Blockchain-based Collaborative Database for a consortium of Fintech companies

Siddharth Singh

Department of Computer Science and Engineering
Indian Institute of Technology Madras

June 6, 2023

Introduction

- In the Fintech industry, companies have to store the KYC data of their users for regulatory compliance.
- KYC data includes details such as Aadhar Number, scan of the Aadhar card, scan of the Passport, Driving License, etc.
- Often, the customers are affiliated with multiple Fintech companies.
- Traditionally, each company maintains the KYC information of its customers in its own databases.
- However, this approach leads to many disadvantages.
  1. **Pain for the customer** - The customer has to do his/her KYC for each company separately that he/she wants to enrol in.
  2. **Pain for the company** - The company has to store the KYC data again (Data Redundancy)

Expected scene

- The companies form a consortium and maintain a shared database.
- It would be easier for the customers as they do not have to redo their KYC if they are applying for a company which is a part of the consortium.
- Similarly, it will be more efficient for the companies as it would reduce their storage costs.
- It could be turned into a business model as big companies will likely have a lot of customers, and they like to take some incentive by providing their data to smaller members of the consortium.
- Similarly, smaller companies would like to join the consortium to get easy access to KYC data from the bigger members.

Requirements from the database

- Updates need to be based on a consensus mechanism
- It needs to give a good throughput
- It needs to be scalable
- It needs to have support for smart contracts
- It needs to be using an efficient consensus mechanism

CLT vs. DLT

- The database can be built on two types of technologies - Centralized Ledger Technology (CLT) or Distributed Ledger Technology (DLT).
- DLTs use the blockchain for storing data, while CLDs typically do not use a blockchain data structure.
- In CLDs, a trusted third party maintains and controls the ledger and does not require a consensus mechanism.
- DLT-based database systems are inherently safer than traditionally distributed databases against malicious attacks.
- The performance and transaction throughput of DLT-based databases is lower than conventional distributed databases or centralised ledger technologies.
- This limitation paved the way for developing a centralized technique that can match the performance of traditional distributed databases.

Options available

Name	LedgerDB
Database type	key-value
Transaction processing	Execute-commit-index
Smart contract supported	not supported
Transactions per second	100k

Name	Blockchain Relational Database	BigchainDB	FalconDB
Database type	PostgreSQL	MongoDB	MySQL and IntegriDB
Replication model	Txn-based	Txn-based	Storage-based
Transaction processing	Execute and order in parallel	Order-then-commit	Order-then-execute
Consensus mechanism	Kafka	Tendermint	Tendermint
Smart contract supported	procedures as smart contract	not supported	Underlying Blockchain required
Transactions per second	1.5k	600	2k

Name	ChainifyDB	BlockchainDB	Hyperledger Fabric
Database type	PostgreSQL and MySQL	Key-Value	CouchDB
Replication model	Txn-based	Storage-based	Txn-based
Transaction processing	Whatever Ledger Consensus	Order-then-execute	Execute-order-validate
Consensus mechanism	Kafka	Proof of Work	Raft
Smart contract supported	not supported	Underlying Blockchain required	supported
Transactions per second	1k	<100	600

FalconDB

- FalconDB is a decentralised network where nodes can function as servers or clients.
- The server nodes exclusively maintain item the storage of the entire database, whereas the clients retain only the headers of the blocks.
- To assist clients in verifying their requests sent to the server nodes, Authentication Data Structures (ADS) are securely stored within the database on the server nodes.
- By leveraging the ADS, a digest is generated to represent the current content of the database.
- This digest serves as a means for clients to authenticate the outcomes provided by the server nodes. Furthermore, each block header contains the digest corresponding to the respective state, thereby allowing clients to access the digest associated with the present database content.
- This design allows FalconDB to tolerate up to 1/3 of malicious nodes.

Authenticated Data Structures (ADS)

- Authenticated data structures (ADS) are commonly used in outsourced databases, where users upload their databases to a cloud server and access the database remotely.
- ADS empowers database clients with the ability to verify the results of queries and updates to the remote database, which could prevent the server from being malicious.
- ADS encompasses functions that facilitate querying, validation, and the generation of a digest representing the current state of the database.
- ADS emerges as a critical component within outsourced database systems, aiming to uphold the integrity, security, and efficiency of operations while effectively mitigating potential malicious attacks on server nodes.
- ADS offers a reliable way for clients to verify the validity of a response from a server node. This helps to minimize the risk of malicious activity, such as tampering with data or attempting to deceive clients.

Design

- **Server nodes:**
  - The database and complete blockchain data are maintained by server nodes operated by various entities.
  - These nodes handle client requests and updates, verify newly generated blocks and produce evidence for query results.
  - These nodes receive financial compensation in exchange for their service.
- **Client nodes:**
  - The collaborative database is supported by client nodes participating in database operations by sending requests to the server nodes.
  - They can read or write specific database portions based on their access privileges. The client nodes don't store the database locally but can authenticate queries or update responses.
  - Some client nodes validate blocks by joining the blockchain network, while others only retrieve the latest blocks without engaging in blockchain consensus.

