CHARACTERIZATION OF EFFICIENT SECURE DATA PIPELINE UTILIZING HYPERLEDGER FABRIC BLOCKCHAIN

by

AURA TEASLEY, B.S.

THESIS
Presented to the Graduate Faculty of
The University of Texas at San Antonio
In Partial Fulfillment
Of the Requirements
For the Degree of

MASTERS OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
College of Engineering
Department of Electrical and Computer Engineering
August 2022

DEDICATION

To my mother and father, I give you all my love. To Buddy, my son, you have everything.

ACKNOWLEDGEMENTS

Thank you deeply to my committee, they have exerted a very generous amount of patience as I developed this work. I'm grateful to the goals we set together because they exceeded what I thought I was capable of and I'm grateful to all of you for allowing me to reach them. A special acknowledgement goes to my advisor, Dr. Prevost. Even with your kindness, you're wise enough to let me learn from my mistakes and I'm very glad I chose your lab.

August 2022

CHARACTERIZATION OF EFFICIENT SECURE DATA PIPELINE UTILIZING HYPERLEDGER FABRIC BLOCKCHAIN

Aura Teasley, M.Sc.
The University of Texas at San Antonio,

Supervising Professor: John J. Prevost, Ph.D.

Blockchain technology has been considered as a viable solution to address the concerns of trust and central points of failure in centralized systems. From it's inception, blockchain has been hailed for its promise of failure resilience, alteration intolerance and privacy preservation. For these reasons academia has sought methods to widen the scope of blockchains applicability however blockchain's qualities must be challenged at scale before the platform may be considered as a viable solution for major service providing organizations. We define that blockchains scope of observation is limited to system wide throughput and latency and propose a hierarchical model to observe fine-grained performance metrics. Our observations yield an observable bottle neck with the ordering service and an unexpected sensitivity to the tuning of block size.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Coined as the immutable ledger, blockchain has gained popularity since its initial proposition by Satoshi Nakamoto as a trustless peer-to-peer network that allows for digital trading without the use of financial institutions [7]. Since then, a growing number of researchers have found viability in using blockchains for a multitude of use cases including decentralized financing, supply chain, healthcare, and many more. Chen et al. highlights blockchains part in a financial revitalization, stating that transactions recorded on blockchains are valid, immutable, and verifiable; allowing for blockchains to serve as a common source of truth for transacting parties [4]. For these same reasons, Saberi et al. details the assistance blockchain technology can bring to mitigate control and sustainability complexities associated with global supply chain management [9].

However, centralized services already have existing mechanisms to support validated and verifiable client and supplier actions, yet there still exists a great interest in utilizing a decentralized trust-less system. This is because trust has been broken so many times before with centralized services. In the food industry, we can acknowledge the general waning in confidence as food safety incidents occur too frequently with Salmonella, Listeria, and E. Coli outbreaks. Xia et al. further illuminates the lack of trust in cloud services as clients lose control of their data. Once data leaves a users device, malicious actors can steal and share private information causing clients and service owners several legal and reputation-attacking problems [13].

While many have proposed integrating security, traceability and visibility to mitigate similar instances, the very important question of whether information shared in these traceable systems can be trusted must be asked [12]. Moreover, our high dependence on centralized services demands the need for resilience to fault, failure and malicious actors. Canada's internet outage in July 2022 highlights the deep concern associated with heavily centralized services. The disruption of this service left millions without access to the internet, emergency services and digital funds. This event ensued decisions from investment executives to suggests that higher participation from competitors along with redundancy would mitigate similar instances [8]. This beckons the enthusiasm directed towards blockchain technology.

Blockchains functionality is attributed to a population of either competitive or collaborative peers. Each peer has an incentive to uphold the global ledger and possesses only a fraction of power to update it. As long as a majority population of peers agrees to the next state of the ledger and upholds it, each peer is trusted to provided the most recent state of a digital asset. Further fortified with unbiased automated business logic, transaction traceability, and ledger redundancy, blockchain technology has identified itself as a candidate solution to address the trust, corruption and single point of failure concerns associated with centralized services.

In this thesis, we propose that empirical analysis along with the metrics of throughput and latency limit the scope of blockchain performance in Chapter 2. Blockchains and their operations will be defined in depth in Chapters 3 and 4, respectively. Chapter 5 highlights existing work performed in this field to characterize blockchains performance. We add to this body of work in Chapter 6 as we introduce our model based on Markov Chain Queuing Theory. Results and comparisons to relating work are made in chapter 7. Lastly, Chapter 8 discuss insights gained, persistent concerns and future work that can be performed to better support applications that would benefit from blockchain technology.

CHAPTER 2: PROBLEM STATEMENT

While decentralized and centralized services have stark variances, they share an obfuscating monolithic structure that inhibits characterization and limits observation to terminal performance metrics such as throughput and latency. Conventionally, empirical testing would support characterization through discrete tuning of known hyperparameters however given the growing number of parameters and unknown constraints, empirical analysis is a costly operation that returns a marginal understanding of system behavior [15]. Moreover, as businesses apply blockchain technology and scale it for their expected workload, empirical analysis no longer maintains accuracy due to the varied system environments that blockchain will be subject to. To address these concerns, our research characterizes blockchain performance through theoretical analysis and further approaches blockchain characterization as an optimization problem.

This thesis mainly seeks to contribute insight into the relationships between performance and security for the private blockchain network, Hyperledger Fabric. Our theoretical model is based on Markov Chain Queuing Theory allowing us to modularize blockchain into several major systems and perform piece-wise characterization through observations in efficiency. To add to the growing pool of researchers who have also developed hierarchical models, our work will challenge:

1. What affect does endorsing policy have on system performance

2. How efficient are the major subsystems of Hyperledger Fabric

3. Are there observable trends in performance degradation as network configurations alter

CHAPTER 3: BLOCKCHAIN NETWORKS

Blockchain networks can be classified into three major archetypes: public, consortium and private. Their variances are attributed to their trust policy, consensus population, validation population, ledger accessibility and client permissions, detailed in Table 3.1.

Blockchain Network Archetypes			
Aspect	Public	Consortium	Private
Trust Policy	Trustless	Trusted	Trusted
Consensus Population	Public	Selected	Private
Validation Population	Public	Selected	Private
Ledger Accessibility	Public	Public/Private	Private
Client Permissions	Public	Public/Private	Private

Table 3.1: Blockchain Archetypes

Blockchain Performance					
Blockchain	Archetype	Consensus	Throughput	Latency	Reference
Bitcoin	Public	Proof of Work	<10 txns/sec	10 minutes	[2]
Litecoin	Public	Proof of Work	<10 txns/sec	10 minutes	[2]
Tangle	Public	Proof of Work	>1K	1 minute	[2]
Peercoin	Consortium	Proof of Stake	>10 txns/sec	10 minutes	[2]
Ouroborous	Consortium	Proof of Stake	>100 txns/sec	1 minute	[2]
Algorand	Consortium	Proof of Stake	>100 txns/sec	20 seconds	[2]
Ethereum	Consortium	Proof of Stake	>10 txns/sec	10 seconds	[2]
ByzCoin	Consortium	PoW+PBFT	>100 txns/sec	10 seconds	[2]
Avalanche	Consortium	PBFT	>1K	1 second	[2]
Cosmos	Consortium	Proof of Stake	>1K txns/sec	1 second	[2]
Ripple	Consortium	PoW+PBFT	>1K txns/sec	<1 second	[2]
Hyperledger Sawtooth	Private	PBFT	<1K	10 seconds	[2]
Hyperledger Fabric	Private	PBFT	>1K txns/sec	1 seconds	[2]

Table 3.2: Blockchain Performance

Public Blockchains

Public blockchains are often synonymous with permissionless blockchains, thus there does not exist an authority that governs whether a client may join or gain privileges to run a blockchain node. The only requirement is that clients must be associated with a valid pseudonym to tether their identity to a digital wallet. Given the lack of governance, public blockchains are often densely populated with clients and likely have a population of malicious nodes [14]. Consensus algorithms are the responsible entity that mitigate malicious acts, coordinate distributed actions and sequence concurrent transactions.

Consensus Algorithms

Proof of Work

The Nakamoto Consensus Algorithm a.k.a the Proof of Work consensus algorithm allows for peers to prove their willingness to participate in consensus by mining. Miners are subject to a computationally hard hash challenge of varying levels of difficulty. Hashing puzzles can be solved by computing the hash of a block given several known inputs with one unknown value called the nonce. Miners will take the inputs mentioned earlier and will enumerate through a 32 bit nonce value to generate a hash less than a defined target, solving the hash challenge. Upon miners solving a puzzle, miners are compensated with a freshly minted crypto coin and also win the sole right to validate the block they mined. The amount of time to generate the correct hash results in a culmination of computational work that gives the Proof of Work consensus model its name and most importantly supports randomization of the validation population from greedy nodes centralizing power.

Blockchains are immune to most malicious node attacks unless malicious miners aggregate more computational power by processing blocks faster or by owning a majority of nodes. Nakamoto's algorithm further states that so long as all nodes uphold the longest blockchain and communicate with each other, malicious acts are an improbable likelihood. However, due to the large population of blocks being appending on to every local ledger, concurrently validated blocks are an inevitability. Blockchains like Bitcoin and Ethereum guarantee eventual finality with some probability, meaning that all local ledgers share a common prefix and that blocks being added to these ledgers are subject to being revoked however this likelihood reduces as more blocks are added [3].

Drawback of Public Blockchain Consensus Models

Major concerns regarding the Proof of Work consensus algorithm focal around its dependency on computational work. Blockchains supporting these consensus algorithms often suffer from low throughput due to an inefficient use of energy and high need for synchronization. Furthermore, the open competition for all miners to partipicate in consensus likely results in miners concurrently solving the same hashing challenge yet only one block will be appended onto the global chain. This yields concerns as a large waste of computational power and time has been exerted with no merit. Moreover, unique hash values fosters authenticity and security however hashing algorithms are subject to collision and prohibit blocks from being generated more efficiently, this does not support modern applications that require real time responses.

Consortium Blockchains

Consortium blockchains can be considered a hybrid of public and private. They address a few energy inefficiency issues by reducing the population of nodes that can participate in consensus. Moreover, only a selected number of nodes are responsible for validating pending blocks. These blockchain variants can be considered demi-centralized given the shifting centralization of validating power [18]. Lastly, these blockchain variants predominately utilize Proof of Stake consensus algorithms or its derivatives.

