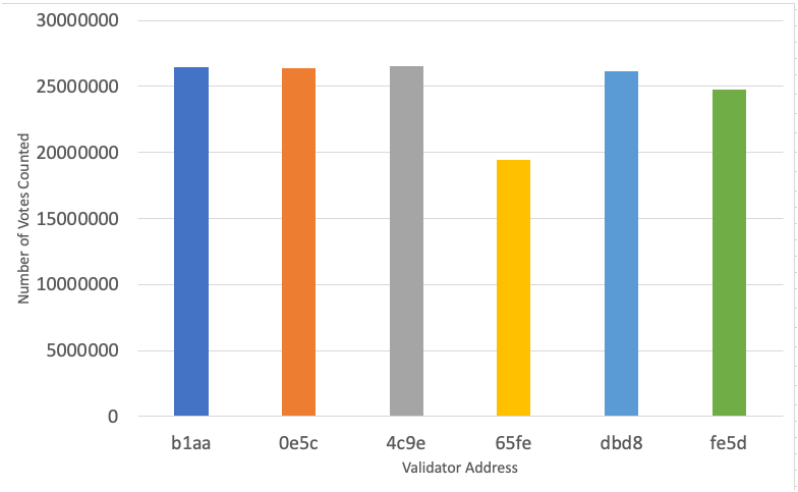


Table 5.6: *Validator voting data from Testnet 10/06/2020 - 18/06/2020*

Validator Address	Number of Blocks Proposed
987e...	5,869,035
7bbc...	5,916,089
65fe...	5,097,108
b1aa...	5,987,350
2d4c...	6,076,875
4500...	6,053,543

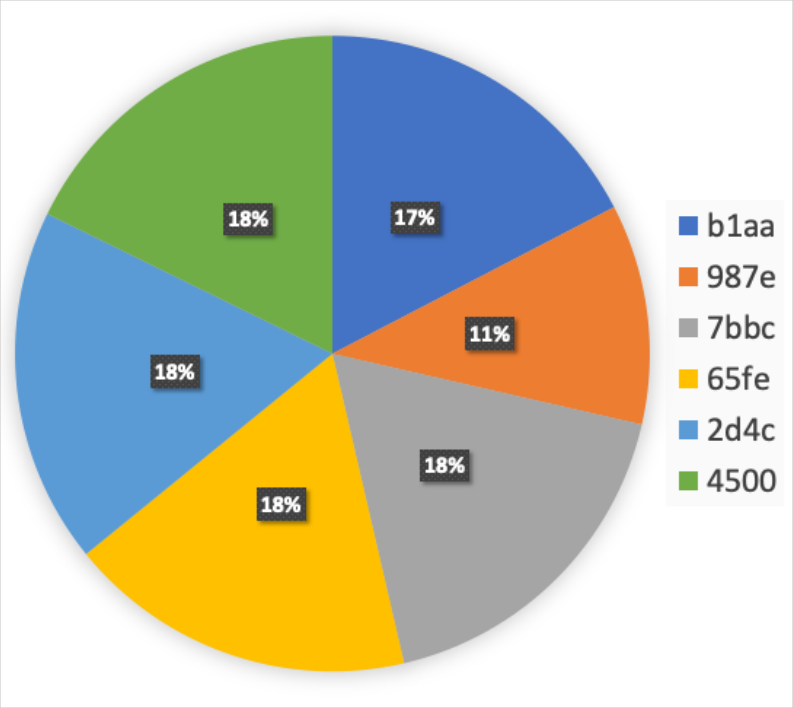
5.3 Discussion

There is little interesting information that can be taken from transaction data on the testnet, since the data corresponds to pretend currency. Because of this, the discussion around this data will focus on the validator data.

In every analysed revision of the testnet it is seen that one validator is nominated as block leader a significantly fewer amount of times than the rest. In order to understand why this is, let's explore the consensus protocol further.

Since LibraBFT is a Byztantine Fault-Tolerant protocol, *2f + 1* validators - a *greater than* two-thirds majority - must be recorded in order to certify a block. When there are six validators, five votes must be recorded. Once the five votes are reached then the sixth is not needed. Assuming that all of the testnet validators are behaving honestly, then the validator whose vote is counted the least number of

Figure 5.4: *Pie chart displaying block proposer proportions from Testnet 03/06/2020 - 09/06/2020*



times is determined to be the *least connected* to the rest of the network.

