2020

Permissioned blockchains for real world applicatio

Ismail, Ashiana

Permissioned Blockchains for Real World Applications

by

ASHIANA ISMAIL

A thesis submitted to the Faculty of Graduate Studies

Lakehead University

in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science

Department of Computer Science

Lakehead University, Thunder Bay

April 2020

ABSTRACT

Blockchain technology, even though relatively new, has evolved rapidly in the past 12 years. Bitcoin's underlying technology - the first blockchain - was a public, permissionless, and completely decentralized network that enabled transactions of a cryptocurrency between trustless parties without the need of a trusted intermediary. Blockchain, as a technology, has evolved and expanded far beyond these parameters. It has come so far from its starting point that recent research has developed blockchain platforms that facilitate consortium networks that are private, permissioned and consists of a governance model. Still this technology falls under the same umbrella of "blockchain" or as more people in the field now address it using the broader term - Distributed Ledger Technology. This thesis follows the evolution of the technology from cryptocurrencies to Turing-completeness to consortium networks and beyond. The research takes a look at some real-world use-cases that can be solved with blockchain systems. A distributed network model is proposed that utilizes cryptotokens to create a micro-economy within an educational institution and also facilitate seamless international transfer of money for the convenience of International students in these institutions. The model interconnects and utilizes the features of two modern blockchain platforms. The thesis also presents the partial draft of a private blockchain ledger, developed for the convenience of the Canada Revenue Agency to monitor sales and tax transactions in order to reduce the current GST/HST tax gap of Canada.

ACKNOWLEDGEMENTS

I wish to convey my sincere gratitude to my supervisor, Dr. Ruizhong Wei, who guided me through every step of this journey. I wish to express my appreciation for my family - my mother, sister and brother-in-law, for providing me with moral support from afar. I also want to thank all my close ones here at Thunder Bay, who cheered me on every step of the way. And I especially want to express my never-ending gratitude to Marc Viherkoski, for being my rock since day 1.

TABLE OF CONTENTS

1. INTRODUCTION

1.1. How it all started: Bitcoin

The first successful and still the most popular cryptocurrency is Bitcoin. It was introduced in the year 2008, in a whitepaper titled “Bitcoin: A Peer to Peer Electronic Cash System”[1] that was published under the pseudonym “Satoshi Nakamoto”. Till date, it has not been confirmed whether “Nakamoto” is a single person or a group. Bitcoin can be defined as a decentralized, distributed virtual currency system. But the reason behind Bitcoin’s rapid rise to fame is its underlying technology, known as the blockchain. The blockchain is an immutable distributed ledger of data (in this case transaction data) grouped into blocks and each block is linked to a previous one through hash values, thus forming a chain.

Nakamoto was not the first to suggest the concept of linking hashed values as chains to enforce integrity. Stuart Haber’s and Scott Stornetta’s work, “How to time-stamp a digital document”[2], is recently being recognized as a prototype of Bitcoin’s blockchain. The academic paper was focused on timestamping digital documents in order to verify their authenticity. The work was published in 1991, 17 years prior to Bitcoin’s release. It proposed to use a cryptographic hash function to create the hash of the document and then use a digital signature scheme with a trusted Time-Stamping Service(TSS) to ensure the authenticity of the document. 3 years later, the authors went on to create Surety, which can be considered as the first (physical) application of “blockchaining”. Surety provided time-stamping services that recorded the hashes of digital documents on an immutable ledger. In this case, the immutable ledger was the “Notices & Lost and Found” section of the physical newspaper, “New York Times”. Nakamoto is thought to have been greatly inspired by their work, as three out of the eight papers cited in the Bitcoin whitepaper[1] are written by this duo.

Other works cited in [1] include Hashcash[3], in which Adam Back proposed the idea of spending CPU cycles on hashing to create an easily verifiable token that can be used as Proof of Work. In 1998, Wei Dai proposed two protocols in the paper "B-money"[4], the first of which he describes as "impractical", is extremely similar in concept to today's blockchain distributed ledger. His protocol called for "a synchronous and unjammable anonymous broadcast channel" where all participants stored a local database containing details regarding the accounts and ownership of money[4].

We can certainly see what concepts from these different works inspired different aspects of Bitcoin. But even with these pre-existing ideas, Bitcoin was the first successful public application of it. This is mainly attributed towards it being the first to achieve complete decentralization in a trustless space through the Proof of Work consensus mechanism. It solved the Byzantine General's Problem, the problem of double-spending and was successful in maintaining user anonymity.

1.2. Current Day

Bitcoin and its underlying technology of blockchain became a huge success. Immediately after Bitcoin's release and ensuing success, a number of "altcoin" cryptocurrencies similar to Bitcoin were created. But it wasn't long before the potential of blockchain technology beyond cryptocurrencies was realized. Blockchain, as a distributed immutable ledger has the potential of being the foundation technology for applications throughout various spectrums including, but certainly not limited to, finding prime numbers, managing smart property, creating decentralized self-sovereign identities and analyzing radio waves from space.

As such, currently we are seeing more and more applications of blockchain being shifted from the primary focus that was currency and into what is called Blockchain 2.0[5] which adds the aspect of logic and programmability to Blockchain 1.0. Introduction of smart contracts allowed the building of applications on top of blockchains. Numerous Blockchain platforms that allow

users to create their own blockchain systems and decentralized applications (Dapps) are emerging. Blockchain is also being adapted for enterprise usage and we see more and more corporations(big and small), eager and readily welcoming the new technology in addition to investing in them. To this end, we can see the trajectory of the blockchain features deviating from the public-anonymous route to more of a private-permissioned route with or without a model for governance. This shift is based on the need for practical solutions for enterprise use-cases. As such, we can see a shift in the nomenclature too. As the technology transforms to fit different use-cases, the terminology also evolves with it as well. Distributed Ledger Technology or DLT is used synonymously with blockchain nowadays. Though a relatively new technology, we can see its growth expanding through various sectors as newer applications still continue to emerge.

1.3. This Thesis

This thesis consists of a total of 5 sections. Throughout the document, we provide some technical background, we discuss the major blockchain applications and platforms that have emerged since the advent of Bitcoin. We also analyze two real-world problems where blockchain solutions are immediately applicable and propose blockchain-based models for the same.

Blockchain can be described as a combination of various cryptographic concepts and data structures. Concepts such as mining, forking and consensus also play a major role in what makes blockchain, blockchain. Technical explanations for these concepts are provided in Section 2.

In Swan's work, "Blockchain: Blueprint for a New Economy"[5], she elaborates on Nakamoto's original vision for Bitcoin and the underlying technology. In Section 3, following the model in Swan's work[5], we explore current development in Blockchain 1.0 and 2.0 and what blockchain research has achieved in terms of this model.

In Section 4, we explore the major setback still faced worldwide, regarding international transfer of monetary funds and how it is affecting International students globally despite such advanced technological advancement. We reflect on the same question asked by David Schwartz, CTO of Ripple: when data can be transported so fast from one corner of the Globe to another, why is the transfer of monetary funds so difficult.