Consensus Algorithms

Proof of Stake

The Proof of Stake consensus model further raises incentive for mining nodes to remain unbiased as they validate blocks. As miners volunteer to participate in consensus, miners will invest a portion of their cryptocurrency into a special wallet, at this point the money can not be utilized. Miners that invest more funds have a higher likelihood to be chosen to publish a new block. The Proof of Stake consensus model can be branched into variants based on how the miners are selected.

Variants can be based on the magnitude of stake that a miner has invested, age of the unspent stake or by magnitude of stake along with random selection. While miners will not be rewarded with a freshly minted crypto coin, they will be given a small transaction fee that the client submitted to process the transaction as compensation. Further, clients can submit larger transaction fees to express that they wish for the transactions processing to be expedited. Unlike Proof of Work, Proof of Stake has deterrents to sway miners from acting maliciously. Miners are subject to losing their stake if they have been detected to corrupt a transaction. Further, the digital identity will be identified and banned from participating in consensus thereafter. Miners that attempt to participate in consensus by submitting a stake less than the transaction fee to be gained are less likely to be chosen. Proof of Stake still suffers from the same weakness of Proof of Work based consensus models being that centralized power will effectively corrupt the mining pool resulting in biased and potentially corrupted blocks.

Delegated Proof of Stake

The Delegated Proof of Stake consensus model attempts to democratize voting power by installing elected delegates. Miners interested in gaining benefits from participating in consensus may invest their stake into these delegates. A limited number of delegates exists, thus miners will aggregate their stake with other miners to increase the chance of their delegate participating. No delegate is eligible to participate in consecutive consensus, thus swaying delegates with more stake from centralizing power. A benefit that this consensus model gains from this orientation is a reduced population of validators, allowing for blocks to be generated at a much faster rate.

Drawback of Consortium Blockchain Consensus Models

Proof of Stake still falls to several attacks including nothing-at-stake and long-range attacks. A nothing-at-stake attack against a Proof of Stake based blockchain is simplified as a large number of shareholders maintaining several blockchains concurrently, manipulating the fact that very little computational power is required to create a Proof of Stake based blockchain [5]. Long range attacks are possible when an attacker possess 1% of all wealth on the network and initiates a fork after the genesis block. The attacker then has the power to recreate the main chain in their preference. Distributed denial of service (DDoS) attacks are one of the most common network bandwidth consumption attacks that have caused trouble for these variants. DDoS attacks on blockchain platforms are similar to the expected approach of generating a large number of small illegitimate transactions and eclipsing miners from receiving valid transactions. A real attack executed this by exploiting the SUICIDE opcodes where attackers simply created a large numbers of cheap, empty accounts on the Ethereum network and through the use of these opcodes, new accounts were 'poked' into existence then repetitiously called [5].

Private Blockchains

While private blockchains are seemingly best suited to address security concerns, these variants possess several centralized units that position the network to have performance bottlenecks during block generation and block validation. Despite this, private blockchains are associated to have the most ideal performance metrics of all variants due to their limited population of participants in consensus and validation. Private blockchains gain their claim to security due to stringent rules embedded into the network. These rules govern who may interact with the network as well their privileges preventing eclipse and Sybil attacks and assisting in identification of malicious clients. Permission to interact with the network is granted to clients and nodes through a centralized authority, which possess a root certificate that distributes derivative certificates that can be referenced back to the root for authenticity. Clients and nodes are associated with isolated channels that obscure knowledge of other interactions and parties that exists outside their channels. Asymmetric cryptography is utilized to encrypt information sent from clients and nodes that exists on these channels, enabling further privacy of data transmitted even given man-in-the-middle attacks. Lastly, channels are instantiated with inherent rules that all participants must adhere to. Only network administrators govern whether these rules may be altered and moreover network administrators define the requirements of which nodes may participate in consensus. Consensus has been altered to be a requirement for all nodes listed in the policy and incentives are no longer necessary.

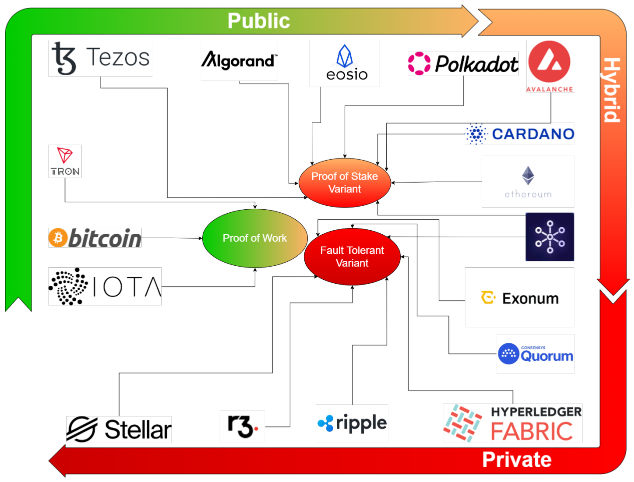


Figure 3.1: Major Consensus Models for Blockchain Archetypes

Certificate Authority

Private permissioned blockchains, like Hyperledger Fabric, may only support private sharing of information though use of certificates. Network administrators are usually the individuals that propose a new client or member to be added to the network and administrators will be the entities to submit the registration for client enrollment. This disables anonymity however client accounts are protected with a user name and a password which is retained in a local database. For every action a client makes on the network, their identity will be referenced. Administrators will always maintain the ability to revoke clients from interacting with the network, should the entity no longer need to do so or if the entity has been considered malicious. Clients, like all members are granted asymmetric keys provided and signed by the certificate authority, this further allows for all members to have identities associated with every other user and for all clients to trust the root certificate authority that signed the certificate. Should users be revoked from the network, the certificate authority will associate the public certificate of the member to the certificate revocation list, allowing for all other users to know that this client is not allowed to interact with the network.

Membership Service Provider

The membership service provider, MSP, defines the rules for valid actors in an organization. The MSP is usually defined by configuration files that associate one or several roles to predefined members that can exists on the network. This includes, members, peers, admins and clients. Each of these members can be granted privileges such as readability or write-ability to the ledger. Further, members can be given elevated privileges to modify the existing rules of the network, allowing for a hierarchy of roles.

Endorsing Policy

Through mandated participation, members defined in the system channel policy weigh in on whether a pending transaction should become valid are support through a series of endorsed transactions. Specialized nodes granted with the ability to host an executable will ingest client requests to interact with the executable. Every executable is defined to be syntactically similar and will support the same business logic. Through this business logic, digital assets can be created and modified, moreover, assets can change ownership. Upon request from the client, endorsing peers will initialize the executable with inputs given from the client and upon receiving a new asset being created or an asset being modified, the responses across all peers that performed this process will be compared. Endorsing peers must adhere to the system channel policy where one member of each organization on the system channel endorsing policy must sign off on every transaction proposal before the proposal can be propagated further to be packaged into a block.

Consensus Algorithms

Byzantine Fault Tolerance

Byzantine fault tolerant (BFT) consensus algorithms are at the core of providing safety and liveness guarantees for distributed systems that must operate in the presence of arbitrary failure. These algorithms coordinate multiple servers in a leader-follower orientation where leaders will be elected through popular vote from the followers. During which, leaders will copy instructions to all followers and tasks the followers to adhere to their instructions and propose their response to each other. A leaders instructions will only be followed and broadcast to the network upon a majority vote being achieved by the followers. Since leaders are the primary authority, the messaging and time complexities associated with distributed systems are not applicable, allowing for private blockchains to obtain immediate block finality [17].

Practical Byzantine Fault Tolerance

PBFT proposed the first state-machine replication protocol that provides safety and liveness properties in the presence of Byzantine faults. Byzantine faults are arbitrary faults that occur during an execution of an order, leaving nodes unaware if error an has occurred [1]. This algorithm is often associated with message delay constraints however due to synchronization amongst nodes, resulting in constraints on message delays. Zhang et al. details that this algorithm operates under two major protocols: the replication protocol and view-change protocol. The consensus process is started when clients invoke a request. The leader handles requests and the client starts a timer, each follower submits a response upon which the client will stop the timer when receiving sufficient responses. However, if the allotted time for responses elapses without sufficient responses, the client will consider a fault to exists and will attempt to broadcast the message to all followers forgoing the leader, in the event the leader is faulty [17]. In the event of leader failure, PBFT will adopt the view-change protocol where a new leader will be selected.

Drawback of Private Blockchain Consensus Models

There are several issues associated with BFT models, while servers are trusted, they are allowed to be faulty and submit incorrect messages to other servers, this draws serious concern when the leader may become faulty or malicious. The leader may trigger unnecessary timeouts, introduce collusion, or stop responding all together. These faulty cases can significantly reduce system availability and reliability. Due to this, leader based BFT algorithms produce at worst a single point of failure and at best suffer from heavy workloads which result in a system bottleneck. To avoid these drawbacks, leaderless BFT algorithms coordinate consensus amongst a coordinated group, allowing for this service to become more decentralized and resilient to failure. However, synchronization is again impacted and message ordering is likely not to be the same across all distributed nodes, in turn producing high messaging and time complexities [17].

CHAPTER 4: TRANSACTION FLOW

Client Interaction

Blockchain networks function as a congregation of distributed independent fully functioning nodes that maintain the same state. Clients can interact with any of these nodes to create, modify or transfer digital assets and tightly couple the asset to an anonymous digital identity though use of asymmetric cryptography, seen in Figure 4.1. Batches of transactions submitted to the blockchain are hashed together to derive a Merkle hash root which allows for easier storage of transactions, efficient querying and easy validation of synchronized ledgers across the distributed network, seen in Figure 4.2. Blockchains heavy utility of hashing algorithms is also the reason why digital states can be created, modified and read however they can never be deleted due to the dependency that every block hash has on its immediate predecessor, Figure 4.3. The immutable database of transactions allows blockchain to function as a distributed storage unit however blockchain networks can be made functional by dedicating either blocks or specialized compute surfaces to host customizable compiled logic.