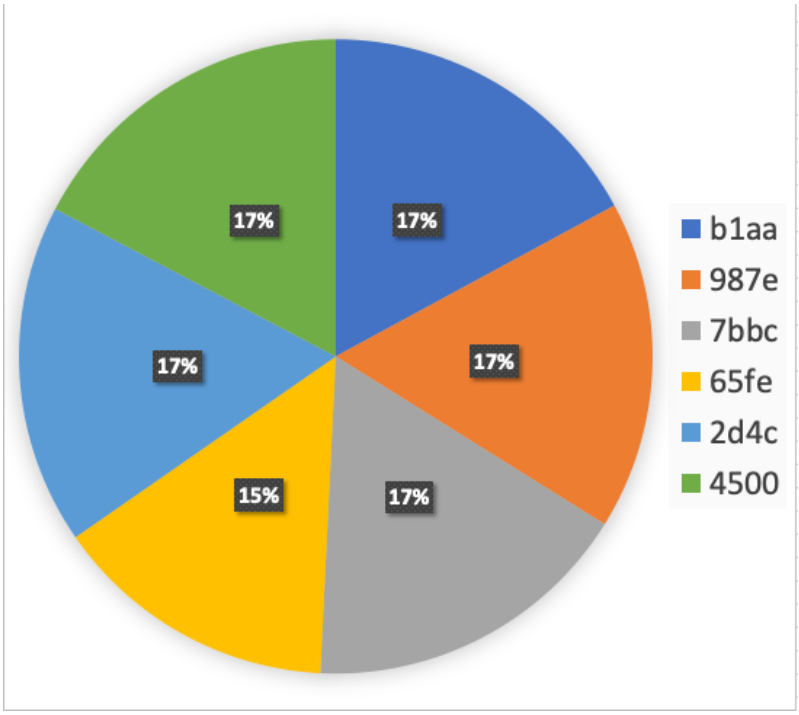
According to [19], [22], the time at which validators send messages is determined by the order in which they receive and parse other messages. The time at which validators receive messages is determined by the time in which they are sent and the transmission latency between two validators. This would explain why in Testnet 1, validator *65fe* is able to cast the fewest number of votes. This would not explain why validator *65fe* is chosen as block leader a fewer number of times.

According to [19], block leaders are chosen randomly, but the data recorded from all three Testnet versions contradict this. Further investigation of the source code reveals [44], a rust module for selecting the round leader based on reputation. The leader selection probabilities are weighted by the number of times that leader has cast a valid vote.

According to the developers, the validator nodes are currently being run off of AWS instances in the same city. Given this knowledge, it is easy to theorise that validator nodes will be penalised by leading fewer rounds of consensus, based on their average distant (and thus transmission latency) to other validators within the network.

Furthermore, if $2f + 1$ validators are all within close proximity - e.g. in the USA or being hosted on cloud services in the USA - it probable that one third of the

Figure 5.5: *Pie chart displaying block proposer proportions from Testnet 10/06/2020 - 18/06/2020*



same validators will not even participate in consensus whilst the rest of the network is acting honestly.

5.3.1 Proposed Resolutions

Two possible resolutions are proposed:

1. Adjust the voting system to allow *for* and *against* votes. Wait for a response from all validators before certifying the block and use this information in the reputation function.

2. Remove the reputation function entirely and use a purely random selection of block leader.

Chapter 6

Conclusions

6.1 Summary and conclusions

In summary, a data collection system was built for transactions on the Libra testnet. The data gathered over three testnet revisions was presented in this paper. This data indicated that the selection of round leader was not random as stated by the Libra documentation, instead, was a reputation-based, weighted selection.

A scenario in which this would be potentially problematic to the fairness of the protocol was described in which two thirds of the validator nodes were geographically centralised - diminishing the reputations of those further away.

6.2 Possible future work

There are two main avenues for future work:

1. Simulate a local version of the Libra testnet with the transmission latency challenges described in section 5.3 and try to simulate an attack on the network.

2. Create a system to collect and analyse data from a Libra Full Node once the functionality to do so has been allowed by the Libra developers.

Appendix A

Project Code

This appendix refers to the submitted project code:

`exploration-libra-testnet.zip`

Please unzip the project and start with the file README.md.