A distributed ledger network model is proposed for a University ecosystem with a cryptotoken system to create a micro-economy within the campus. One of the latest and modern Distributed Ledger Technology(DLT) platforms - Corda and the Corda Distributed Ledger Network is used to model the proposed distributed network. The system also facilitates fast, hassle-free transfer of money across international borders through another modern DLT platform focusing on frictionless International payments - Ripple.

Section 5 of this thesis takes a look at the GST/HST tax gap of Canada and the intentional and unintentional nature of non-compliance. This research was started as a collaboration with the Canadian Revenue Agency (CRA) in the hopes of providing a blockchain ledger solution to better track and monitor sales and tax transactions. As discussions proceeded, the CRA was unable to divulge relevant information regarding the data model and current audit processes, due to confidentiality concerns. Therefore, we were unable to proceed further in regards to streamlining the structure of blockchain solution and as a result, the model presented in this section is still in its preliminary stage.

2. TECHNICAL BACKGROUND

2.1. Cryptographic Concepts

2.1.1. Cryptographic Hash functions

A hash function is a mathematical function that takes an input of arbitrary length and always produces a relatively unique fixed-length output. The output of a hash function is called the *message digest* or simply *digest* or *hash*. The smallest change in the input of a hash function will drastically change the digest. Hash functions are irreversible by design. This property is known as Preimage resistance. It is infeasible to find the valid input of a Hash function even if you have the output digest.

A hash function must also be efficiently computable i.e., for an input string of length n, the computing time of the Hash function should be in the order of O(n). Technical description of two kinds of preimage resistance follows:

1) Preimage Resistance — for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x′ such that h(x′) = y when given any y for which a corresponding input is not known.[6]

2) 2nd Preimage Resistance — it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a 2nd-preimage x′≠ x, such that h(x) = h(x′).[6]

Cryptographic hash functions are important cryptographic primitives that have an important role in modern cryptography. They are often used for data integrity and message authentication. In addition to the properties of a conventional hash function, they often have other properties such as Collision Resistance, Hiding and Puzzle Friendliness making them cryptographically secure. These additional properties are important for the functioning of a cryptocurrency.

Collision resistance: Since a hash function takes an input of an arbitrary size and produces an output of a specific length, the input space is larger than the output space. Therefore, there can be a chance of collision. We are not claiming the absence of collision. In fact, collisions do exist because the hash function is a mapping from a much larger space to a smaller space.

A hash function H is said to be collision-resistant if it is infeasible to find two values, x and y, such that $x \neq y$, yet $H(x) = H(y)$.

Hiding: A hash function H is hiding if: when a secret value, r is chosen from a probability distribution that has high minentropy, then given H(r ‖ x) it is infeasible to find x.

Puzzle friendliness: A hash function H is said to be puzzle-friendly if for every possible n-bit output value y, if k is chosen from a distribution with high minentropy, then it is infeasible to find x such that H(k ‖ x) = y in time significantly less than 2^n.

“High min-entropy” means that the distribution is “very spread out” so that no particular value is chosen with more than negligible probability.

There are a number of hashing algorithms currently being used. SHA256 is a very popular hashing algorithm used in many current blockchain technologies including Bitcoin. It has all the properties stated above. It stands for Secure Hash Algorithm and gives an output of size 256 bits. Other cryptocurrencies such as Ethereum use Keccak-256 (based on SHA-3), while Litecoin uses the *scrypt* hashing algorithm[7]. *Scrypt* is a password-based key derivation function. Password-based derivation functions derive one or more secret keys from a secret value or password[7]. Other examples are MD5 crypt, SHA2CRYPT and BCRYPT (Blowfish-based).

2.1.2. Asymmetric Key Cryptography & Digital Signatures

Asymmetric Key Cryptography uses a pair of keys that are mathematically related. The public key is made public to everyone in the system and the private key is kept a secret. Even though they are mathematically related, the private key cannot be derived from knowledge of the public key. The key pair can be used for a number of functions:

The public key is used to derive addresses for the user. Multiple addresses can be derived from a public key, making the concept of pseudonymity possible. Pseudonymity means the user is anonymous but the accounts are not.

Private keys are used to digitally sign a transaction. The corresponding public key can be used to verify the signature. Digital signatures are often used for message authentication - verify that the message was created and signed by the owner of the private key; message integrity - verify that the message was not altered after signing; and non-repudiation - once a message has been signed, the signer cannot deny the fact.

2.1.3. Elliptic Curve Digital Signature Algorithm (ECDSA)

Elliptic Curve Cryptography(ECC) is an advanced encryption technique based on the structure of elliptic curves over a finite field. ECC is considered more advanced because it can be used to generate faster and smaller keys. The keys are generated through the equation of an elliptic curve rather than large prime numbers as in RSA. And this generation is as easy and fast as the prime number generation but provides a lot more security.

2.2. Data Structures

2.2.1. Blockchain

In the blockchain or cryptocurrency scenario, a *transaction* is the record of a transfer of assets within the system. A *ledger* is a collection of transactions. It is the digital version of the conventional ledger that is traditionally used to record transactions. The blockchain ledger is made of blocks that contain multiple transactions. In the case of a blockchain ledger, we mitigate some of the issues its conventional counterpart might have. For example, since the blockchain ledger is distributed to all nodes on the network, the blockchain environment removes the need for a trusted 3rd party to act as middlemen, thus enabling complete strangers to transact directly in a trustless environment without worrying about security. Also, a blockchain ledger is immutable, i.e., once information has been recorded onto it, it cannot be altered in any way. Any changes needed to a past transaction have to be corrected through subsequent transactions.

Blocks in a blockchain are connected in a way similar to that of a linked list. In a linked list, the nodes are linked using pointers that usually point to the address of the information. However, blockchain uses hash pointers which point to the location of the information along with the hash value of the information. Using this hash value it can be confirmed that the information hasn't been tampered with, and therefore is secure.

2.2.2. Merkle Trees

A Merkle tree is another data structure that is built using hash pointers. It is named after its inventor Ralph Merkle. It can be described as a Binary Tree built with Hash pointers. Merkle trees are created by repeatedly hashing pairs of nodes until there is only one hash left called the *merkle tree root hash* or simply *root hash*.

Merkle invented the protocol Merkle Signature Scheme (MSS) for better public key management. This technique has the great advantage of eliminating large storage requirements,

especially in One-Time Signature Schemes. But Merkle emphasizes in his paper that this protocol "[...] can also be used to solve authentication problems which do not involve digital signatures: that it is being used to generate tree signatures in this paper should not prejudice the reader into thinking that is its only application"[8].

A Merkle tree is a well-known cryptographic scheme that is commonly used to provide proofs of inclusion and data integrity. Merkle trees are widely used in peer-to-peer networks, blockchain systems and git. In Corda (a fairly new blockchain platform) for example, merkle trees are used for what is termed as transaction tear-offs or filtered transactions. In the concept, nested Merkle trees are used to "tear-off" any parts of the transaction that an oracle or a notary does not need to see before presenting it to them for signing[9]. Thus, it provides a level of abstraction for ledger data, while being able to access the services from outside the network.