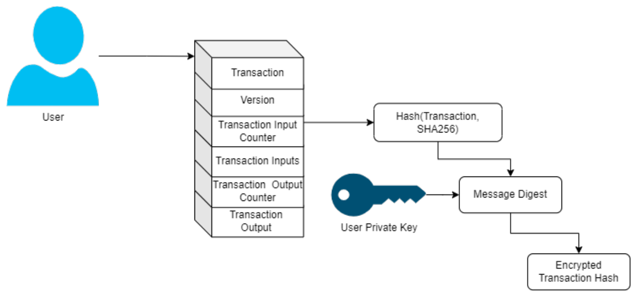

User
Transaction
Version
Transaction Input Counter
Transaction Inputs
Transaction Output Counter
Transaction Output
Hash(Transaction, SHA256)
User Private Key
Message Digest
Encrypted Transaction Hash

Figure 4.1: Transaction Submission into Blockchain

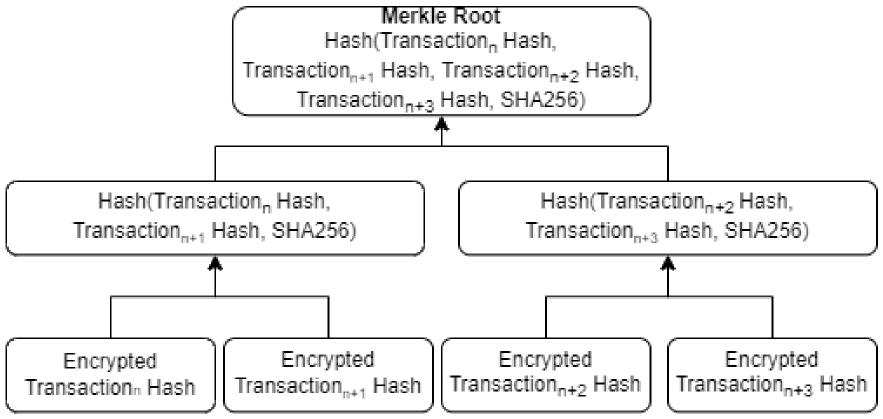

Merkle Root
Hash(Transaction$_n$ Hash, Transaction$_{n+1}$ Hash, Transaction$_{n+2}$ Hash, Transaction$_{n+3}$ Hash, SHA256)
Hash(Transaction$_n$ Hash, Transaction$_{n+1}$ Hash, SHA256)
Hash(Transaction$_{n+2}$ Hash, Transaction$_{n+3}$ Hash, SHA256)
Encrypted Transaction$_n$ Hash
Encrypted Transaction$_{n+1}$ Hash
Encrypted Transaction$_{n+2}$ Hash
Encrypted Transaction$_{n+3}$ Hash

Figure 4.2: Merkle Hash Tree

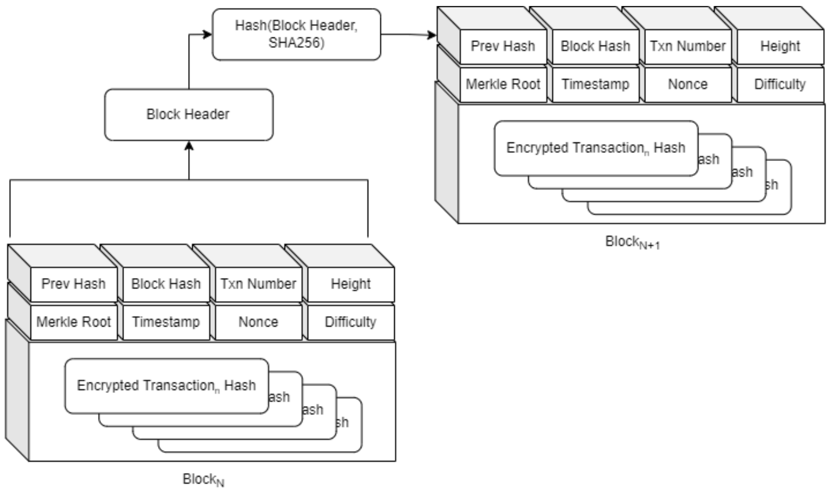


Figure 4.3: Generation of Blocks

Peer Nodes

Hyperledger Fabric governs itself through an organized delegation of power. Some delegates such as the peer node server has more power than other servers, such as the orderer. Powers of the peers can be further tiered into roles based on the network administrators security or performance posture. To maintain the normal behavior of blockchain, where all peers can be queried to collect the state of a digital asset located on the blockchain, all peers must host the blockchain and must have a tether to a state database. State databases can be modified to less complex databases such as LevelDB if the intended application does not commute large data, other databases are better suited for such applications. While all peers are hosts to both the blockchain and the database, special peers are granted permission to communicate with other peers. These peers provide anchors for the network and are the reference point that new peers will use to be made aware of the present state of the blockchain, this functionality is most useful when network administrators are scaling the network. The involvement of peers with this functionality assists greatly with throughput. Due to the enormous resource constraints applied onto the peer server, most applications that seek to utilize blockchains must have a dedicated environment that can support the growing ledger as well as the load of transaction requests. Further, due to the peers high utility, organization of the peer node is pivotal to the network achieving its most ideal security-performance posture.

All blockchains delegate a large responsibility to their distributed servers to handle incoming transactions and queries, making peers servers a significant component if performance and security are brought into question. Hyperledger Fabric presently hosts all peer servers along with all other servers through managed containers with one dedicated port exerted towards handling new transactions. With the private blockchain, Hyperledger Fabric, the distributed peer servers have delegated roles to support custom performance and security postures. Peers can be assigned to a number of organizations and then delegated unique roles for every organization. Every peer will at minimum hold an exact copy of the ledger for every channel that the peer is involved with. Peers communicate through gRPC streaming protocols with other nodes to communicate updates to transaction proposals and update the state of the blockchain. Most significantly, peers belonging to a permissioned network are responsible for validating whether a transaction inside of a block is signed by the correct party. A detailed view of all blockchain members and their major roles and responsibilities can be seen in Figure 4.4

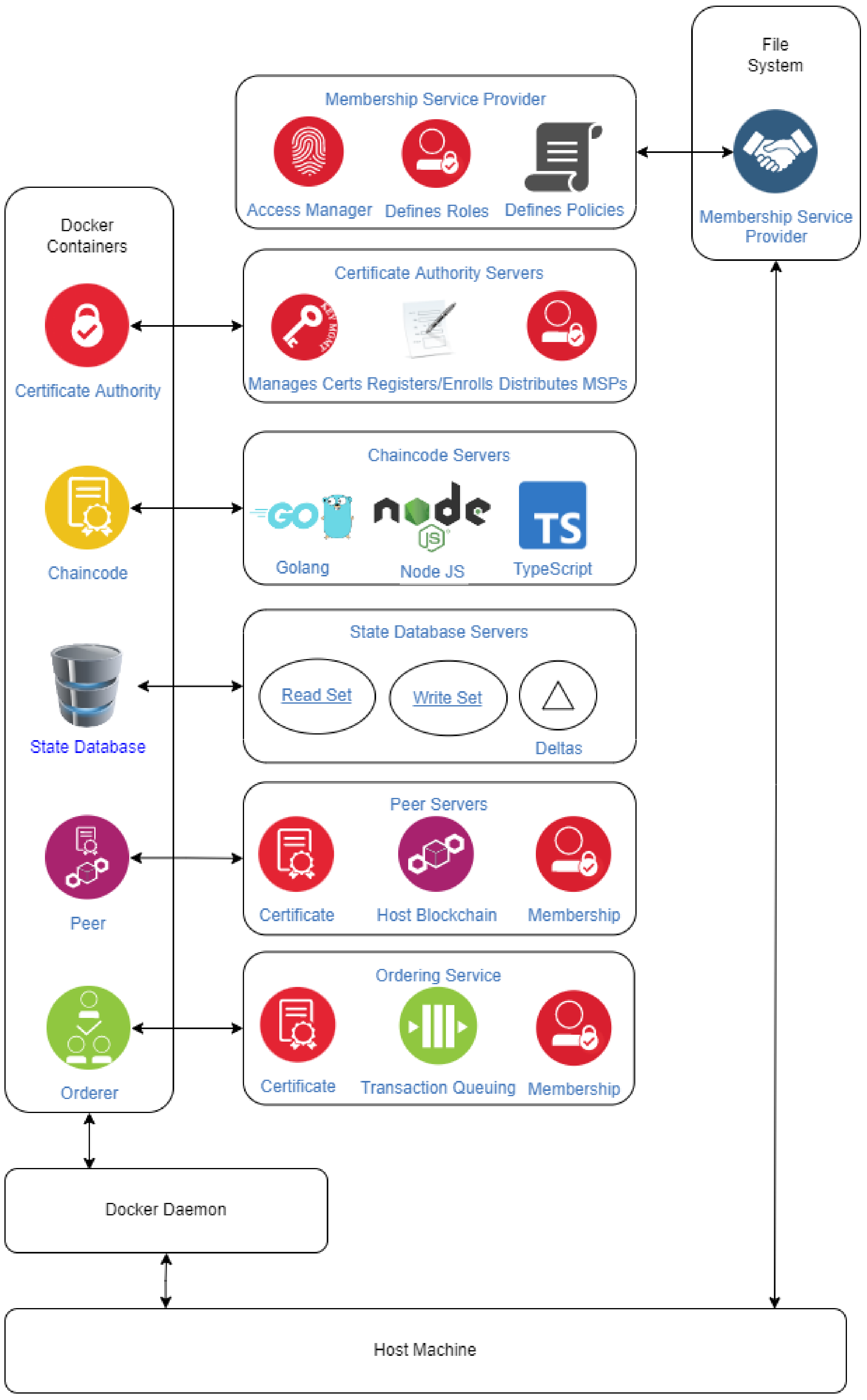


Figure 4.4: Network Architecture of Hyperledger Fabric

Transaction Execution

Many have condensed blockchain networks into a sequence of actions: transaction execution, ordering, block commitment to the ledger. Should a client wish to make revisions to their digital asset, whether that be to create or modify, they must execute the business logic that handles state transitions associated with the digital asset. This is accomplished through the use of the chaincode. Embedded as compiled logic on specialized endorsing peer nodes, the presence of the chaincode identifies a privileged peer rank that mandates that peer to participate in endorsement. Responses returned from chaincodes must be deterministic. Endorsement may require some input from the user to transition the state of the digital asset to another state and as a result a set of read and write deltas are generated. These key value pairs are the entries queried from the state database and are crossed again the action that the client wishes to take. Any digital assets added to the database or modifications to the read keys will result in a write set representing the deltas intended to be implemented into the world state Figure 4.5.