Appendix B

Testnet Data

B.1 Validators

B.1.1 28/05/2020 - 02/06/2020

0e5c92a42ce0aa7114763dcb9e4a32ae
4c9e1d0692a606cbad79d4d4550bfba4
65fe97db2d34bad62a1f2dc93b94b41a
b1aaf856811b2b15501483acbac48af8
dbd863574de2e8e2042a3c8894f548f3
fe5dafe601441ecdb9b212df494d809b

B.1.2 02/06/2020 - 09/06/2020

987e34eacaffc638c4c92f94f55dc325
7bbc44bbeda1ab0b89fe8b66dad0c295
65fe97db2d34bad62a1f2dc93b94b41a
b1aaf856811b2b15501483acbac48af8
2d4c9ad518480545f9a70b30e0389694
450092009c245134186ff91aa49e159c

B.1.3 10/06/2020 - 18/06/2020

987e34eacaffc638c4c92f94f55dc325
7bbc44bbeda1ab0b89fe8b66dad0c295
65fe97db2d34bad62a1f2dc93b94b41a
b1aaf856811b2b15501483acbac48af8

2d4c9ad518480545f9a70b30e0389694
450092009c245134186ff91aa49e159c

B.2 JSON-RPC Return Objects

B.2.1 Block Metadata Transaction

A JSON representation of a Block Metadata object, as returned by the JSON-RPC endpoint can be seen in Figure B.1 on page 27.

Figure B.1: *JSON representation of JSON-RPC Block Metadata object*


```
"events": [
  {
    "data": {
      "proposed_time": 1590615427553018,
      "proposer": "65fe97db2d34bad62a1f2dc93b94b41a",
      "round": 7567,
      "type": "newblock"
    },
    "key": "1000000000000000000000000000000000000000a550c18",
    "sequence_number": 7564,
    "transaction_version": 7565
  }
],
"gas_used": 0,
"transaction": {
  "timestamp_usecs": 1590615427553018,
  "type": "blockmetadata"
},
"version": 7565,
"vm_status": 4001
```


B.2.2 Peer to Peer User Transaction

A JSON representation of a User Transaction object, as returned by the JSON-RPC endpoint can be seen in Figure B.2 on page 28.

Figure B.2: *JSON representation of JSON-RPC User Transaction object*


```
"events": [...],
"gas_used": 0,
"transaction": {
  "expiration_time": 1591016968,
  "gas_unit_price": 0,
  "max_gas_amount": 1000000,
  "public_key": "f3a3cde7c4be6e24eabb8400d9a37b4dd11190d1c0...",
  "script": {
    "amount": 10000000,
    "auth_key_prefix": "4091d307671a741984e2cf4e7ab40c2c",
    "metadata": "",
    "metadata_signature": "",
    "receiver": "d1d4bd5ded3d4b3ea86ad69fc99c19d7",
    "type": "peer_to_peer_transaction"
  },
  "script_hash": "c8bc3dda60e9662965b3223c22e3d3e3e7b6f698c...",
  "sender": "e58234cd43e27732f8d79c49a6ba02e0",
  "sequence_number": 1,
  "signature": "4aa75f2535b18cabcb5d26fcbca1af03bdc1d632233...",
  "signature_scheme": "Scheme::Ed25519",
  "type": "user"
},
"version": 26909472,
"vm_status": 4001
```


B.3 gRPC Return Objects

B.3.1 Block Metadata Transaction

A JSON representation of a Block Metadata object, as returned by the gRPC endpoint can be seen in Figure B.3 on page 29.

B.3.2 Peer to Peer User Transaction

A JSON representation of a User Transaction object, as returned by the gRPC endpoint can be seen in Figure B.4 on page 30.

Figure B.3: *JSON representation of gRPC Block Metadata Transaction object*


```
"id": "638ae382b34ac8321404b36cfb0c0669c44684b772fe0f87...",
"round": 10,
"timestamp_usecs": 1590021963409094,
"previous_block_votes": [
  "0e5c92a42ce0aa7114763dcb9e4a32ae",
  "65fe97db2d34bad62a1f2dc93b94b41a",
  "b1aaf856811b2b15501483acbac48af8",
  "dbd863574de2e8e2042a3c8894f548f3",
  "fe5dafe601441ecdb9b212df494d809b"
],
"proposer": "0e5c92a42ce0aa7114763dcb9e4a32ae",
"transaction_info": {
  "transaction_hash": "64832ece49809eb9081530c5d16819265...",
  "state_root_hash": "124e2dabcf1ca65297c83a68d06f26b9d2...",
  "event_root_hash": "d267da7c8a3cb79cf3c46e14eb55ff8fb7...",
  "gas_used": 0,
  "major_status": 4001
}
```