2.3. Decentralization Techniques

What Nakamoto actually contributed was a technique to come to consensus in a completely decentralized, distributed network solving the Byzantine General's problem. Thus, the Proof of Work concept is also referred to as Nakamoto Consensus. Various other consensus models, designed for both permissionless and permissioned blockchain systems, have emerged since the Bitcoin whitepaper[1] release in 2008. This section explores the decentralization techniques of mining, consensus and forking.

2.3.1. Mining and Consensus Models

A major concept in Bitcoin and many other cryptocurrencies is mining. Mining is a critical component that sustains trustless blockchain systems that are completely decentralized and permissionless. Mining is the process in which miners validate the current transactions in the transaction pool, create blocks out of them and append them to the blockchain.
In the Bitcoin blockchain, the process is as follows:

- A miner creates a new block with transactions from the transaction pool.

- The miner performs a resource-intensive task and produces proof that the work has been done, usually in the form of a Nonce.
- The new block is distributed throughout the network.
- The block is validated by other miners in the network. The proof can be verified easily as compared to the creation of it.
- Once the block is validated by 51% of the nodes in the network, it is permanently added to the blockchain.
- The miner who created the block receives the predetermined reward, if any.

There is the possibility that a malicious user can try to create multiple nodes and try to validate an invalid transaction. The resource-intensive task is introduced to prevent this from happening. Creating a nonce requires a considerable amount of memory and/or electricity. It is a computational puzzle that is difficult to solve but can be validated easily. Thus, providing a correct nonce is providing proof that you have actually done the work. Therefore, this concept is appropriately coined **Proof of Work (PoW)**. This is the method by which the completely decentralized network of untrusting parties comes into consensus about which block to be added to the current blockchain. The PoW consensus model is designed for the case where there is little to no trust amongst users of the system. It becomes meaningless in scenarios with some level of trust between parties. Various other consensus models have been developed to fit the criteria of different blockchain systems. We explore a few below.

Proof of Stake Consensus Model

In the Proof-of-Stake(PoS) consensus model, there is no computationally intensive puzzle for the miner to solve, instead, the system depends on how much stake the miner has in the blockchain system. Stake is defined as the amount of cryptocurrency a user owns, which is considered proportional to the stake the user has invested in the system, incentivizing the user to play fair. The stake is locked into the blockchain system either via a special transaction type or by sending it to a specific address. The amount of staked cryptocurrency is usually cannot be spent until the next block is created[10].

The miner that is chosen to add the new block is also dependent on how much stake the user has in the system. The method that determines which miner is chosen to add the next block to the system can vary from random selection to multi-round voting, to a coin-aging system. In a multi-round voting system, a group of stakeholders go through multiple rounds of voting before the block publisher is chosen. In the coin-aging system, the user's staked currency has an "age" property that gets spent when the stakeholder publishes a block. The user needs to wait the time period again for their currency to "age" before they can publish again[10]. Thus, this prevents users with higher stake to dominate by slowing them down. Regardless of the ordering process, users with higher stakes are guaranteed to publish more blocks. [10]

Proof of Elapsed Time

Hyperledger Sawtooth introduced a new "lottery-based Nakamoto Consensus"[11] called Proof of Elapsed Time(PoET). This model doesn't use electricity as the scarce resource but instead, it uses time. The PoET consensus elects a leader on the basis of a guaranteed wait time provided by a Trusted Execution Environment(TEE). Every validator is provided a random amount of wait time by the TEE and the validator with the shortest wait time, after waiting that period of time is elected as leader. A function such as "CreateTimer" creates a timer and is guaranteed to have been created by the TEE[12]. And another function such as "CheckTimer" can be used to verify that the timer was indeed created by the TEE[12].

The leader election process of PoET meets the criteria of a good lottery process[11]. The election process is fairly and randomly distributed across the entire validator population and verification of the authenticity of the election process is simple too. This process eliminates the need for costly investment of power and specialized hardware in mining. The low cost to participate in the election will tend to result in large validator populations thus increasing the robustness of the algorithm.[11]

Current Sawtooth implementation is built on the TEE provided by Intel's Software Guard Extension (SGX)[11].

Proof of Authority

Proof of Authority (PoA) consensus is a form of Proof of Stake in which your identity or reputation is at the stake. It is a protocol designed for permissioned networks with some kind of governance model. The protocol requires all the validators to be known or verified by some trusted authorities such as a public notary. The eligibility must be relatively difficult to obtain. Therefore the parties who put their time and work into obtaining it are vested in the overall good of the whole system. It should also be easily verifiable that the validator has obtained authorization from the appointed notary service.[13]

Since there is an established authority present, it is more of a centralized consensus model. But this makes it ideal for permissioned consortium networks. There are a number of PoA consensus models currently being used, including IBFT (Istanbul Byzantine Fault Tolerance), IBFT 2.0 and Clique to name a few.

2.3.2. Forking

Blockchains are immutable and distributed across the globe and for these same reasons that make blockchain what it is, updating the underlying technology can become difficult. Every technology needs to be updated at some point and Bitcoin style early blockchains left little to no provisions for such updations. Therefore, these kinds of changes called for some hard decisions that had the potential to alter the very structure and/or trajectory of the blockchain network. Such changes are called *forks* in the blockchain terminology. There are two kinds of forks in blockchain architecture: soft forks and hard forks.

Soft Fork - A *soft fork* gives the users on the blockchain a choice to adopt the change or not, without much of a consequence. The nodes can choose to update to the latest version. The

non-updated nodes are still able to function without adopting the update. It only requires a majority of the nodes on the blockchain to upgrade to enforce a *soft fork* change.
Example: Bitcoin's soft fork.

Hard Fork - A *hard fork* completely prevents the nodes who don't adopt the change from accessing the updated blockchain. When a hard fork is formed, the majority who accepted the changes continues on the updated branch of the blockchain and the others continue on the same blockchain.
Example: Ethereum's Hard Fork to Ethereum Classic.

3. EVOLUTION OF BLOCKCHAIN TECHNOLOGY

Bitcoin started in 2009 by Nakamoto set the stage up for cryptocurrencies and blockchain technology. But cryptocurrency and blockchain technology has far advanced beyond bitcoin. Currently, there are numerous cryptocurrencies and blockchain technology platforms evolving to contribute to the various use-cases of different sectors. Even though a decentralized cryptocurrency was the initial use-case and cause of the recognition and popularity of blockchain or Distributed Ledger Technology(DLT), the technology is growing rapidly to accommodate use-cases ranging from tracking farm produce to better management of healthcare data to platforms facilitating seamless and instantaneous global payments.

According to Swan[5], two out of the three steps of Nakamoto's vision has been implemented in Bitcoin 1.0. This includes the completely decentralized distributed ledger or the blockchain and the Bitcoin protocol which facilitates transactions between trustless parties without any third-party intermediaries. The third step in Nakamoto's vision - Turing completeness - was required for the evolution of Blockchain 2.0. That milestone was first achieved by Ethereum, facilitating more complex transactions over blockchain networks such as the transfer of complex assets like smart property and smart contracts.