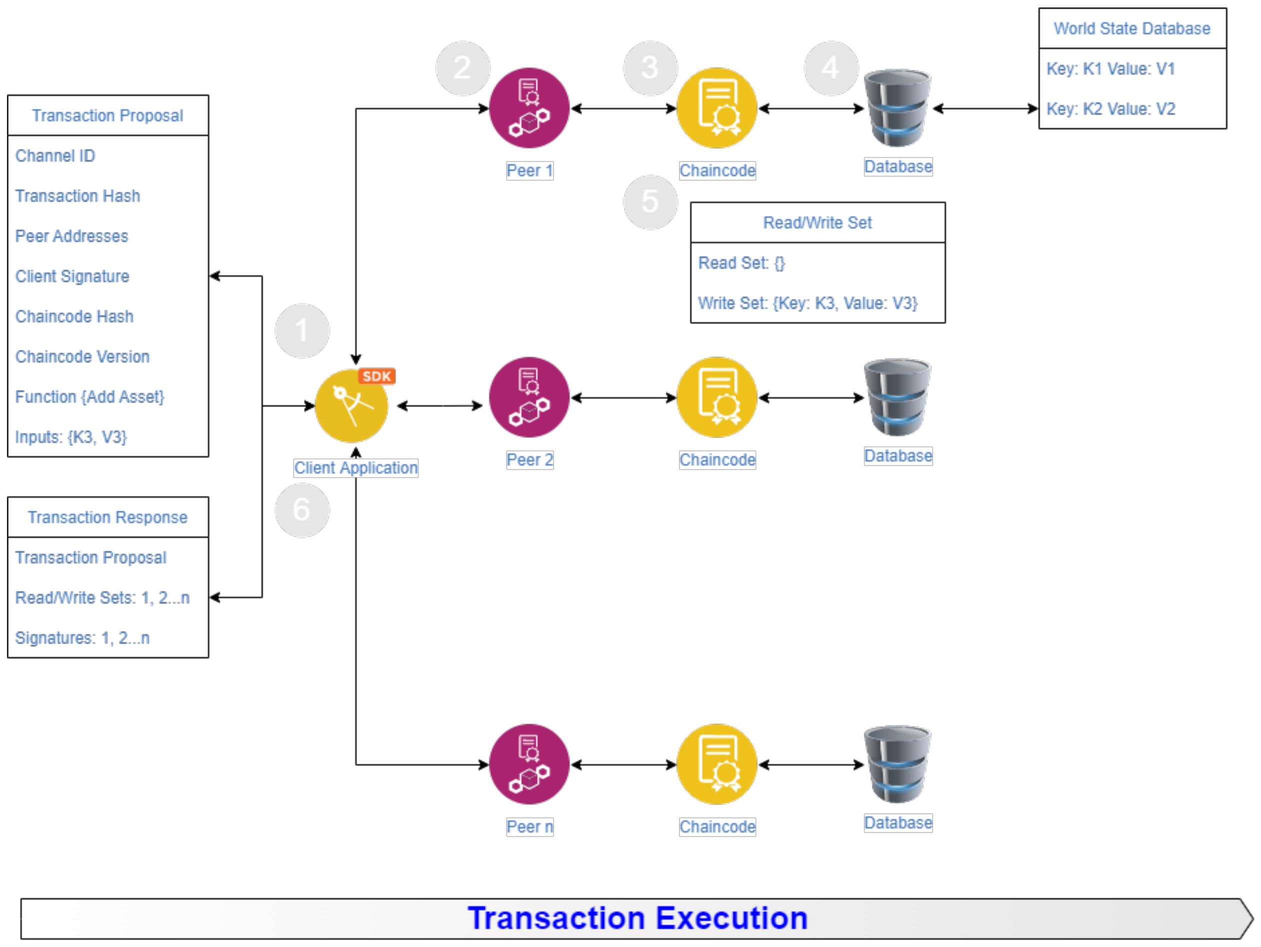
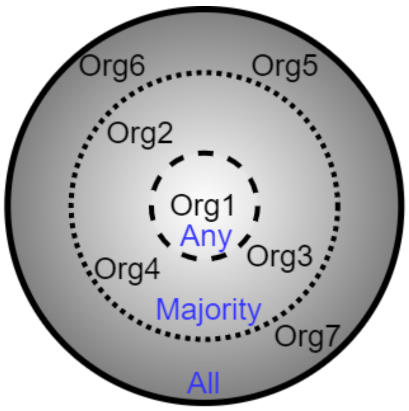
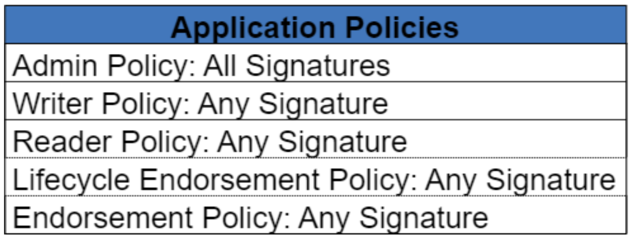


Figure 4.5: Transaction Endorsement and Execution

Transaction Endorsement

Transaction requests that successfully produce a read and write set will be signed by the endorsing peer responsible for querying the world state. Defined in the system channel, the predecessor of all channels of the network, exists the defined rules of the network. This includes the signature policy that defines which privileges each member of an organization possesses. From this, the system channel will be referenced every time a network party interacts with the network and their membership certificate will be challenged against the present rules of the system channel. Further, the endorsement policy assesses how many responses from the endorsing peers are need before the transaction request can be validated and passed along to the subsequent step of transaction ordering. The signature policy will be joined in concert with other policies such as the implicit meta policy and application policy to quickly reference entities as needed to satisfy the endorsement policy, seen in Figure 4.6.





Application Policies
Admin Policy: All Signatures
Writer Policy: Any Signature
Reader Policy: Any Signature
Lifecycle Endorsement Policy: Any Signature
Endorsement Policy: Any Signature

Figure 4.6: Channel Policies

Transaction Ordering

Transactions that meet the constraints will satisfy the application channel endorsement policy. Policies can range from flexible to stringent, where flexible policies only require one peer member from any organization defined in the policy to sign off on a transaction. This is a vital area that the private blockchain uses to deviate from other blockchains, being that majority vote is not always required for a transaction to become valid. Networks can be instantiated with orientations seen in Figure 4.6 where 'any' one member of the endorsing population can satisfy endorsement and analogous cases can be made for 'majority' and 'all' inclusive policies. Transactions that adhere to the signature policy of the system channel will then be forwarded to the ordering node for sequencing.

Raft Consensus

Consensus can be achieved a number of ways as seen with how other blockchain networks implement consensus. However, the general orientation is the same, pending transaction are pushed into blocks, pending blocks have some action applied to them to validate they are appropriate, and blocks are broadcast to peer nodes that hosts the ledger. Managing this collective broadcast, is the leader-follower ordering service, Raft. Similar to the leader follower orientation of Kafka services, the raft consensus protocol enables only the leader micro server to act as a recipient to all pending blocks, all pending blocks are translated into sequence of instructions and the leader replicates these instructions on other logs for the follower micro services to replicate after an allotted time, seen in Figure 4.7. This consensus mechanism is crash-fault tolerant meaning, with the number of micro servers inside the consensus protocol, less than 50% are allowed to be faulty or malicious without compromising the cluster. Leader nodes are subject to replacement as well, should they be compromised, this is adhered to by leader nodes sending out period heartbeat gRPC calls, helping the follower nodes understand they the leader is functioning. Should the leader be offline, the followers will hold an election to vote for a new leader.

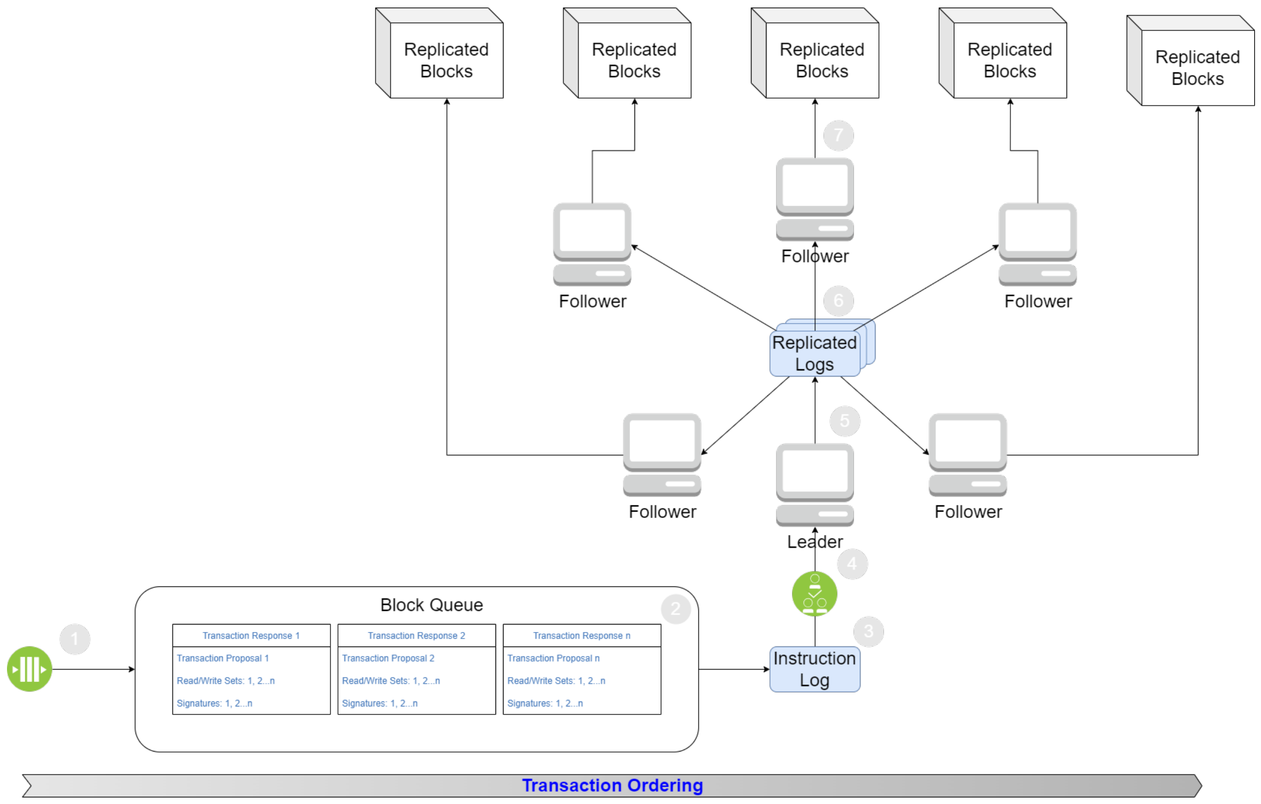


Figure 4.7: Transaction Sequencing with Raft Consensus Algorithm

Transaction Validation and Commitment

Validation is the last stage of the transaction process. All peers in the isolated channel will perform a final inspection of the transactions packed in each block. Transactions that have not satisfied the endorsement policy will be marked as invalid however they will still be appended onto the blockchain. Valid transactions will then submit their read write delta to the database to assess if the read set matches the present state of the database. If all goes well, a new block file will be appended into a local directory in each peer server and write sets will be submitted to local databases to add the new digital asset as a new entry, seen in Figure 4.8.