Figure B.4: *JSON representation of gRPC User Transaction object*


```
"raw_txn": {
  "sender": "00000000000000000000000a550c18",
  "sequence_number": 845,
  "payload": {
    "Script": {
      "code": "a11ceb0b010007014600000002000000000000...",
    }
  },
  "max_gas_amount": 1000000,
  "gas_unit_price": 0,
  "gas_currency_code": "LBR",
  "expiration_time": 1590554690
},
"authenticator": {
  "public_key": "c56b384d31b4e2e66dc0b78de7ce537e07...",
  "signature": "b39fa94ed5472920eee820370dce87b6e7a..."
},
"transaction_info": {
  "transaction_hash": "90e5daf7e117ca41e5641f025b32cc...",
  "state_root_hash": "a7b173346172324b4e52b3a7677bf25...",
  "event_root_hash": "e2b3c5612f7245ab6da9fa10f1f06d4...",
  "gas_used": 0,
  "major_status": 4001
}
```

Bibliography

[1] J. Johnson, "Bitcoin and venezuela's unofficial exchange rate," eng, *Ledger (Pittsburgh, Pa.)*, vol. 4, 2019, ISSN: 2379-5980. [Online]. Available: https://doaj.org/article/3f541f8436c845ef8fd926cceb45b87b.

[2] R. Sharma, *Hyperinflation produces surge in bitcoin trading in venezuela*, Available: https://www.investopedia.com/news/hyperinflation-produces-surge-bitcoin-trading-venezuela/. [Accessed: 01-June-2020].

[3] "Louis vuitton and microsoft are building a blockchain project," *ICT Monitor Worldwide*, 2019.

[4] S. Angraal M., H. Krumholz L., and W. Schulz L., "Blockchain technology: Applications in health care," *Circulation: Cardiovascular Quality and Outcomes*, vol. 10, no. 9, e003800–e003800, 2017, ISSN: 1941-7713.

[5] L. Mertz, "(block) chain reaction: A blockchain revolution sweeps into health care, offering the possibility for a much-needed data solution," eng, *IEEE Pulse*, vol. 9, no. 3, pp. 4–7, 2018, ISSN: 2154-2287.

[6] K. W. Prewett, G. L. Prescott, and K. Phillips, "Blockchain adoption is inevitable—barriers and risks remain," *Journal of Corporate Accounting & Finance*, vol. 31, no. 2, pp. 21–28, 2020, ISSN: 1044-8136.

[7] M. Isaac and N. Popper, *Facebook plans global financial system based on cryptocurrency*, 2019. [Online]. Available: https://www.nytimes.com/2019/06/18/technology/facebook-cryptocurrency-libra.html?action=click&module=Top%20Stories&pgtype=Homepage.

[8] Libra Association, *The libra project is for the world*, Available: https://libra.org/en-US/vision/. [Accessed: 06-June-2020].

[9] ——, *Libra white paper*, Available: https://libra.org/en-US/white-paper/. [Accessed: 01-May-2020].

[10] R. Rirsch and S. Tomanek, "Facebook's libra: A case for capital markets supervision?" eng, *Journal of Payments Strategy & Systems*, vol. 13, no. 3, pp. 255–267, 2019, ISSN: 1750-1806.

[11] K. Petrou, "The threat posed by facebook's libra," eng, *American Banker Magazine*, vol. 129, no. 8, pp. 20–20, 2019, ISSN: 21623198. [Online]. Available: http://search.proquest.com/docview/2272758186/.

[12] J. Clement, *Number of monthly active facebook users worldwide as of 1st quarter 2020*, Available: https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/. [Accessed: 17-June-2020].

[13] J. Taskinsoy, "This time is different: Facebook's libra can improve both financial inclusion and global financial stability as a viable alternative currency to the u.s. dollar," eng, *Journal of Accounting, Finance and Auditing Studies*, vol. 5, no. 4, pp. 67–86, 2019, ISSN: 21490996. [Online]. Available: http://search.proquest.com/docview/2305779319/.

[14] *The EtherScan project*, Available: https://etherscan.io/. [Accessed: 6-July-2019].

[15] *The Blockchain.com Explorer project*, Available: https://www.blockchain.com/explorer. [Accessed: 06-July-2019].

[16] Libra Engineering Team, *Libra: The path forward*, Available: https://libra.org/en-US/blog/the-path-forward/. [Accessed: 02-July-2019].

[17] N. Statt, *Facebook is shifting its libra cryptourrency plans after intense regulatory pressure*, 2020. [Online]. Available: https://www.theverge.com/2020/3/3/21163658/facebook-libra-cryptocurrency-token-ditching-plans-calibra-wallet-delay.