In this thesis, we follow a structure similar to Swan's model with Bitcoin and Altcoins categorized into Blockchain 1.0. Altcoins are defined as cryptocurrencies very similar to Bitcoin and are often described as having "little to no innovation"[17]. Therefore, in the following section of 3.1, we list a few of the "altcoins" of Bitcoin and highlight their innovative parts and major differences from Bitcoin. In section 3.2, we discuss the introduction of smart contracts and Dapps into the blockchain system. Ethereum, currently the second most popular blockchain system, that was the first Turing-complete blockchain ecosystem and the Hyperledger Greenhouse consisting of its various Blockchain projects are discussed in Blockchain 2.0. Section 3.3 portrays the upcoming, more real-world dependable Blockchain 3.0 that dwells into

applications for the banking and financial sector, now popularly known as FinTech. The section includes the new debt-based cryptocurrency XRP, released by Ripple and R3's Corda platform.

3.1. Blockchain 1.0: Altcoins

In this section, we cover a few of the "altcoins" of Bitcoin that rapidly evolved after Bitcoin was released.

3.1.1. Omni (MSC)

Mastercoin, later renamed as Omni in 2015, was the first altcoin to venture into Bitcoin 2.0. The concept was proposed by J.R Willett in the whitepaper titled "The Second Bitcoin Whitepaper"[14], that came out in 2012 to facilitate building new currency layers on top of the existing bitcoin protocol. It was proposed that without changing anything about Bitcoin, it could be used as a foundation protocol layer to build higher-level protocols on top of it. Thus enabling the creation of smart contracts and user-defined currencies on top of the Bitcoin blockchain.

Today, Omni is a platform for creating and trading custom digital assets and currencies mostly in the form of tokens. And the Omni Layer protocol is designed to exist between the original bitcoin protocol and user-defined currencies. It allows users to generate tokens representing any kind of asset and use them for any kind of transactions including trading, crowdfunding and making bets[15].

3.1.2. Litecoin (LTC)

Popularly known as the silver cryptocurrency to Bitcoin's gold, was launched in 2011. The underlying technology is so similar to that of bitcoin, not even a whitepaper on it exists. Even the GitHub repository of Litecoin states it is forked from Bitcoin.

But Litecoin has some significant differences from Bitcoin. The major one being its hashing algorithm. Litecoin uses scrypt as opposed to SHA256 of Bitcoin. The scrypt hash function uses

SHA256 as a subroutine but uses much less power. Normal computers and CPUs are able to carry out Litecoin mining. Thus, making it possible for a larger demographic to participate in the mining. This gives the potential to Litecoin to have a mining process that is more decentralized than Bitcoin's, as there is currently a growing concern about Bitcoin mining becoming more and more centralized towards areas with cheaper electricity.

Litecoin is currently under the process of integrating the MimbleWimble (MW) privacy protocol. MimbleWimble uses Elliptic Curve Cryptography and far outperforms Bitcoin's privacy protocol[16]. Even though Bitcoin makes efforts to ensure the anonymity of users transacting on the blockchain, it is still possible to filter out the specifics about the identity of who is involved in a transaction.

Scrypt also makes the mining process 4 times faster for Litecoin, generating a block every 2.5 minutes on the average. Litecoin currently awards its miners 12.5 new litecoins per block, an amount which gets halved roughly every 4 years. Litecoin is also estimated to produce 4 times the currency units, compared to Bitcoin, in its lifetime - a total of 84 million LTC[17].

3.1.3. Peercoin (PPC)

Peercoin was the first cryptocurrency to implement Proof-of-Stake instead of Proof-of-Work. It was launched in 2012. Due to large-scale mining operations, it can be said that PoW mining in the bitcoin network is becoming more and more centralized. Sunny King and Scott Nadal proposed the new consensus model, and as a result, the cryptocurrency, as "a response to increasing doubts on Bitcoin's PoW consensus model with regards to its increasing centralization and conflict of interests between miners"[18]. Peercoin uses Proof-of-Stake model, in implementing the security of the blockchain network and also in the minting process. PoW is actually used at the beginning of the minting process of the coin but is later phased out. The authors claim that Peercoin can be described as a "*long-term energy-efficient*"[19] cryptocurrency as the energy consumption on proof-of-work is allowed to approach zero.

A duplicate-stake protocol is designed to defend against denial-of-service attacks instigated by an attacker using the same Proof-of-Stake to generate multiple blocks. Every node on the network keeps track of the (kernel, timestamp) pair of all coinstake transactions it has seen. And if it receives a block with a duplicate pair, the block is ignored[19].

Even though Peercoin was created to mitigate centralization of Bitcoin it still employs a centrally broadcasted checkpointing mechanism, to solidify the blockchain history, that is not completely distributed. But this is considered acceptable for now, as a practical distributed counterpart is not available presently. The broadcasted checkpointing ensures all nodes can agree on all transactions prior to one month[19]. This also helps in the coinstake kernel verification carried out by a node as part of the duplicate-stake protocol.

The major advantage of this cryptocurrency is of course how it is not dependent on energy consumption, immensely saving on energy and cost overhead.

3.1.4. Primecoin (XPM)

Primecoin[20] is the first cryptocurrency with an additional scientific value derived from the proof-of-work energy consumption. Sunny King, a pseudonym used by one of the founders of Peercoin, proposed the innovative idea of using the computational energy spent whilst mining to derive prime numbers instead of letting it go to complete waste, as is in the case of traditional Bitcoin mining.

The infinite existence of prime numbers was known as early as 200BC. Since then mathematicians have been researching them, especially about their infinite distribution and discovering patterns to find more of them. As a tribute to Bernhard Riemann, who was a pioneer mathematician in the field of Prime numbers, Primecoin is represented by the currency symbol

Ψ, which is the greek letter psi. The same symbol represents the Riemann zeta function which played a crucial role in proving the *prime numbers theorem*.

Primecoin uses 3 kinds of prime chains for their proof of work model. These are the Cunningham chain of the first kind, Cunningham chain of the second kind, and bi-twin chain. These chains were selected because verification of these chains remains efficient even when the size of the prime reaches record heights. In fact, since its launch, Primecoin has found 27 World Record primes through its mining process. And, Primecoin processes payment transactions 10 times faster than the bitcoin network, with a block added to the chain every minute.

Another notable feature of Primecoin is its smooth difficulty adjustment. It adjusts its difficulty slightly every block[21], which is incredible considering Bitcoin only adjusts its difficulty every 2016 blocks in order to match the target rate of 1 block every 10 minutes. This ensures that even if computation power to find the primes suddenly doubles, through technological innovation or otherwise, within the second block the difficulty would be adjusted accordingly.

3.1.5. Dash (DASH)

Dash[22] introduces the concept of incentivizing full nodes called master nodes. Bitcoin full node numbers seem to be decreasing a lot and as a result block propagation time has also gone upward of 40 seconds. The major reason behind this is believed to be the absence of any kind of incentive to run a full node that takes quite a bit of computational power and memory.