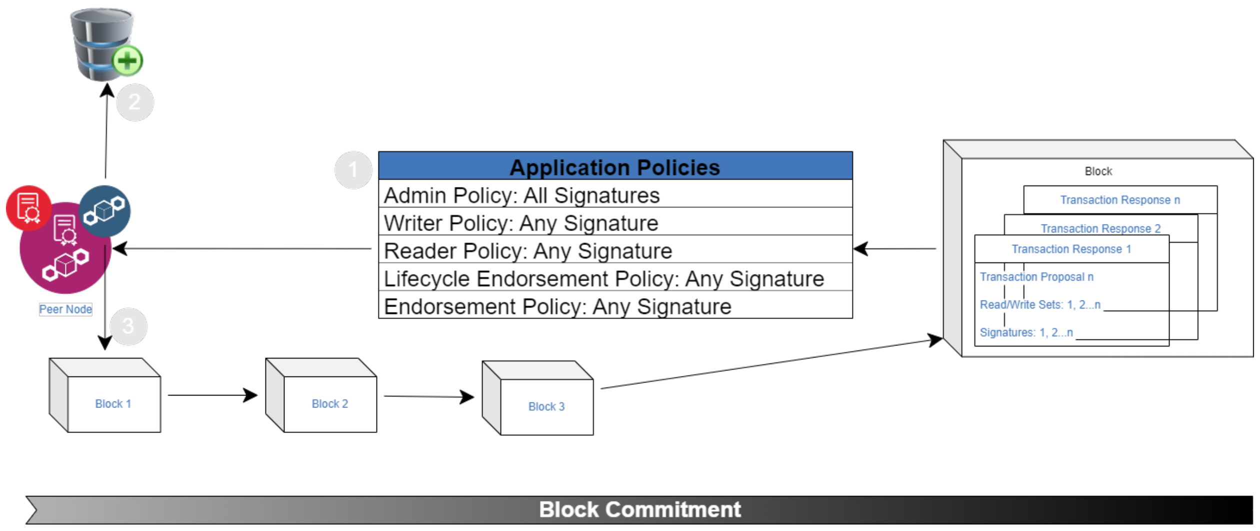

2
1
Application Policies
Admin Policy: All Signatures
Writer Policy: Any Signature
Reader Policy: Any Signature
Lifecycle Endorsement Policy: Any Signature
Endorsement Policy: Any Signature
Block
Transaction Response n
Transaction Response 2
Transaction Response 1
Transaction Proposal n
Read/Write Sets: 1, 2...n
Signatures: 1, 2...n
Peer Node
3
Block 1
Block 2
Block 3
Block Commitment

Figure 4.8: Validation and Commitment

CHAPTER 5: LITERATURE REVIEW

Despite efforts invested into making private blockchains a functional distributed database, there exists latency in several phases of the network that inhibit it from being further adopted. Sukhwani et al. aptly recognizes the tremendous benefit private blockchain networks bring however concerns revolved around their performance. They further query whether the resilience gained through the practical byzantine fault tolerant consensus model actually inhibits throughput especially with networks of a larger size. Their method of modelling the consensus protocol is through the use of Stochastic Rewards Nets to compute the mean compute time for a network of a size of 100 peer nodes. The group challenges predominately the transmission time of consensus messages between peers, the time to process incoming consensus messages and the time to prepare consensus messages for the next stage. Given a block append frequency of 800 blocks an hour, their team yielded the mean time for consensus increased as more peers join the network. In general, they found that the slope of the amount of time it takes for consensus to occur for a network of one hundred peers increased 5.34 times that of a network of a size of four peers [10].

Sukhwani's group further challenged major questions regarding the efficiency of the peer server, due to it being a critical component of the network. They observe that bursty arrivals dampen peer performance as a large cluster of transaction are appended to the network. This performance model allowed their team to gauge the throughput, utilization and mean queue length associated with these distributed servers. Their assessment was captured through the use of Hyperledger Caliper, the tool developed by IBM to assess the performance of several blockchain networks. Results concluded that more stringent endorsement policies cater to more latency during the execution phase of the transaction life cycle. Further, their observation of the transaction sequencing phase of the life cycle concluded that constraints such as block size and block timeout do affect the efficiency of this process, thus resulting in their observation that the performance bottleneck of the ordering service and ledger writing can be assisted if the block size constraint is larger [11].

Presenting the work of Xu et al., their group performed a theoretical analysis on latency using sequential M/M/1 queues to represent the execute-order-validate stages of a multi-channel Hyperledger Fabric network. Their results validate a positive correlation between block size, block interval and latency, further supporting existing work that agrees to the best case and worst case scenario during block generation. Best case being that the block buffer will be satiated before the buffer must timeout, yielding the worst case. Their results conclude that the ordering node can be the largest contributor to latency, 67%, if the block interval is too high for the expected number of transactions [15]. Yuan et al. further echoes these results by stating that the endorsement policy is a major performance bottlenecks. Yuan supplies an analytical model of Hyperledger Fabric v0.6 using Generalized Stochastic Petri Nets to solve a maximization problem of which system parameters contribute to the highest system throughput. Further utilizing calculated queue length as a metric for latency, their results identified the committing phase as an source of latency during low transaction rates. However, as transaction rates increased, the ordering phase stifled the workload of the committing peers, inducing a bottleneck [16].

Jiang et al. addressed a research gap in the performance analysis of Hyperledger Fabric considering endorsing and block timeout. Their work validated how these timing constraints impact the probability of transaction rejection and delay by utilizing M/M/1 queues. Their group bifurcated the monolithic model into two major processes, transaction execution and transaction validation and utilized this as the base for the probability of transactions being blocked due to a full queue and probability of a transaction being dropped due to endorsement timeout. Based on server utility and queuing length of each server, assumptions were developed on the probability of the endorsing server being available to endorse the transaction prior to a timing constraint. Given the endorsing policy and the probability of the required number of endorsing peers being available to sign the transaction proposal, a tertiary assessment of whether the transaction transitions into block sequencing was developed [6].

CHAPTER 6: METHODOLOGY

Hyperledger Fabric's evolution from versions 0.x and 1.x installed a few performance variances that aren't expressed in related works. Firstly, the Raft consensus protocol introduces potential latency in the ordering phase given its need for node synchronization, a quality absent in the deprecated Solo consensus protocol. Moreover, Hyperledger Fabric v2.0+ introduced a service discovery node. This functions to obscure gateway overhead from the client and makes this logic more efficient, marginally heightened throughput. Our work will generate a theoretical model for the most recent version of Hyperledger Fabric, 2.4, while also considering scalability, block parameters and endorsing policy in mind. Like other works, we start with the standard flow of execute-order-validate, seen in Figure 6.1. We then extend the transaction flow to included two additional phases due to Hyperledger Fabrics integration of a gateway service. We utilized the coupled Continuous Time Markov Chains to classify the multiple stochastic processes of Hyperledger Fabric. Transactions are graded to arrive in a Poisson distribution thus marking the transition from one state to its immediate successor at a rate of λ. Similarly, when a process is finished with a transaction, it will transition back to its prior state at a rate of μ, seen in Figure 6.3. These forward and backward transitions represent the probable inflation and deflation of the queue. Coupling the subsystems of Hyperledger Fabric, we can identify the transition of one system to the next using a Jackson Network. This allows for us to still assume transactions to arrive and be serviced in a Poisson distribution into the subsequent queues. We highlight this in our network, seen in Figure 6.2.