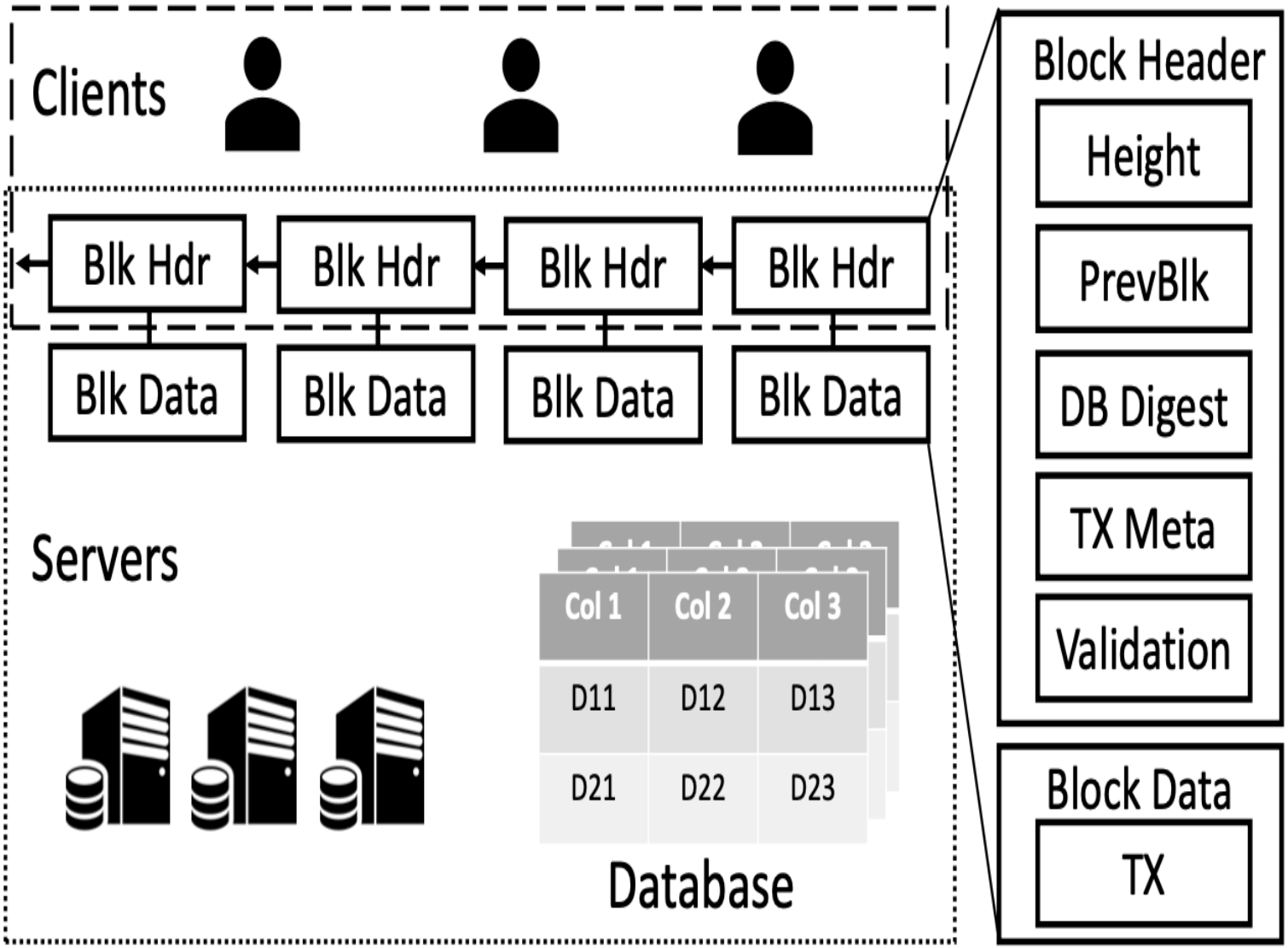


Figure: System overview

- A client node can initiate a query by transferring the query fee from its deposit to a server node and receiving the result. If the client is unsatisfied with the result, it can challenge it by requesting authentication from the smart contract.
- To do this, the client must pay the server an additional fee. The server must then generate an ADS proof that the digest can validate.
- If the server fails to provide this proof, it will lose all fees in its smart contract account, including its initial deposit and client revenue.