Dash runs a secondary network of master nodes called masternode network and these nodes are incentivized with the masternode rewards program. In order to be a masternode, a node needs to have control of a 1,000 DASH currency. The amount is sent to a specific address in a wallet, which will activate the node to a masternode. A secondary private key is generated. While operating as a masternode, the 1,000 DASH is locked as collateral but is never forfeit. This system also helps in reducing the volatility of the currency[22].

The Masternode Rewards Program awards 45% of a block as a reward to the masternodes. The percentage of the reward is fixed and therefore the reward varies according to the network. The reward also depends on how many masternodes are currently active on the network. The requirement of 1,000 DASH also makes it pretty difficult for someone to take control of over 50% of the network for a Sybil attack.

Dash also creates trustless quorums to perform special tasks, by choosing a number of pseudorandom masternodes. Two major features Dash implements are PrivateSend and InstantSend. PrivateSend increases the privacy of the users through a decentralized mixing process. Users submit amounts of Dash currency in standard denominations to a mixing session. At least three users are required for a mixing session. All inputs are mixed together and sent back to the users. It takes several mixing sessions or "rounds" to thoroughly anonymize transaction information. This makes it difficult to trace user information through backtracking transactions.

The InstantSend feature utilizes trustless quorums of masternodes to send and receive transactions instantaneously. This is done through a concept called transaction locking. The submitted transaction for InstantSend is broadcasted throughout the networks and once it reaches a voted quorum of masternodes the transaction is validated, and the inputs are locked so they cannot be double spend[23]. This information is broadcast to the entire network and the transaction is executed.

3.2. Blockchain 2.0: Smart Contracts & Dapps

Blockchain 2.0, the next big level in blockchain evolution is a broad space that is still in development and classifications are still emerging. Blockchain 2.0 goes beyond the decentralization of currency to the decentralization of more complicated assets in different markets. According to Swan's[5] distinction, Blockchain 2.0 is a broad space that can include a range of concepts such as Bitcoin 2.0, Bitcoin 2.0 protocols, smart contracts, smart property, Dapps (decentralized applications), DAOs (decentralized autonomous organizations), and DACs

(Decentralized Autonomous Corporations). Blockchain 2.0 is considered a major technological leap due to the achievement of a Turing-complete system through the introduction of smart contracts.

Smarts contracts provide a way to allow logic to be added to blockchain transactions. They are the computational muscle of a blockchain. It allows blockchain transactions to go beyond just simple currency transactions and have more extensive instructions embedded into them. In a traditional contract each party must trust the other party to fulfil its side of the obligation. Smart contracts feature the same kind of agreement, but they remove the need for any type of trust between parties. This is because a smart contract is both defined by the code and executed by the code.

Swan[5] cites the major features that make a smart contract distinct to be autonomy, self-sufficiency, and decentralization. Autonomy is the ability of the smart contract to run autonomously on the blockchain network once it has been launched. They are also required to be self-sufficient so they have the ability to obtain resources and spend them accordingly to sustain themselves. Also, smart contracts should not be tied to any single centralized entity. They need to be distributed across the whole network and execute accordingly when triggered.

In this section, we explore Ethereum, the first Turing Complete blockchain system that brought about the evolution of Blockchain 2.0. We also look at the Blockchain incubator - Hyperledger, managed under the Linux Foundation and the various distributed ledger platforms developing under its “umbrella”.

3.2.1. **Ethereum (ETH) -** A Turing-complete Virtual Machine

The Ethereum blockchain was the first to create a turing-complete deterministic blockchain environment. The Ethereum Virtual Machine, or EVM, is a sandboxed virtual environment emulating a physical computing machine completely isolated from the network or any process on

the host node. Every ethereum node runs an instance of EVM and allows for anyone to write their own smart contracts and decentralized apps (Dapps) that run on top of the ethereum blockchain. Ethereum has created its own high-level programming language called Solidity in which the smart contracts are typically written. The smart contracts are compiled to EVM bytecode in order for the EVM to run them.

Being turing-complete means that the EVM code can encode any computation that can be conceivably carried out, especially loops. EVM code allows looping in two ways. First, there is the JUMP instruction that allows the program to jump back to a previous spot in the code, and also a JUMPI instruction that allows conditional jumping. Second, contracts can call other contracts, therefore allowing looping through recursion[24]. The major deterrent about involving loops is the possibility of a code inciting infinite loops. As this provides an opportunity for a malicious user to manipulate and force nodes into infinite loops thus creating a Denial-Of-Service attack.

Ethereum averts this possibility by charging a fee for each software instruction run by the code. Ethereum has its own native cryptocurrency called ether and this is the “gas” that is used to pay the transaction fees. The concept can be split into 3 parts: the gas, the gas price and the gas limit. Gas is the measure of how much it costs for each computation, usually 1 per computation, more if it’s a complicated computation. Gas Price is the price a party is willing to pay for the validation of the transaction. It is usually paid in terms of Wei, which is the smallest unit of the ether currency. And the Gas Limit limits the number of computations that can be executed.

By using the Gas mechanism, two major issues are solved: A miner still receives payment for the work they have done even if the transaction execution fails. An execution cannot run longer than the pre-paid amount would allow. Instead of looping indefinitely, the execution would run until it runs out of gas[25]. Therefore we can say that the EVM code is bound by gas to avoid infinite loops, and hence it is technically a quasi-turing complete machine.

3.2.2. **Hyperledger** - The Greenhouse for Enterprise Blockchain

The Hyperledger Project, managed under the Linux Foundation, began in 2015 as an incubator for enterprise-grade DLT platforms. The Hyperledger community envisions the future of blockchain to involve modular, open-source platforms that are easy to use and available to everyone. In this section, we explore the different blockchain frameworks that are growing under its "greenhouse".

Hyperledger Fabric - Enterprise-grade DLT

Hyperledger Fabric is the first open-source enterprise-grade distributed ledger platform that enabled the use of standard programming languages[26]. Hyperledger Fabric, or Fabric, does not have a domain-specific language(DSL) and smart contracts (called "chaincode" in Fabric) can be written in general programming languages such as Java and Node.js. It works on a permissioned network and is governed by an open governance model. Open governance means that technical decisions are made by a group of community-elected developers drawn from a pool of active participants.

Fabric also introduced pluggable consensus protocols that eliminate the need for a mining process. This is more appropriate for the enterprise context where the network is permissioned and operated by a trusted authority. Participants are known or at the least, traceable. So the risk of a participant intentionally introducing malicious code is diminished. Therefore, in this context, the use of a fully Byzantine Fault Tolerant consensus might be considered unnecessary or even an excessive drag on performance and throughput[26]. Instead, pluggable ordering services implement consensus protocols such as the crash fault-tolerant (CFT) consensus protocol.