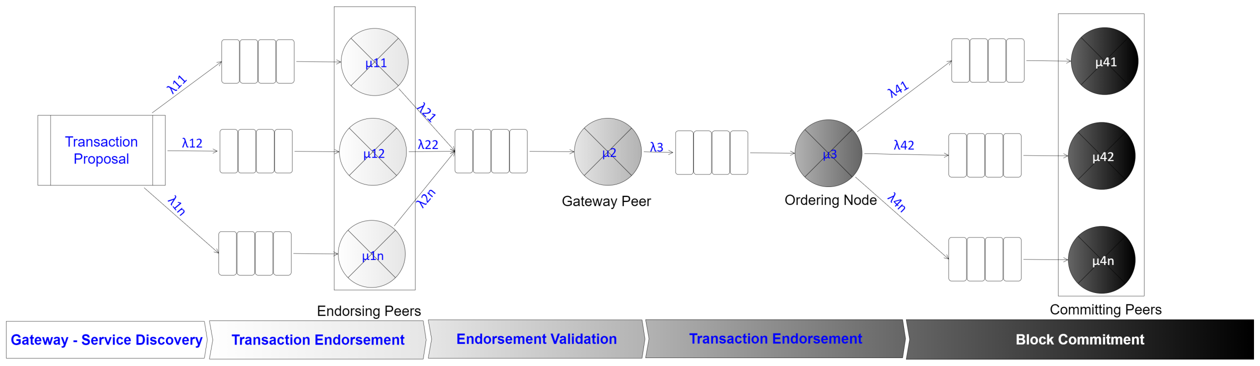


Figure 6.1: Private Blockchain Modelled as Sequential Markov Chains

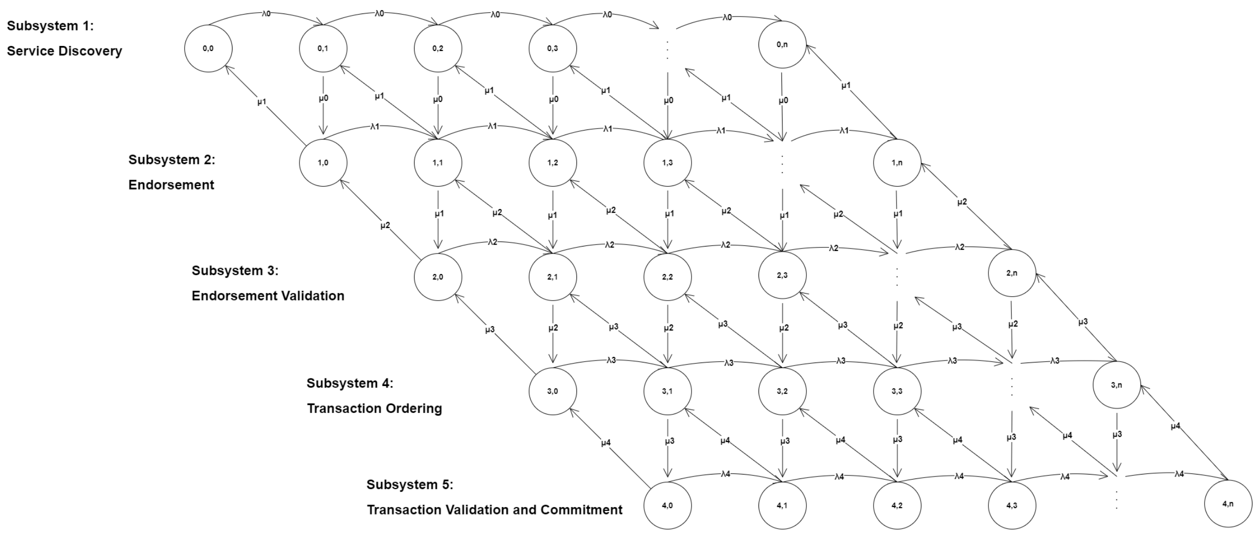


Figure 6.2: Compartmentalized Jackson Network

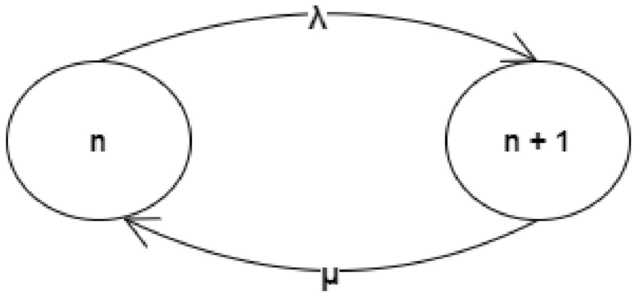


Figure 6.3: Fundamental State Transition

From the principles of queuing theory, we can identify the efficiency of each subsystems by ρ:

$$\rho_n = \frac{\lambda_n}{\mu_n} \tag{6.1}$$

For a general M/M/1 queuing model, the probability of transactions arriving into the queue may be observed as π. Afterwards a simple linear equation can be developed, Equation 6.2.

$$\begin{aligned} \pi_n &= \rho^n \pi_0 \\ \lambda \pi_n &= \mu \pi_{n+1} \end{aligned} \tag{6.2}$$

Generating the network of subsequent queuing models, transactions arrive to populate the queue of one process and must be serviced before it may arrive into the queue of its immediate successor. The probability of state transitions for the first two subsystems can be represented as in Equations 6.3 and 6.4, respectively and can be expanded to express the global state-space in Equation 6.5.

$$\pi_{0,n} = \frac{\mu_{n+1}\pi_{0,n+1} + \mu_{n-1}\pi_{1,n-1}}{\mu_n + \lambda_n} \tag{6.3}$$

$$\pi_{1,n} = \frac{\lambda_n \pi_{0,n} + \mu_{n+1}\pi_{1,n+1} + \mu_{n-1}\pi_{2,n-1}}{2\mu_n + \lambda_n} \tag{6.4}$$

$$P_{m,n} = \sum_{m=0}^{4} \sum_{n=0}^{T} \pi_{m,n} \tag{6.5}$$

Equation 6.5 allows us to solve for rates of transition between the major subsystems of the network. Constraints were applied to dampen the range of possible solutions. This is to uphold M/M/1 theory stating that the rate at which transactions enter the queue can not be higher than the rate that transactions are emptied from the queue, else the queue will grow to an infinite size, expressed in 6.6. Moreover, upholding Hyperledger Fabric's tolerances, transactions are not permitted to take more an a defined time before they must report back to the gateway peer for transaction response validation, this defined time is the endorsing timeout, expressed in Equation 6.7. Analogously, transaction ordering queues will wait an allotted time before they package all transactions in their queue into blocks. At best case, transactions will satisfy another constraint before batch timeout occurs, this would be if enough transactions filled the queue to satisfy the batch size constraint Equation 6.8. Moreover, utilizing Hyperledger Caliper, we identified the average time it takes for

Model Parameters		
Notation	Description	Value
λ_{1*}	Transaction Arrival Rate	10 - 500 txns/s
μ_{ab}	Transaction Service Rate	-
B_S	Maximum Transactions per Block	10 - 300 txns
B_{TO}	Timeout until Block Generation	1 - 3 s
C	Committing Peer Size	4,8,12 peers
E	Endorsing Policy Size	4,8,12 peers
E_{TO}	Endorsing Timeout	20000 ms
P	Payload	50 Bytes

Table 6.1: Markov Chain Parameters

a transaction to complete, Equation 6.9, thus the rate at which transactions are sent to each phase and processed must occur by this time. Lastly, we do want to acknowledge that in two phases, endorsement and validation/commitment, transactions are sent to multiple peers, this introduces complexities when describing the rate at which transactions are sent and received from each of these peers. For simplicity, our model assumes that the variances in these rates are negligible, seen in the constraints 6.10.

$$\rho_n \leq 1 \tag{6.6}$$

$$\sum_{n=0}^{1}(\lambda_n + \mu_n) + \lambda_2 < E_{TO}^{-1} \tag{6.7}$$

$$\frac{B_S}{\sum_{n=0}^{2}(lambda_n^{-1} + \mu_n^{-1}) + \lambda_3^{-1}} = \mu_4 \geq B_{TO}^{-1} \tag{6.8}$$

$$\sum_{n=0}^{4} \lambda_0^{-1} + \mu_n^{-1} <= AverageLatency \tag{6.9}$$

$$\begin{aligned} \lambda_2 &\approx \frac{\sum_{i=1}^{E} \lambda_{2i}}{E} \\ \lambda_5 &\approx \frac{\sum_{i=1}^{C} \lambda_{5i}}{C} \end{aligned} \tag{6.10}$$

Our solution develops a performance analysis of networks based on private blockchains with a focus on endorsing policies. As mentioned earlier, endorsing policies range from lenient to stringent. We integrate this into our assessment by scaling up a Hyperledger Fabric to varying sizes and varying endorsement policies. Three orderer nodes were utilized to support Raft consensus and transactions of varying arrival rates were propagated toward the network to assess how Hyperledger Fabric version 2.4 handles bursty requests.

Lastly, Hyperledger Caliper was utilized to provide a reproducible environment. The tool functions by configuring several files that manage how a workload will be applied to our network. Hyperledger Caliper lastly implements the workload by a leader-follower orientation. A central leader node will spin up as many follower nodes necessary to supply the workload at fixed rates. While our interest was to push a defined workload to the network, due to lack of compute resources the nodes did not maintain the workload, thus our highest transaction arrive rate is roughly 500 transactions per second. We utilized the defined workload as the arrival rate λ_1 and average latency for the Constraint Equation 6.9. These were utilized to solve a minimization problem for Equation 6.5. The large state-space was resolved using Matlab's optimization solver tool. This experiment was performed on a Intel(R) Core i5-1035G1 CPU with 1 GHz clock and 256GB RAM.

CHAPTER 7: PERFORMANCE AND EVALUATION

Results collected were based on coverage maps generated from Matlab, plotting the rate at which major phases are transitioned from one to another. Regions in blue represent lower efficiency of the server compared to lighter regions in yellow which represent the highest record of server utilization. Variances in transaction arrival rate can be observed as we transverse laterally, while variance in network size can be observed lengthwise.