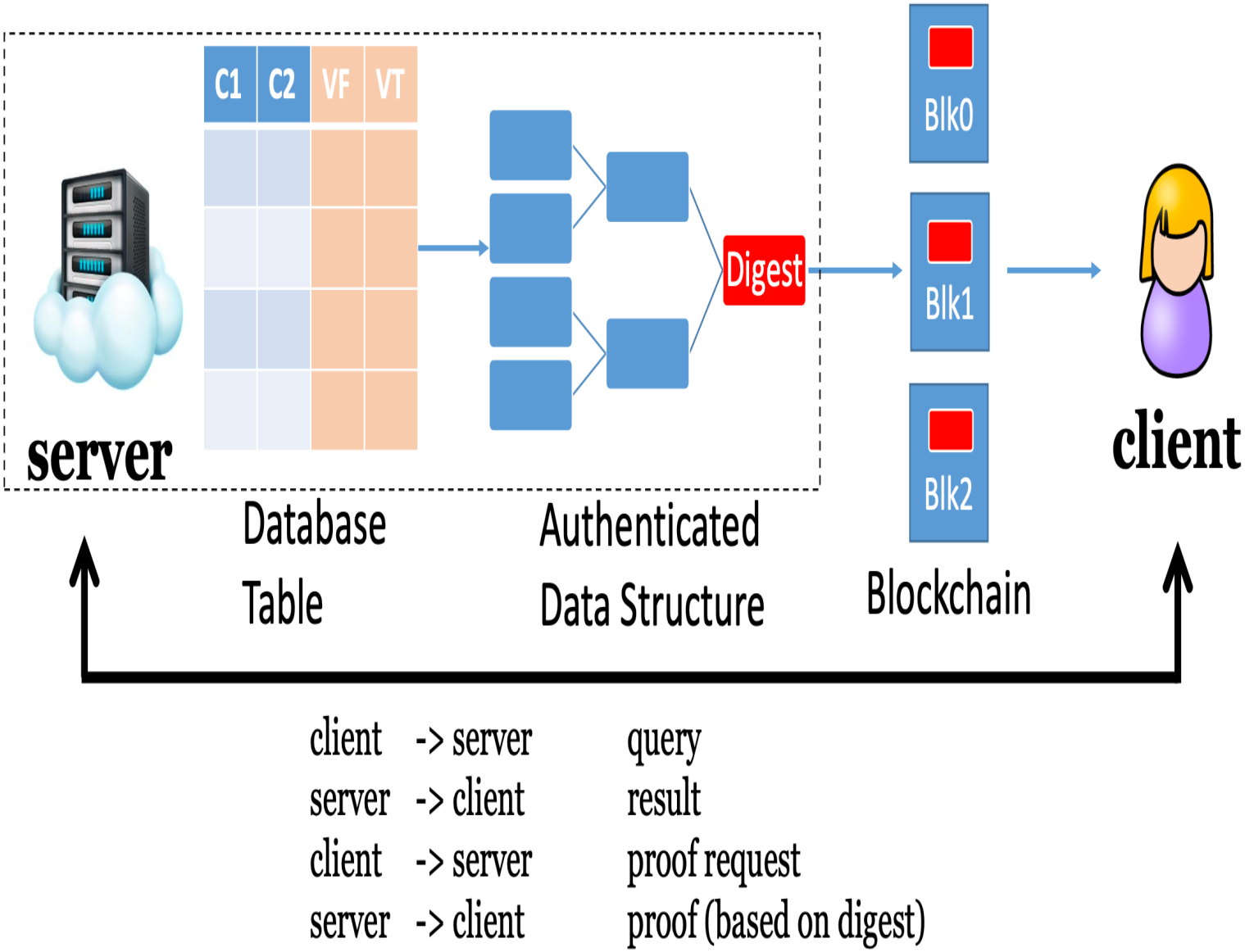


Figure: Simplified query workflow of FalconDB

Role of IPFS

- In situations like Know Your Customer (KYC) data, where scanned documents like Aadhar Cards, PAN Cards, passports, and driving licenses need to be stored, storing these large image files on the blockchain can quickly become cumbersome.
- Therefore, it is necessary to store these files on a separate network that is decentralized and only stores their hashes in the blockchain transactions and database.
- We have implemented the use of IPFS, a decentralized network that allows clients and servers to store and retrieve data, as a solution to this problem.
- With this approach, the clients can first store the scanned images on the IPFS network and then include the hashes returned by IPFS in the read or update queries submitted to the server instead of the image files themselves.

Implementation

The clients can issue three types of queries in our system to the server nodes - ADD, UPDATE, and DELETE.

1. **ADD**
   - Syntax: ADD Name AadharNo AadharHash PassportNo PassportHash
   - Semantics: Adds the entry of a user having the mentioned details in the database
2. **UPDATE**
   - Syntax: UPDATE AadharNo PassportNo PassportHash
   - Semantics: Updates the passport number and scan of the passport of the customer having the specified Aadhar Number
3. **DELETE**
   - Syntax: DELETE AadharNo
   - Semantics: Deletes the entry corresponding to the customer with the specified Aadhar Number