[18] Libra Association, *How to Become a Founding Member*, Available: https://libra.org/en-US/wp-content/uploads/.sites/23/2019/06/HowtoBecomeaFoundingMember_en_US-2.pdf. [Accessed: 01-July-2019].

[19] The LibraBFT Team, "State Machine Replication in the Libra Blockchain," 2020.

[20] Libra Association, *Libra protocol: Key concepts*, Available: https://developers.libra.org/docs/libra-protocol. [Accessed: 03-June-2020].

[21] Libra Engineering Team, *Full node basics: An introduction to full nodes in the libra network*, Available: https://libra.org/es-LA/blog/full-node-basics/. [Accessed: 05-Feb-2020].

[22] Z. Amsden et al., "The Libra Blockchain," 2019.

[23] Libra Association, *Libra Codebase - Storage*, Available: https://developers.libra.org/docs/crates/storage. [Accessed: 04-June-2020].

[24] S. Blackshear et al., "Move: A Language With Programmable Resources," 2019.

[25] Libra Association, *JSON-RPC Specification*, Available: `https://github.com/libra/libra/blob/master/json-rpc/json-rpc-spec.md`. [Accessed: 15-May-2020].

[26] *The gRPC project*, Available: `https://grpc.io/about/`. [Accessed: 04-July-2019].

[27] K. Sandoval, *When to Use What: REST, GraphQL, Webhooks, & gRPC*, Available: `https://nordicapis.com/when-to-use-what-rest-graphql-webhooks-grpc/`. [Accessed: 05-Feb-2020].

[28] Libra Association, *New JSON-RPC Libra client API: The gRPC API is being deprecated*, Available: `https://libra.org/en-US/blog/new-json-rpc/`. [Accessed: 15-May-2020].

[29] Cameron Harder-Hutton, "Thesis Project Proposal: Technical Analysis of the Libra Blockchain," 2019.

[30] L. Anderson, R. Holz, A. Ponomarev, P. Rimba, and I. Weber, "New kids on the block: An analysis of modern blockchains," eng, *arXiv.org*, 2016, ISSN: 2331-8422. [Online]. Available: `http://search.proquest.com/docview/2079615829/`.

[31] V. Buterin, *Ethereum Whitepaper*, Available: `https://ethereum.org/whitepaper/`. [Accessed: 02-July-2019].

[32] M.-S. Pedro, Z. M. Bilal, and K. Aniket, "Listening to whispers of ripple: Linking wallets and deanonymizing transactions in the ripple network," eng, *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 4, pp. 436–453, 2016, ISSN: 2299-0984. [Online]. Available: `https://doaj.org/article/952f0dd39df94a01aa141696d26f08c3`.

[33] Q. Nasir, I. A. Qasse, M. Abu Talib, and A. B. Nassif, "Performance analysis of hyperledger fabric platforms," *Security and Communication Networks*, vol. 2018, 2018, ISSN: 1939-0114.

[34] *The LibraBrowser project*, Available: `https://librabrowser.io/`. [Accessed: 06-July-2019].

[35] *The MoveOnLibra project*, Available: `https://explorer.moveonlibra.com/`. [Accessed: 06-July-2019].

[36] *The MOL LibraExplorer project*, Available: `https://github.com/MoveOnLibra/LibraExplorer`. [Accessed: 04-June-2020].

[37] *The LibraClient project*, Available: `https://github.com/yuan-xy/libra-client`. [Accessed: 04-June-2020].

[38] *The LibraClient gRPC project*, Available: `https://github.com/yuan-xy/libra-client-grpc`. [Accessed: 04-June-2020].

[39] *The LibExporer project*, Available: `https://libexplorer.com/`. [Accessed: 04-June-2020].

[40] D. Wolinsky, *Moving Toward RawTransactions Serialized in Canonical Format*, Available: `https://community.libra.org/t/moving-toward-rawtransactions-serialized-in-canonical-format/1561`. [Accessed: 21-Sep-2019].

[41] *The RocksDB project*, Available: `https://rocksdb.org/`. [Accessed: 04-June-2020].

[42] M. Stonebraker, "Sql databases v. nosql databases," eng, *Communications of the ACM*, vol. 53, no. 4, pp. 10–11, 2010, ISSN: 00010782.

[43] MongoDB Inc, *Advantages of NoSQL*, Available: `https://www.mongodb.com/scale/advantages-of-nosql`. [Accessed: 04-June-2020].

[44] Libra Assocation, "The libra association,"