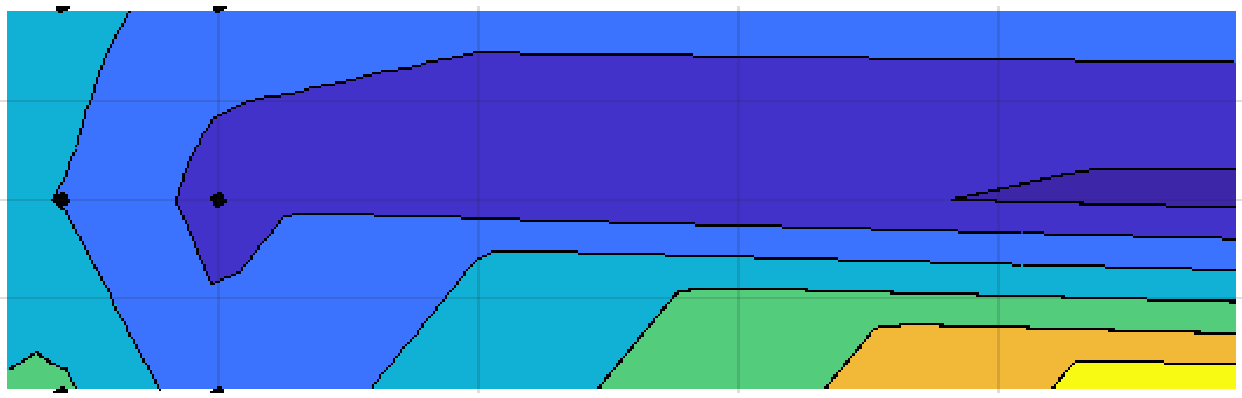


Figure 7.1: Gateway Efficiency: Blocksize 100

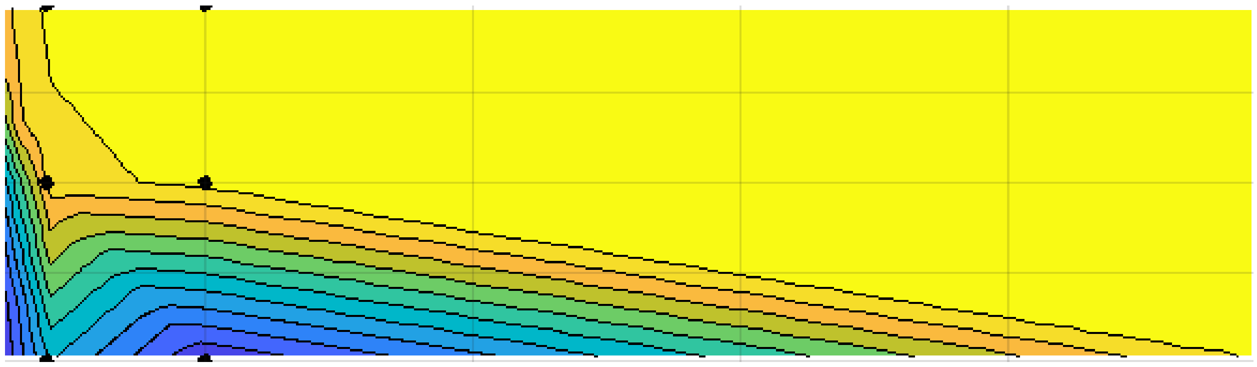


Figure 7.2: Gateway Efficiency: Blocksize 300

Figures 7.1 and 7.2 highlight the coverage maps for the gateway service, the member responsible for service discovery. This service predominately contains embedded logic that forwards transactions to several endorsing peers, thus this server should contain lower latency and high efficiency. The lateral increase in transactions arriving into the network expectantly yields elevated efficiency as the service is keeping up the increasing requests. However, an observation can be made that increases in network size don't result in increased utility of this system. Unexpectedly, changes in block size greatly affect the utility of this server, showing sharp increases in efficiency as transactions arrive at a higher rates. This phase does not interact at all with the orderer yet and transactions are not being prepared to be packaged into blocks.

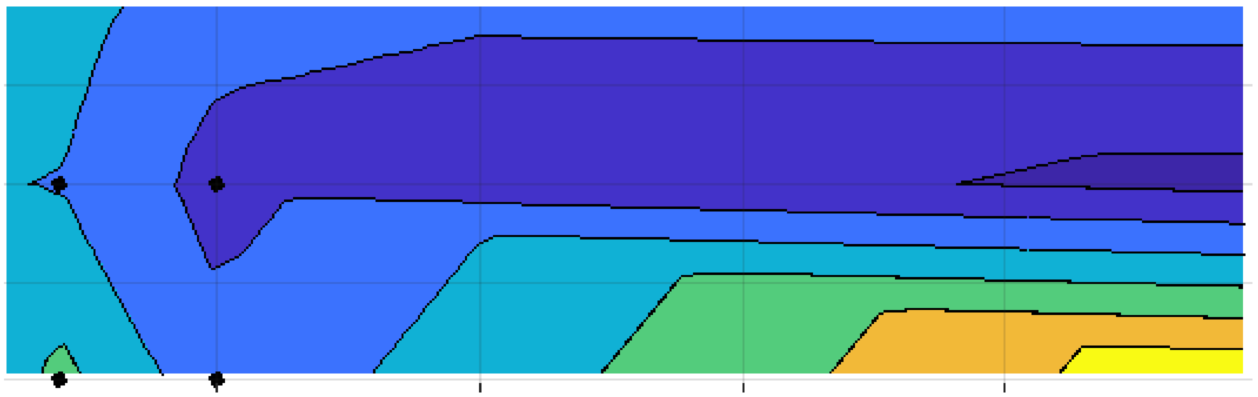


Figure 7.3: Endorsing Efficiency: Blocksize 100

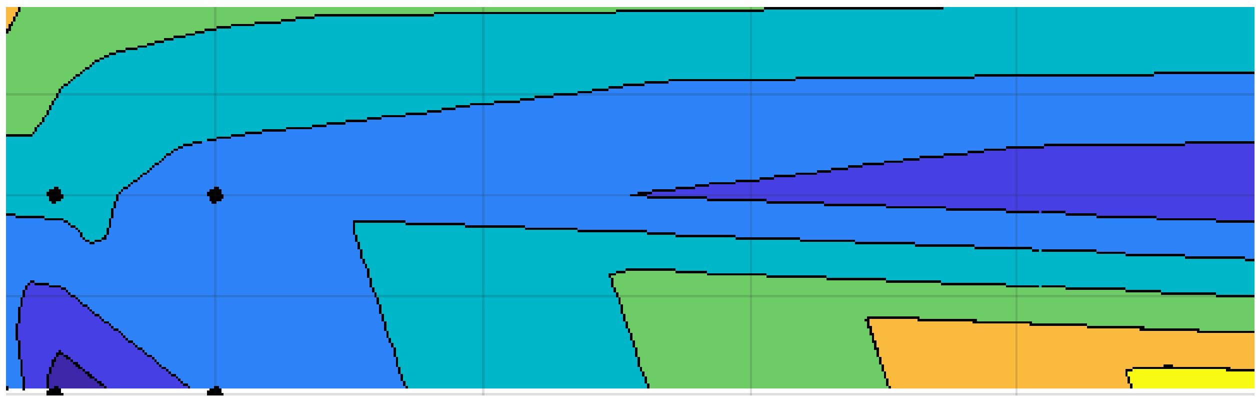


Figure 7.4: Endorsing Efficiency: Blocksize 300

Figures 7.3 and 7.1 further indicating that the gateway do not perform much processing on incoming transactions, allowing transaction to be shuttled to the endorsing nodes with low latency. Slightly variances are noted as we compare 7.3 and 7.4. Both networks respond marginally the same to changes in workload and networks sizes however we still maintain the odd artifact of block size affecting performance, seemingly prematurely. Regions in blue should represent when the endorser is not being utilized, this can occur if the endorsing queue is being depleted relatively quickly and the server is idle for a period before the next transaction.

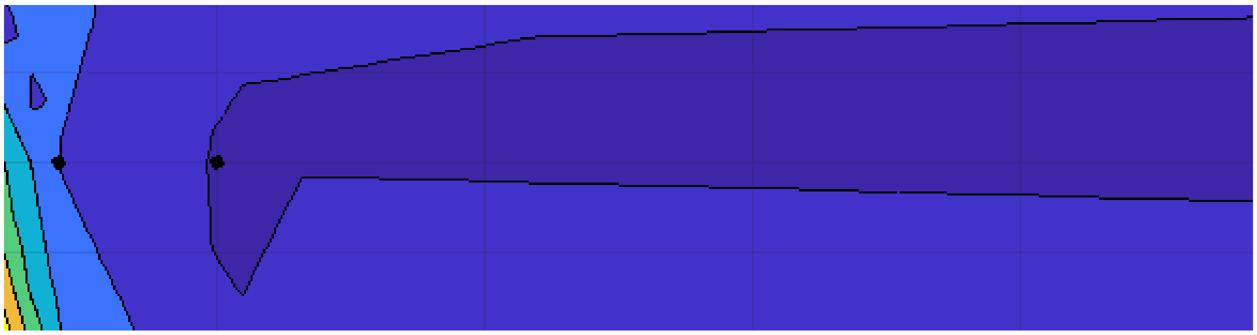


Figure 7.5: Endorsing Validation: Blocksize 100

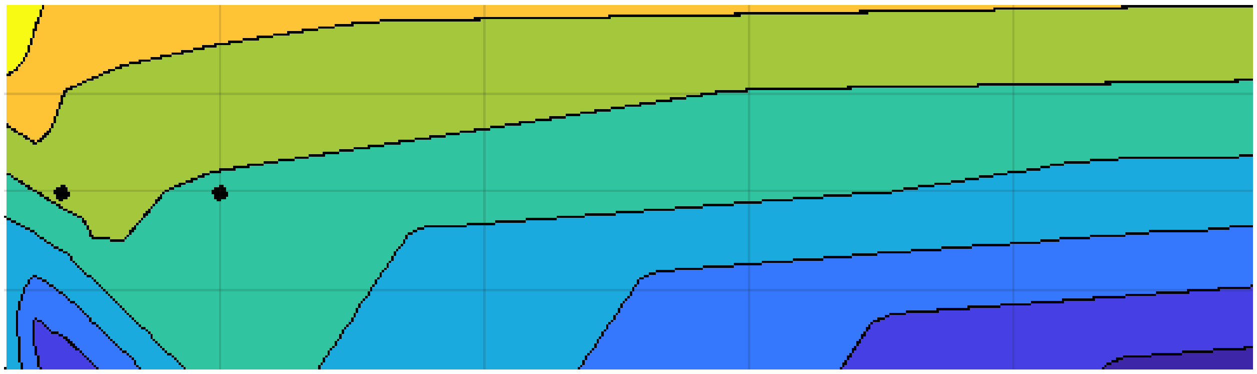


Figure 7.6: Endorsing Validation: Blocksize 300

Figure 7.3 helps supports the general trend in performance degradation as transactions meet communication delays, while Figure 7.4 displays more dynamic transitions. Larger networks need to submit several times more responses back to the gateway than networks that have a smaller endorsing pool, thus it makes sense that the gateway is more busy with larger network sizes.