Workflow

- Start the IPFS node.

```
siddharthsingh@siddharths-MacBook-Pro ipfs % IPFS_PATH=~/Desktop/BTP/ipfs ipfs daemon

Initializing daemon...
Kubo version: 0.19.1
Repo version: 13
System version: amd64/darwin
Golang version: go1.19.8

Computed default go-libp2p Resource Manager limits based on:
    - 'Swarm.ResourceMgr.MaxMemory': "4.3 GB"
    - 'Swarm.ResourceMgr.MaxFileDescriptors': 5120

Theses can be inspected with 'ipfs swarm resources'.

Swarm is limited to private network of peers with the swarm key
Swarm key fingerprint: ad3b8dcdc326b6728b7566bc40f0bb8c
Swarm listening on /ip4/10.42.80.156/tcp/4001
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip6/::1/tcp/4001
Swarm listening on /p2p-circuit
Swarm announcing /ip4/10.42.80.156/tcp/4001
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip6/::1/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
WebUI: http://127.0.0.1:5001/webui
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

Figure: IPFS node gets started

- Add the scans of the Aadhar card and Passport to the (private) IPFS network. In case of an UPDATE query, you have to add the Passport scan. This step is omitted in a DELETE query,
- Above step will return the hashes of the scans uploaded.

```
siddharthsingh@siddharths-MacBook-Pro BTP % IPFS_PATH=~/Desktop/BTP/ipfs ipfs add aadhar2.jpg
added QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV aadhar2.jpg
 26.97 KiB / 26.97 KiB [=============================================================================================================================================] 100.00%
```

Figure: Aadhar scan gets added to the IPFS network

- Start the application.

```
siddharthsingh@siddharths-MacBook-Pro App % zsh ./gradlew run

> Task :run
gRPC server started, listening on $port
<===========--> 85% EXECUTING [9s]
> :run
```

Figure: The custom application gets started

- Start the Tendermint node

```
siddharthsingh@siddharths-MacBook-Pro bin % tendermint node --abci grpc --home ~/Desktop/BTP/mytestnet/node0 --proxy_app tcp://192.168.0.3:26658
I[2023-05-13|18:59:04.949] service start                                module=proxy msg="Starting multiAppConn service" impl=multiAppConn