Hyperledger Fabric has a highly modular structure providing resiliency, flexibility, scalability, and confidentiality[27]. At a high-level, Fabric can consist of different modules implementing the different functionalities of a standard permissioned blockchain system. These can include:

- A pluggable consensus service called *ordering service*, that establishes consensus on the order of the transactions and then broadcasts the block to peers.
- A pluggable membership service, responsible for handling the identities of participants in the network.
- Smart contracts or "chaincode"
- An optional peer-to-peer gossip protocol to send ordered blocks to peers.
- Provision to support a variety of existing DBMSs[28].

Hyperledger Fabric also replaces the, till then followed, **validate-order-execute** structure of consensus protocols with a new **execute-order-validate** approach. In the former architecture, the consensus protocol **validates** transactions, **orders** them into blocks and distributes it throughout the network. And as the final step, each peer **executes** every transaction sequentially. [26]

In Fabric's model, the transaction is **executed** initially and checked for correctness. The transaction is then "*endorsed"*. The *endorsed* transactions are **ordered** via a consensus protocol and as the final step, the transactions are **validated** against an application-specific endorsement policy. [26]

The application-specific endorsement policy specifies how many of the peer nodes need to vouch or *endorse* for the correct execution of a transaction. Thus, each transaction need only be executed and endorsed by a subset of the peer nodes and not the whole network. This allows for parallel and faster execution of transactions[26].

These unique features coupled with its modular structure makes Fabric a lot powerful and flexible to configure it for various different enterprise-level use cases[26].

Hyperledger Sawtooth

Originally created by Intel, and later seeded as the second project incubated by Hyperledger, Hyperledger Sawtooth is a modular platform for building, deploying, and running distributed

ledgers[29]. Sawtooth is highly modular and gives enterprises the freedom and flexibility to select their permissioning, transaction rules and consensus algorithms as per their business requirements[30]. Another major feature of Sawtooth is that it supports both permissioned and permissionless blockchain deployments. There are a number of other technological innovations pertaining to Sawtooth.

Sawtooth makes a complete separation of its application level from its core system level. This is the basis of most of the unique features of Sawtooth. This abstraction allows users to write smart contracts in almost any programming language. Transaction business logic is separated into a concept called Transaction Families. A transaction family includes these components:

- A transaction processor that defines the business logic
- A data model to record and store data
- A client to handle the client logic

Sawtooth introduced the innovative Proof of Elapsed Time (PoET) consensus model that redefined the meaning of "efficiency" since Nakamoto's PoW model. PoET is a lottery-based consensus algorithm similar to PoW but without the drawback of high energy consumption. It elects a leader based on a guaranteed wait time provided through a "trusted execution environment" or TEE[11]. The current implementation of Sawtooth depends on Intel SGX to act as the TEE and provide time functions. It is highly scalable and capable of supporting large network populations[12]. Sawtooth provides pluggable consensus for a number of other consensus models too, including Sawtooth Raft which provides Crash Fault Tolerance consensus for small networks and a simple random leader algorithm called Devmode. Sawtooth has the unique feature called Dynamic Consensus which allows you to change the consensus algorithm of a running blockchain through executing one or two transactions. This is also made possible because of Sawtooth's abstraction of core concepts from transaction semantics.

Parallel transaction execution is another feature that Sawtooth has which is a difficult feat for blockchains. A major requirement for blockchains is to execute transactions across the network.

Sawtooth accomplishes this by using an advanced parallel scheduler that splits blocks into parallel flows[12]. This allows for faster block processing compared to other traditional blockchains.

Sawtooth also has a Sawtooth-Ethereum integration project called Seth, which allows EVM smart contracts to be run on the Sawtooth blockchain. Sawtooth worked with another Hyperledger project, Hyperledger Burrow, and borrowed their EVM implementation called Burrow EVM to integrate Ethereum into the project.

Hyperledger Iroha - Mobile Applications and IoT

Iroha was the third distributed ledger platform to join the Hyperledger family in October 2016. Hyperledger Iroha is designed with an emphasis on mobile application development and IoT applications. It was originally developed by Soramitsu in Japan. Because of its Japanese origin, it follows the Japanese principle of *kaizen* which is to eliminate excessiveness. It features simple deployment and maintenance and a variety of libraries for developers[31].

Iroha is a permissioned blockchain platform that is ideal for digital assets and identity management. It also introduced two new consensus algorithms, the first, a chain-based fully Byzantine Fault-Tolerant consensus algorithm, called Sumeragi. And the second, a Crash Fault Tolerant consensus algorithm called YAC, which stands for Yet Another Consensus. YAC has high-performance and allows for the finality of transactions with low latency[31]. Iroha also claims to be the only ledger that has this level of robustness in its permissioning system, which allows setting permissions for commands, queries, and also in the process of joining the network[31].

Hyperledger Burrow - Permissionable smart contract machine

Hyperledger Burrow was the fourth distributed ledger platform that joined Hyperledger in April 2017. Formerly ErisDB, Burrow is a permissioned, Ethereum smart contract enabled blockchain platform originally developed and proposed to Hyperledger by Monax (formerly Eris).[29]

Burrow was built to the specifications of Ethereum's EVM. It executes Ethereum's smart contracts on its own custom permissioned virtual machine[32]. Currently, it supports both EVM and WASM (WebAssembly) based smart contracts. Burrow implements Proof-of-Stake based BFT consensus through the Tendermint consensus engine.

Burrow was ideally built for "public" permissioned networks, where the permissioned networks are "visible" to the public, with a dynamic permissions model where validators can be added only on invite. But it can also be configured for permissionless public networks like Ethereum or even private networks. Burrow is built as a strongly deterministic smart contract machine that can run any smart contract code compiled by an EVM language compiler. As a node, it has three main components: the consensus engine, the permissioned Ethereum virtual machine and the API gateway[32].

Hyperledger Indy - Decentralized Identity

The Indy project was contributed to the Hyperledger greenhouse by the Sovrin Foundation. Hyperledger Indy is a distributed ledger platform with a narrow focus on decentralized self-sovereign identity. Indy aims to eliminate the need for users to have multiple login credentials for different platforms. Thus, reducing the chances of users leaving digital traces of their confidential data behind. Indy wishes to leave identity control in the hands of the users, whether they are individuals, institutions or IoT.

Indy provides a set of tools and libraries for creating independent, decentralized and digital identities that are rooted on blockchains or distributed ledgers[33]. These universal identities are

designed to be interoperable across all administrative domains, applications, and any organizational silo[29]. Privacy of these identities is guaranteed as private data is not recorded on a distributed ledger of any kind but exchanged over peer to peer encrypted connections. A unique ID is created for each relationship and is designed to be correlation-resistant.

Indy uses a consensus algorithm called Plenum which is a Byzantine Fault Tolerant Protocol based on RBFT. It is specially crafted for use in an identity system.

Indy is also currently working on the concept of Verifiable Claims, which is an interoperable format that combines basic credentials such as birth certificates, driver's licenses and passports, and use zero-knowledge proofs to enable selective disclosure of only the data that is required at any particular context[29]. This concept is currently in the standardization pipeline at the World Wide Web Consortium (W3C).