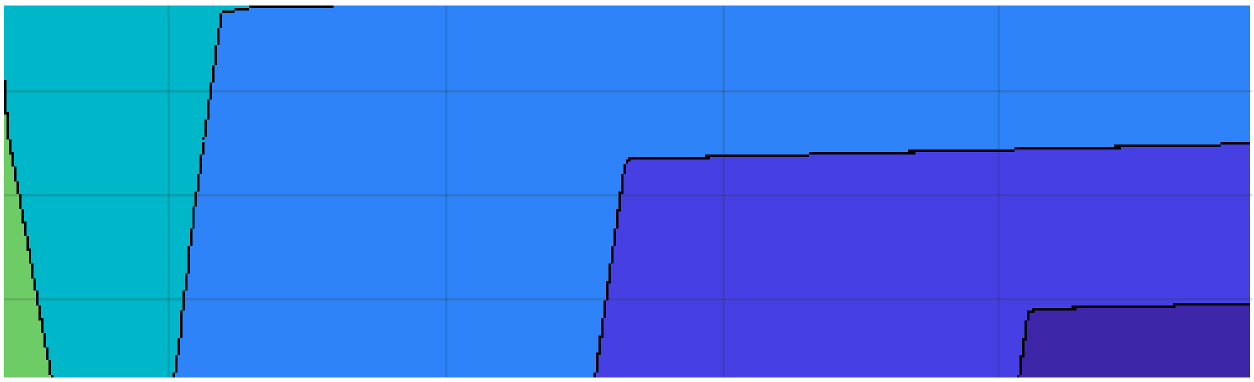


Figure 7.7: Ordering Efficiency: Blocksize 100

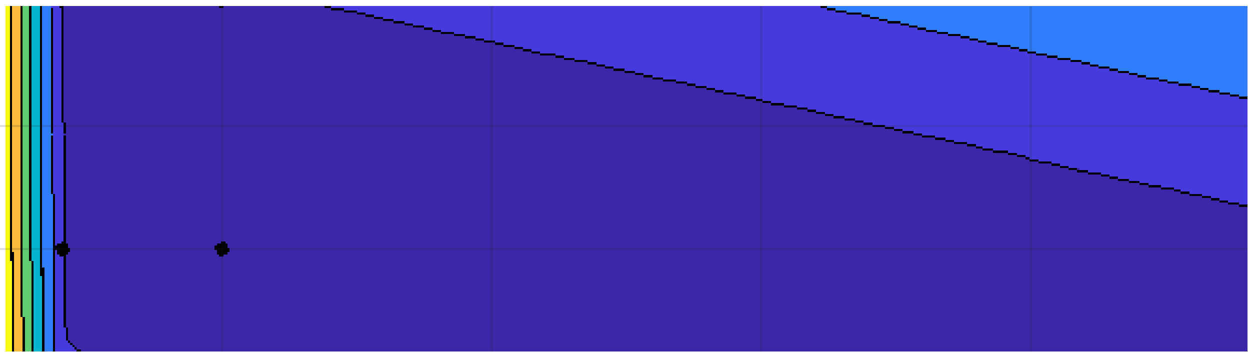


Figure 7.8: Ordering Efficiency: Blocksize 300

Figures 7.7 and 7.8 show that the orderering nodes are in general under utilized. Higher utility can be noticed with transactions that are arriving in lower rates because batch timeout appears to be faster than the rate at which blocks can fill the queue. Given increases in block size, we receive the common observation that the orderer remains under utilized, from this we can also agree that block sizes should be smaller to support high utility.

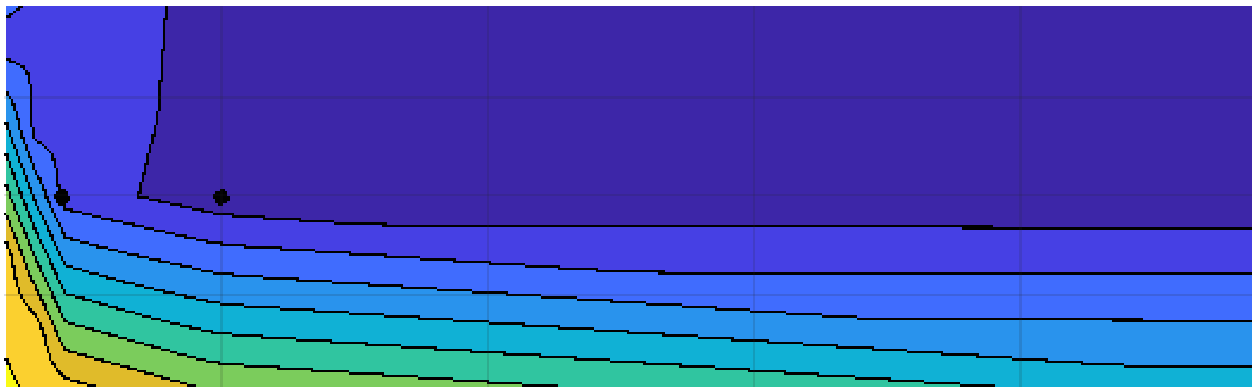


Figure 7.9: Validation and Commitment Efficiency: Blocksize 100

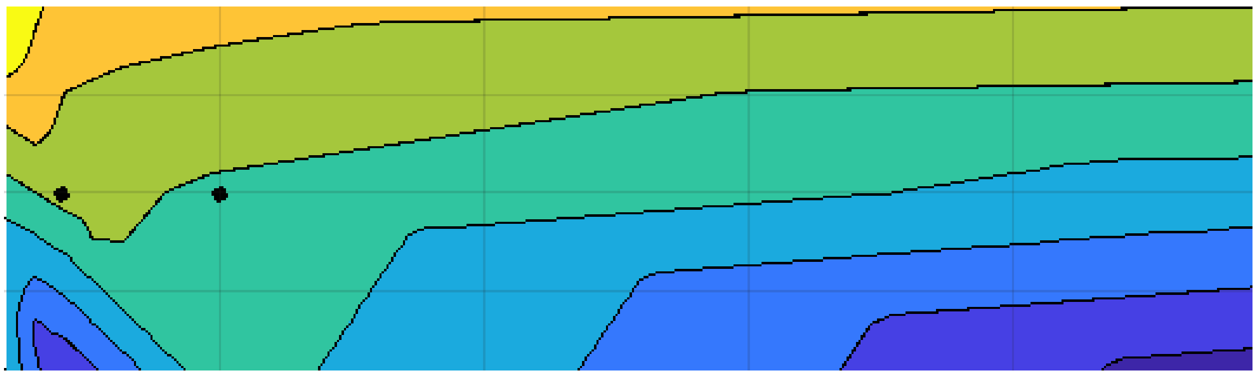


Figure 7.10: Validation and Commitment Efficiency: Blocksize 300

Lastly, Figures 7.9 and 7.10, highlight that performance of the validation and committing nodes fall closely behind performance of the orderer, resulting in sharp degradation in performance as blocks come at lower rates. This agrees with the work of [15] who observes that if the orderer reaches a bottleneck, the committing node will follow in performance.

CHAPTER 8: CONCLUSION

Our results do support the observations of others who mention that the orderer becomes a bottleneck at higher loads. Furthermore, our generation of a hierarchical model give insight into the incremental latency that transactions are subject to as they process through each of these phases. Considerations for future work are to develop more constraints for our model. Matlab's optimization tool looks for the most minimal solution that satisfies our results however the solution is not inherently feasible. This could expose why the endorsement and execution phase generated vastly varied values when only block size was changed. Moreover, our model was run on one machine however this does not consider network communication delay associated with blockchain networks that span across multiple machines. Upon revising this model, we seek to apply it to a wider blockchain network operating on several machines. This would allow for us to broadly characterize performance, security and efficiency of Hyperledger Fabric and other blockchains.

BIBLIOGRAPHY

[1] Byzantine fault tolerance.

[2] Financial cryptography and data security: FC 2016 international workshops, BITCOIN, VOTING, and WAHC, christ church, barbados, february 26, 2016, revised selected papers.

[3] E. Anceaume, A. Pozzo, T. Rieutord, and S. Tucci-Piergiovanni. On finality in blockchains.

[4] Y. Chen and C. Bellavitis. Blockchain disruption and decentralized finance: The rise of decentralized business models. 13:e00151.

[5] H. Hasanova, U.-j. Baek, M.-g. Shin, K. Cho, and M.-S. Kim. A survey on blockchain cybersecurity vulnerabilities and possible countermeasures. 29(2):e2060. 50 citations (Crossref) [2022-07-13] _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nem.2060.

[6] L. Jiang, X. Chang, Y. Liu, M. Jelena, V. B. MiÅ¡iÄ, and t. l. w. o. i. a. n. w. Link to external site. Performance analysis of hyperledger fabric platform: A hierarchical model approach. 13(3):1014–1025. 10 citations (Crossref) [2022-04-16] Num Pages: 1014-1025 Place: Norwell, Netherlands Publisher: Springer Nature B.V.

[7] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. page 9.

[8] Reuters. Massive network outage in canada hits homes, ATMs and 911 emergency lines.

[9] S. Saberi, M. Kouhizadeh, J. Sarkis, and L. Shen. Blockchain technology and its relationships to sustainable supply chain management. 57(7):2117–2135. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207543.2018.1533261.

[10] H. Sukhwani, J. M. MartÃnez, X. Chang, K. S. Trivedi, and A. Rindos. Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric). In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 253–255. 143 citations (Crossref) [2022-07-13].

[11] H. Sukhwani, N. Wang, K. S. Trivedi, and A. Rindos. Performance modeling of hyperledger fabric (permissioned blockchain network). In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pages 1–8. 55 citations (Crossref) [2022-07-13].

[12] F. Tian. A supply chain traceability system for food safety based on HACCP, blockchain & internet of things. In *2017 International Conference on Service Systems and Service Management*, pages 1–6. ISSN: 2161-1904.

[13] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani. MeDShare: Trustless medical data sharing among cloud service providers via blockchain. 5:14757–14767. Conference Name: IEEE Access.

[14] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou. A survey of distributed consensus protocols for blockchain networks. 22(2):1432–1465.

[15] X. Xu, G. Sun, L. Luo, H. Cao, H. Yu, and A. V. Vasilakos. Latency performance modeling and analysis for hyperledger fabric blockchain network. 58(1):102436. 51 citations (Crossref) [2022-07-13].

[16] P. Yuan, K. Zheng, X. Xiong, K. Zhang, and L. Lei. Performance modeling and analysis of a hyperledger-based system using GSPN. 153:117–124. 15 citations (Crossref) [2022-07-13].

[17] G. Zhang, F. Pan, M. Dang'ana, Y. Mao, S. Motepalli, S. Zhang, and H.-A. Jacobsen. Reaching consensus in the byzantine empire: A comprehensive review of BFT consensus algorithms.

[18] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang. An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE International Congress on Big Data (BigData Congress)*, pages 557–564.

VITA

Aura Teasley attends the University of Texas at San Antonio. She acquired a prior degree in Biomedical Engineering however found an enthusiasm for Electrical Engineering, Computer Science and Cyber Security. Aura has pursued several internship opportunities with the Federal Government and Livermore National Laboratory to gain exposure to the demands of critical infrastructure and the need for information security. Her future endeavors are to start the Ph.D program at the University of Texas at San Antonio and encourage enthusiasm around blockchain technology to support novel applications and research.