I[2023-05-13|18:59:04.949] service start                                module=abci-client connection=query msg="Starting grpcClient service" impl=grpcClient
I[2023-05-13|18:59:04.949] Dialed server. Waiting for echo.             module=abci-client connection=query addr=tcp://192.168.0.3:26658
I[2023-05-13|18:59:05.283] service start                                module=abci-client connection=snapshot msg="Starting grpcClient service" impl=grpcClie
nt
```

Figure: Tendermint node gets started

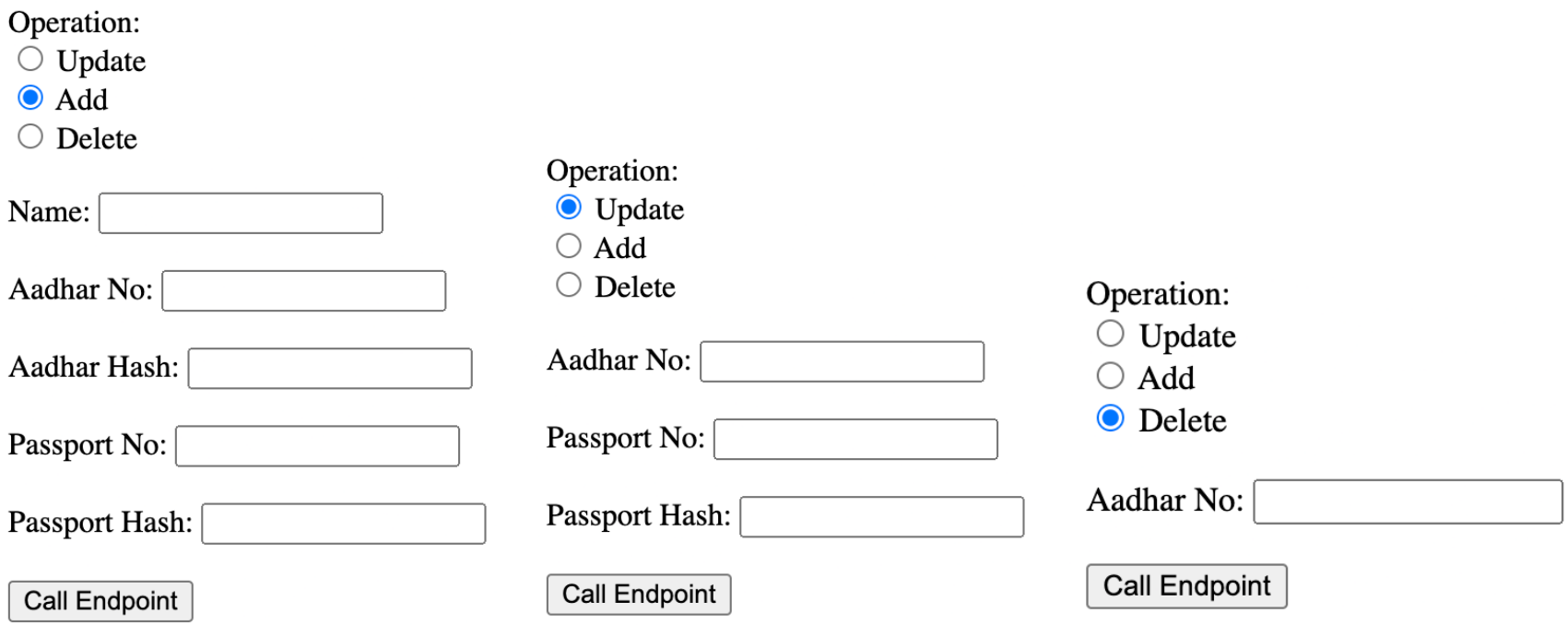
- Now pass the query to the *broadcast_tx_commit* endpoint of the tendermint node

Available endpoints:

Endpoints that require arguments:
//10.42.80.156:26657/abci_info?
//10.42.80.156:26657/abci_query?path=_&data=_&height=_&prove=_
//10.42.80.156:26657/block?height=_
//10.42.80.156:26657/block_by_hash?hash=_
//10.42.80.156:26657/block_results?height=_
//10.42.80.156:26657/block_search?query=_&page=_&per_page=_&order_by=_
//10.42.80.156:26657/blockchain?minHeight=_&maxHeight=_
//10.42.80.156:26657/broadcast_evidence?evidence=_
//10.42.80.156:26657/broadcast_tx_async?tx=_
//10.42.80.156:26657/broadcast_tx_commit?tx=_
//10.42.80.156:26657/broadcast_tx_sync?tx=_
//10.42.80.156:26657/check_tx?tx=_
//10.42.80.156:26657/commit?height=_
//10.42.80.156:26657/consensus_params?height=_
//10.42.80.156:26657/consensus_state?
//10.42.80.156:26657/dump_consensus_state?
//10.42.80.156:26657/genesis?
//10.42.80.156:26657/genesis_chunked?chunk=_
//10.42.80.156:26657/health?
//10.42.80.156:26657/net_info?
//10.42.80.156:26657/num_unconfirmed_txs?
//10.42.80.156:26657/status?
//10.42.80.156:26657/subscribe?query=_
//10.42.80.156:26657/tx?hash=_&prove=_
//10.42.80.156:26657/tx_search?query=_&prove=_&page=_&per_page=_&order_by=_
//10.42.80.156:26657/unconfirmed_txs?limit=_
//10.42.80.156:26657/unsubscribe?query=_
//10.42.80.156:26657/unsubscribe_all?
//10.42.80.156:26657/validators?height=_&page=_&per_page=_

Figure: List of all endpoints provided by Tendermint

- We also designed a frontend UI (User Interface) through which the client can issue the request to servers. Below are the snapshots of that.



(a) ADD (b) UPDATE (c) DELETE

Figure: Frontend User Interface

- Peers get added.