Hyperledger Besu

The latest addition to the Hyperledger Framework, Hyperledger Besu, formerly known as Pantheon, was added to the Hyperledger Greenhouse in August 2019. Besu is an open-source Java-based Ethereum client developed under the Apache 2.0 license. It is the first Hyperledger project that can operate on a public blockchain. It runs on Ethereum public and private networks, and also on Ethereum's test networks such as Rinkeby, Ropsten, and Görli.

Besu supports smart contract and Dapp development, and deployment on Ethereum blockchain networks. It can be used to develop enterprise applications that require secure, high-performance transaction processing in a private network[34]. Besu also provides tools to monitor Ethereum nodes and network performance.

Besu implements a number of consensus protocols including Ethereum's PoW - Ethash and Proof of Authority protocols such as IBFT 2.0 and Clique[35].

3.3. Blockchain 2.0: FinTech

Coming to blockchain applications in the Business and Finance industry, we move to a broader spectrum of DLT to deal with real-world use-cases and provide real-world solutions. Real-world enterprise use-cases require permissioned systems as most of these use cases still run within a corporate framework that will have a form of authority and some level of trust. This trust is helpful to increase efficiency in the consensus process as it eliminates the mining process which is the resource swallower of initial blockchain systems.

At the heart of a blockchain network is a distributed ledger that records all the transactions that take place on the network. Thus, as the technology expands, the nomenclature also keeps evolving accordingly. We lose the tight constraints around "traditional" blockchain images so it has the space to spread to accommodate use-cases across various sectors.

3.3.1. Ripple (XRP)

Ripple is a company that focuses on facilitating frictionless International payments with distributed ledger technology. They envision the *Internet of Value* which is similar to the Internet but transacts packets of value instead of data. At the core of this design is the InterLedger Protocol or ILP which is once again inspired by the Internet protocol stack's IP protocol.

They created the XRP Ledger in 2011 which is not strictly a bitcoin style "blockchain ledger", but more of a "global consensus ledger". XRP is the native currency of the XRP Ledger. It is a debt-based cryptocurrency that enables almost instantaneous international transfers. Ripple describes XRP as the "ultimate bridge currency". It is described so, because of its ability to transfer across borders, hopping through different currencies coined by Ripple as "rippling". Ripple achieves this by collaborating with banks and other financial institutions that will hold large amounts of the Ripple currency, XRP. Currently, there are over 100 different banks in this roster, including highly credible ones such as Bank of America and JP Morgan. This technology

is revolutionizing the current banking ecosystem by helping existing traditional banks to improve the efficiency in their own transactions and cutting down their operational costs while at the same time making cross-currency payments across international borders easier and more accessible to their consumer population.

The XRP ledger is actually a series of individual ledgers or ledger versions managed by distributed server nodes called rippled (pronounced “ripple-dee”). The rippled server code is open-source and is available to anyone who wishes to establish their own instance of the server. An instance of the rippled server can be seen as similar to the concept of a “full node” in the bitcoin blockchain. Every instance of the rippled server carries a complete copy of the latest confirmed state of the XRP Ledger called the “Last closed ledger” and also some recent transactions that have not been verified yet. Each rippled instance independently verifies the set of transactions and if the result is in sync with the rest of the network they are considered validated. Therefore the XRP Ledger uses this overlapping consensus algorithm without the need of any mining process.

As mentioned previously, XRP is a debt-based currency and does not consume transactions like Bitcoin’s UTXO model. A finite amount of the XRP currency exists within the network and more will never be produced again. 100 billion XRP was released initially when the XRP Ledger was created and the currency is subdivisible upto 6 decimal points - totalling up to a total of 100 quadrillion *drops* of XRP[36].

A small amount of XRP is destroyed for every transaction on the network. Each account on the XRP Ledger is required to hold a minimum amount of XRP as a reserve. This is known as a Base Reserve and is currently at 20 XRP. There also exists an Owner Reserve, which is an increase to the reserve requirement for each object that an address owns in the Ripple Ledger. Currently, this is 5 XRP per item. Even though these measures make the currency slightly deflationary, it will take an estimate of 70,000 years to run out of all XRP[36].

Ripple mainly relies on the same ECDSA digital signature system used by Bitcoin. But it is also built to support newer and modern efficient algorithms like Ed25519. Its extensible nature allows it to add and disable algorithms as required.

3.3.2. R3 Corda

R3, initially a huge bank consortium consisting of some of the world's leading financial institutions has now transformed into a leading enterprise blockchain firm. It released its open-source distributed ledger technology platform, Corda, in 2016. In 2018, R3 released the commercial version of Corda for enterprise usage, called Corda Enterprise.

R3 built Corda targeting to solve real business problems in complex real-world markets. R3 prioritizes interoperability and envisions achieving "[...] optimisation at the level of markets, not at the level of firms"[37]. Thus Corda became a permissioned blockchain platform that would also allow competing parties to co-exist and transact across the same open network[38]. Other important factors in consideration are the privacy of transaction information and legal certainty. This means that parties are not completely anonymous on the network. A node has to go through a Know Your Customer (KYC) process, similar to a Bank's, and obtain a certificate before it can join the network. This certificate maps the node identity to a real-world legal identity and a public key[38]. Therefore, Corda does not employ a gossip protocol or broadcast communication all across the network. Communication is propagated only on a need-to-know basis also known as lazy propagation. This enhances privacy as all data is not made public to all nodes in the network.

The Corda Ledger is made up of data, that are called facts, shared among the nodes on the network. Each node on the network maintains a vault of facts it knows. Facts are shared among parties who were involved in the corresponding transaction. A fact that is shared with more than one node is always synced to appear the same to all of those nodes across the network. The whole of the ledger is made of the union set of all shared facts among nodes, therefore, there is

no node that can see all the facts on the ledger, each node only sees a subset of facts that are actually on the ledger. Hence, the Corda ledger is described as a "subjective construct from each peer's point of view"[9]. A node can also have a fact in its vault that is not shared with any other node. These are called unilateral facts but they are not a part of the ledger.

A Corda node is a deterministic JVM run-time environment with a unique identity on the network.[9] Each node on the network runs an instance of Corda and one or more CorDapps. CorDapps are distributed applications that take the form of a set of JAR files containing class definitions written in Java and/or Kotlin. These class definitions will commonly include States, Transactions, Flows, Contracts, Notaries and Oracles. These **CorDapp Concepts** are further discussed below.

States

States represent shared facts, known by one or more nodes on the distributed network. States in Corda take the form of immutable objects containing atomic units of data. Since they are immutable, states cannot be modified. Instead, when a change needs to be made, a new state is created which represents the new or modified fact and the existing state is marked as "historic". Thus, the lifecycle of a shared fact is represented by the *state sequence*. Each node on the network has a *vault*, which is the local database that keeps a record of all the current and historic states known to it. Thus, the ledger from a node's point of view is the collection of all shared facts known to it. There is no peer on the Corda Ledger that has a view of the entire ledger. Therefore, the Corda Ledger is considered to be a subjective construct that is made of all the views of the peers in the network.[9]

States contain a list of participants who are required to sign any transaction it is a part of. A state also contains a reference to the *contract* that governs it and a *notary* which has to notarise the transaction it is a part of.