```
I[2023-05-13|18:59:06.058] P2P Node ID                                  module=p2p ID=102cb9d877e37e1c700514f07f0fd1aa3880f5c3 file=/Users/siddharthsingh/Desk
top/BTP/mytestnet/node0/config/node_key.json
I[2023-05-13|18:59:06.058] Adding persistent peers                      module=p2p addrs="[102cb9d877e37e1c700514f07f0fd1aa3880f5c3@192.168.0.3:26656 07c1a73c
619a4be41a2914bc7075c5e92777054b@192.168.0.4:26656]"
I[2023-05-13|18:59:06.058] Adding unconditional peer ids                module=p2p ids=[]
I[2023-05-13|18:59:06.059] Add our address to book                      module=p2p book=/Users/siddharthsingh/Desktop/BTP/mytestnet/node0/config/addrbook.json
 addr=102cb9d877e37e1c700514f07f0fd1aa3880f5c3@0.0.0.0:26656
```

Figure: Peer nodes get added

- Transaction gets committed.

```
I[2023-05-13|19:09:52.020] received proposal                            module=consensus proposal="Proposal{4/1 (CEED2F0FCCF958B25AE2F38B0C5C47099793C06ABF2E3
95691323BB9D4D83D26:1:3FDC542491DE, -1) D939EB1110F9 @ 2023-05-13T13:39:52.015188Z}"
I[2023-05-13|19:09:52.024] received complete proposal block             module=consensus height=4 hash=CEED2F0FCCF958B25AE2F38B0C5C47099793C06ABF2E395691323BB
9D4D83D26
I[2023-05-13|19:09:52.584] finalizing commit of block                   module=consensus height=4 hash=CEED2F0FCCF958B25AE2F38B0C5C47099793C06ABF2E395691323BB
9D4D83D26 root=0000000000000000 num_txs=1
I[2023-05-13|19:09:52.619] executed block                               module=state height=4 num_valid_txs=1 num_invalid_txs=0
I[2023-05-13|19:09:52.653] committed state                              module=state height=4 num_txs=1 app_hash=0000000000000000
I[2023-05-13|19:09:52.664] indexed block exents                         module=txindex height=4
```

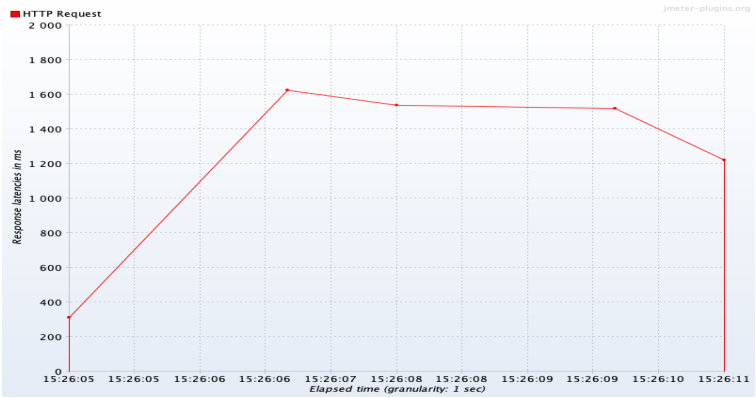
Figure: Transaction gets committed to the blockchain

- Entry gets added to the database

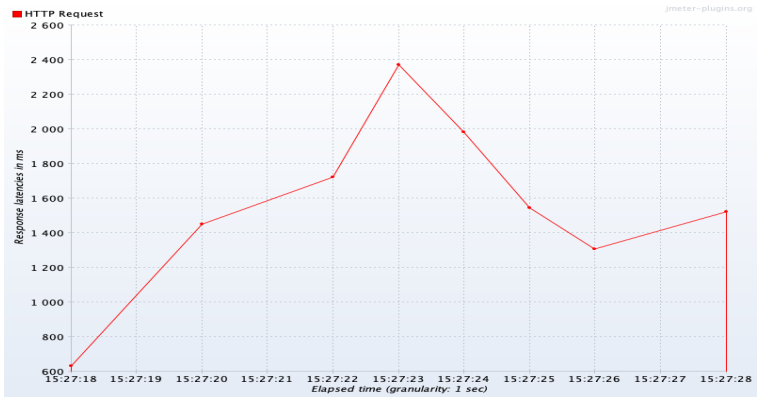
```
mysql> USE testDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from KYC;
+-----------+----------+----------------------------------------------+------------+----------------------------------------------+
| Name      | AadharNo | AadharScanHash                               | PassportNo | PassportScanHash                             |
+-----------+----------+----------------------------------------------+------------+----------------------------------------------+
| SID       | 0        | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10010143 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10012266 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10015730 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10025732 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10027781 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10032600 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10038287 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10039022 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10048279 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10053927 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10065631 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10071747 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10071964 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10071967 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10074169 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10075258 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10076284 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10088197 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10090286 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10091558 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10096364 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10099051 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10100370 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10104568 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10107213 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10111657 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10121625 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10122638 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10127947 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Sid       | 10137959 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10137985 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10138405 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10139154 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10139933 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| SID       | 10149890 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
| Siddharth | 10151160 | QmbZxVaJtikCxPjpAh6AhLAqkWaY5afFocBGYBomEE1LSV | 123456789  | Qmf4f1s5h3npE116JbTdvLJNRZcWZnyHRTf2PN1kUgfkoS |
```

Figure: Database gets updated

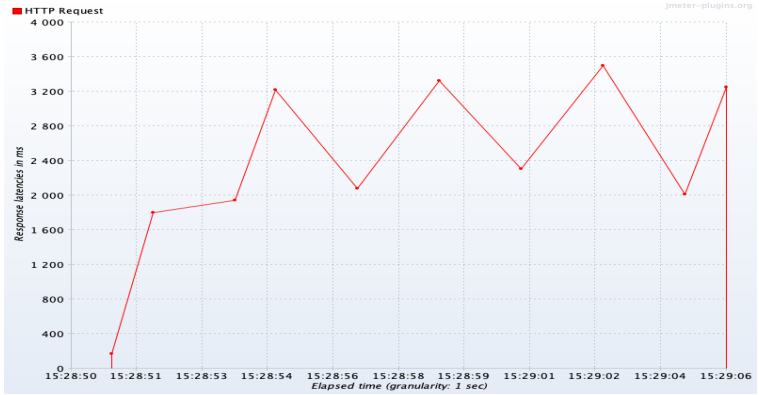
Analysis



(a) Number of threads = 1



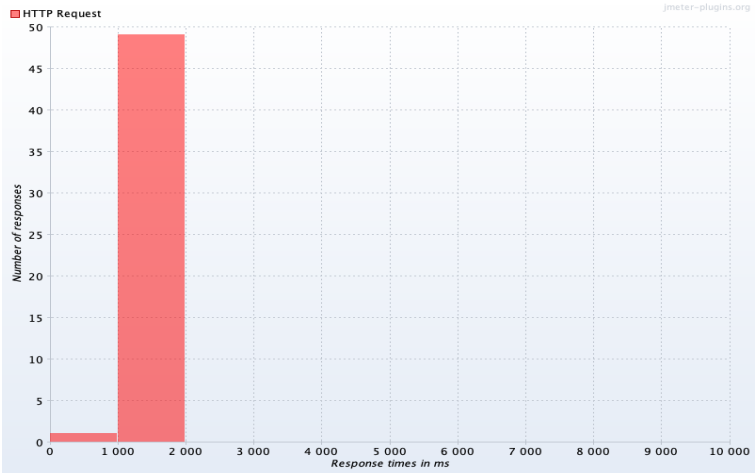
(b) Number of threads = 50



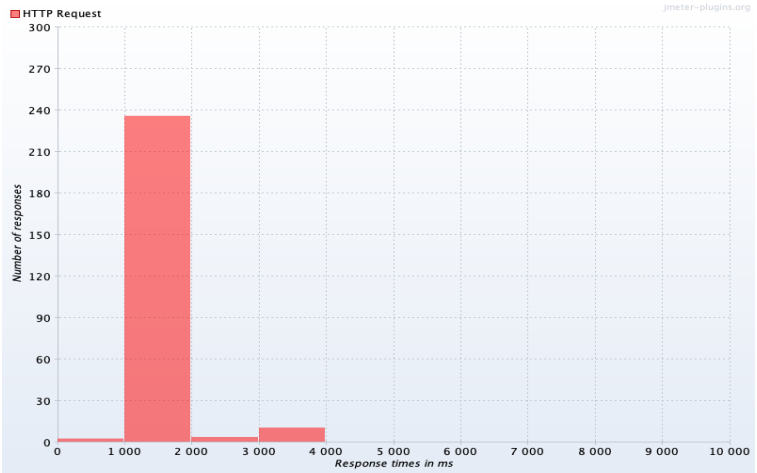
(c) Number of threads = 100

Figure: Latencies vs. Time

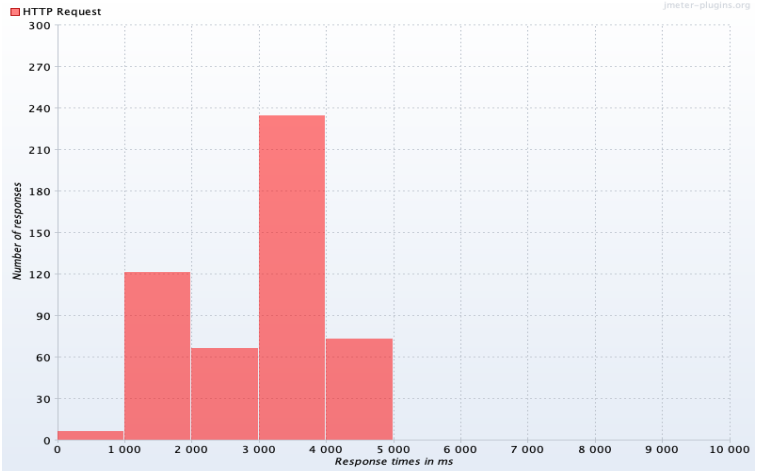
- The average latency increases with an increasing number of threads.
- This is an expected behaviour as network congestion increases with more threads, which leads to more latency in fulfilling the requests.
- Also, when the number of threads is less there is lesser variability in latency. After a time, it almost remains constant.
- However, as the number of threads keeps on increasing, the variability in latency becomes more noticeable.
- When there are 100 concurrent threads requesting the server node for a query, the latency almost starts oscillating up and down.



(a) Number of threads = 1



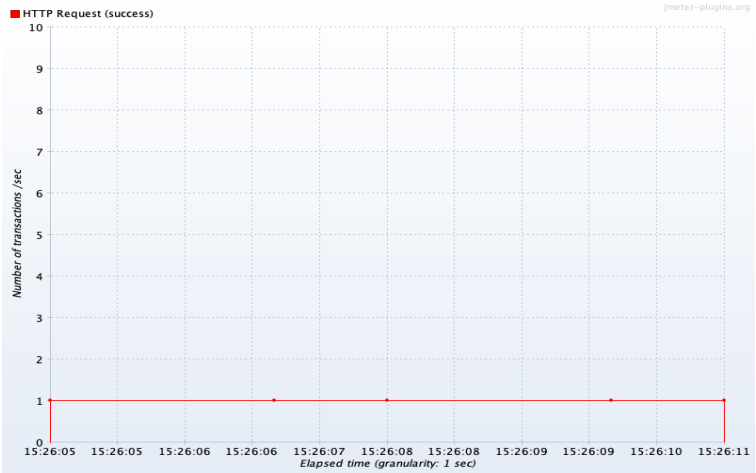
(b) Number of threads = 50



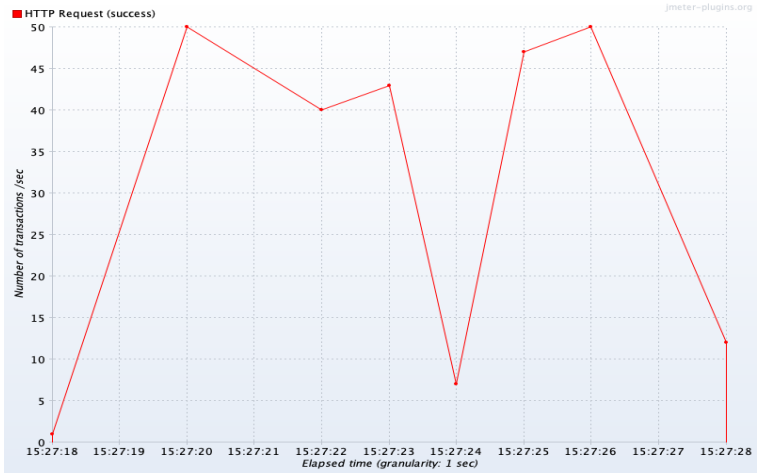
(c) Number of threads = 100

Figure: Response time distribution

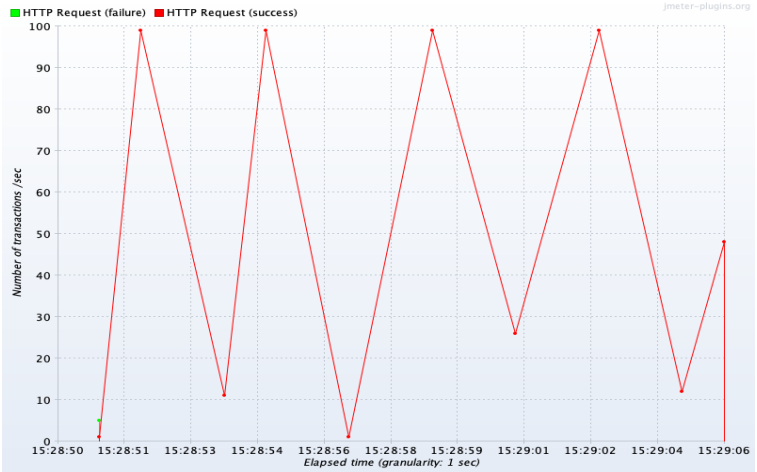
- Response time is the total time it takes for a system or component to respond to a request. It includes the time required for processing the request, executing any necessary operations, and preparing the response.
- In contrast, latency focuses specifically on the time it takes for a request to travel from the source to the destination and back. It represents the delay or time lag introduced by the network or communication channels during the transmission of the request and response.
- Similar to latency, response time also increases with an increasing number of threads.
- Almost all of the requests took between 1 and 2 seconds to complete in the case of a single user making the requests, whereas as the number of threads got increased to 100, response time for the majority of requests shot up to 3-4 seconds.



(a) Number of threads = 1



(b) Number of threads = 50



(c) Number of threads = 100

Figure: Transactions per second

- In the case of a single thread making requests to the server, we observe a constant throughput of 1 tps (transactions per second).
- As we increase the number of threads, the throughput also increases. The throughput can reach as high as the number of concurrent threads.
- But the variability in throughput increases at a higher number of threads, so a relatively lower number of threads is preferred for applications that demand a comparatively constant throughput.

Future Work

- One of the immediate directions to pursue would be implementing the Authenticated Data Structures (ADS) layer.
- IntegriDB is a promising open-source ADS that can be used.
- Another area for future work is addressing the privacy issue posed by FalconDB.
- As a transparent public database, FalconDB makes the data accessible to all participants, including untrusted servers.
- We can instead use encrypted databases in our system.
- However, exploring ways to query and update encrypted databases securely remains an open problem in academia.
- These areas are ripe for further research and development to enhance the security and privacy of our collaborative database system.

- It is worth exploring the utilization of alternative distributed ledger technologies (DLTs), such as Hedera, that offer enhanced transaction speeds, reduced latency, and efficient scalability.
- These DLTs have demonstrated notable improvements in these areas, which could potentially contribute to the advancement of various applications and systems.
- By considering adopting and integrating such DLTs, organizations can aim to achieve higher transaction throughput, faster response times, and improved overall performance.
- Further research and experimentation are required to assess the feasibility and suitability of these DLTs in specific use cases, with the potential to unlock new possibilities and opportunities for innovation in the field.
- Expanding the investigation into these alternative DLTs can pave the way for future advancements in distributed systems and contribute to the ongoing evolution of decentralized technologies.

References

- Peng, Y.; Du, M.; Li, F.; Cheng, R.; Song, D. FalconDB: Blockchain-based collaborative database. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020; pp. 637–652.
- Fekete, D.L.; Kiss, A. A Survey of Ledger Technology-Based Databases. Future Internet 2021, 13, 197.