Transactions

The Corda network validates every transaction in real-time. Corda follows the UTXO ("Unspent Transaction Outputs") model similar to bitcoin's. Facts are represented by states on the Corda Distributed Ledger and transactions are proposals to modify them. A transaction on the ledger consumes 0 or more input states and produces 0 or more output states. States can be consumed but they are never deleted. Consumed or spent states are marked as "historic" and persist on the ledger forever. It may persist for years and survive many upgrades and infrastructure changes. This is useful as ledger data typically represents business agreements and data and these spent states can be referenced to in the future if the need arises.

Another important component of a transaction, other than input and output states, are commands. Commands convey the nature of a transaction. A transaction may contain multiple types of states, transitioning from input state to output state. And each of these pair will have a command associated with it. For example, a transaction containing a cash state and a bond state as inputs and as outputs might be a bond purchase transaction or a transaction for a partial payment on the bond. Thus, the commands associated with them will clarify the intent of the transaction. Each command also has a list of required signers.

A transaction can optionally contain references to any number of attachments. Attachments are ZIP/JAR files that contain some piece of arbitrary data that might be needed for the transaction. For example, an attachment can be a calendar or a supporting legal document. Attachments are referenced by their hash in transactions and are not actually included in the transaction. The information in these files can be accessed by the contract code and used while checking the validity of the transaction.[9]

A transaction can also include a time-window that specifies a time period by which the transaction has to be committed. A time-window can be open-ended or bounded on both sides. A notary acts as the timestamping authority and verifies the transaction occurred during the time-window before notarising it.[9]

In order for a transaction's outputs to be considered valid, the transaction has to be both valid and unique. The transaction is valid if it is contractually valid and it is digitally signed by the union set of all required signers in the transaction. The transaction is unique if none of its input states has been consumed by an existing committed transaction. If all these conditions are met, the output states are considered valid and they become part of the current state of the ledger.[9]

Contracts

A transaction may contain multiple states and each state has a *contract* associated with it. These contracts carry the logic of the transaction. Each contract takes the transaction as input and states if the transaction is valid based on its rules. The transaction has to be contractually valid for each state in the transaction.

Contracts in Corda also have the provision to include legal prose. At times contract code is not enough to handle complex transactions that have real-world implications. Corda does not consider contract code as law, as in real-world that is not the case. Therefore, in order to handle complex transactions that may involve any element of legality, a contract can contain a reference to a legal prose document. This document is relied upon in case of any legal dispute regarding the transaction[9]. Thus, "prose trumps code" in Corda as in the real world.

Flows

As information is not broadcasted on the Corda network, nodes communicate with each other through the concept of Flows. Corda's Technical Whitepaper[39] defines flows as multi-party sub-protocols through which all communication takes place. Flows automate mundane tasks on the network and the process of updating facts on the ledger. They abstract "[...] all the networking, I/O and concurrency issues away from the node owner"[9]. They encapsulate the business process.

A flow can be started as a subprocess within the context of another flow. The subprocess is called a subflow, and the parent flow will wait until the subflow returns. Corda provides a library of flows and subflows to handle common tasks such as notarising, recording a transaction, verifying a chain of transactions and gathering signatures.[9]

Notaries and Consensus

Since the Corda network is permissioned there is no mining involved either, nor does the need exist for a native cryptocurrency to incentivize it. But it still relies on consensus. Corda implements what is called “pluggable consensus” through its notaries.

A Notary in its simplest form can be a single machine or a pool of mutually untrusting nodes coordinating via a byzantine fault-tolerant algorithm that provide various services to the Corda network including validation, timestamping and uniqueness services. Notaries can be validating or non-validating. A network can have several notary clusters, each running a different consensus algorithm[9]. They verify the transactions submitted to them and return a signature over the transaction if the transaction is successfully validated. In the case of a failure, it returns an error. The presence of a notary signature from the state’s chosen notary indicates transaction finality[39].

Oracles

Some transactions might require information from outside the blockchain network. This data can be a currency exchange rate, weather forecasts, market forecasts, etc. But whatever information or calculations that come into the ledger must be ensured to be deterministic in regards to the whole system. That is, every node on the network should be able to verify the transaction and come to the same conclusion at any time and on any computer[39]. This task is carried out by Oracles in the blockchain world.

Oracles are special nodes that provide some required information from outside the deterministic blockchain network. In Corda, if a node needs to use such information in a transaction, it

requests a command asserting this fact from an Oracle. The Oracle sends back the command if it finds the fact to be true [9]. The command is added to the transaction and the Oracle signs the entire transaction to validate the fact.

Even though the Oracle signs the entire transaction, it might not need (in most cases) to see the entire details of a transaction. The unnecessary data is abstracted from the Oracle using the concept called filtered transactions or transaction tear-offs. Wherein which a nested Merkle tree is used to “tear-off” any part of the transaction the oracle need not see[9].

4. UNIVERSITY CRYPTOTOKENS & INTERNATIONAL TRANSFERS

Problem

According to the Canadian Bureau for International Education[40], there has been a 185% increase in the number of international students in Canada throughout the years 2010-2019. For various purposes including tuition and other living expenses, parents or guardians of International students often need to send money to them across borders. Due to the inefficiencies and complications in the conventional banking systems, a significant amount is remitted as transactional fees from banks on both sides and sometimes by even more intermediaries.

Another alternative commonly used by International students are GICs or Guaranteed Investment Certificates. This requires a person to deposit a large sum of money in a Bank's account for a specified period of time. This period of time could be upto 5 years and there might even be a penalty if the amount is withdrawn before the specified time.

Proposed Model

In this section, we propose a crypto token system for universities to implement a micro-economy within their community. The highlight of the model is the feature to accommodate easy and fast global transfer of funds with minimal transaction fees for the convenience of international students. The following model is highly practical and designed by combining two new, yet powerful Distributed Ledger Technology platforms - R3 Corda and Ripple. The cryptotoken system and the distributed network for the university is proposed to be built using Corda and to be run on Corda's Distributed Ledger. The International transfer of funds is realized through initiating what is known as flows in the Corda framework between the Corda nodes and the XRP Ledger Network.

This model provides a number of advantages to the institution implementing it. The top one being that Universities can provide a solution to a very widespread financial problem affecting

its international student body. The institution is also able to monitor incoming international funds and apply constraints whenever necessary on factors such as the maximum amount of International transfer allowed per student per year.

The parents or guardians of the International students will also be provided with an easy, fast and direct method to transfer money to their wards, without losing a significant amount as transaction fees.

Token System

Tokenization is not a new concept in the blockchain or cryptocurrency paradigm. The initial crypto token was made possible by Ethereum with its ERC20 tokens. Corda provides the same functionality through the TokenSDK. The token SDK provides APIs and flows to implement standard token related tasks for UTXO based ledgers[39]. The below figure shows the structure of Token Classes implemented in Corda.

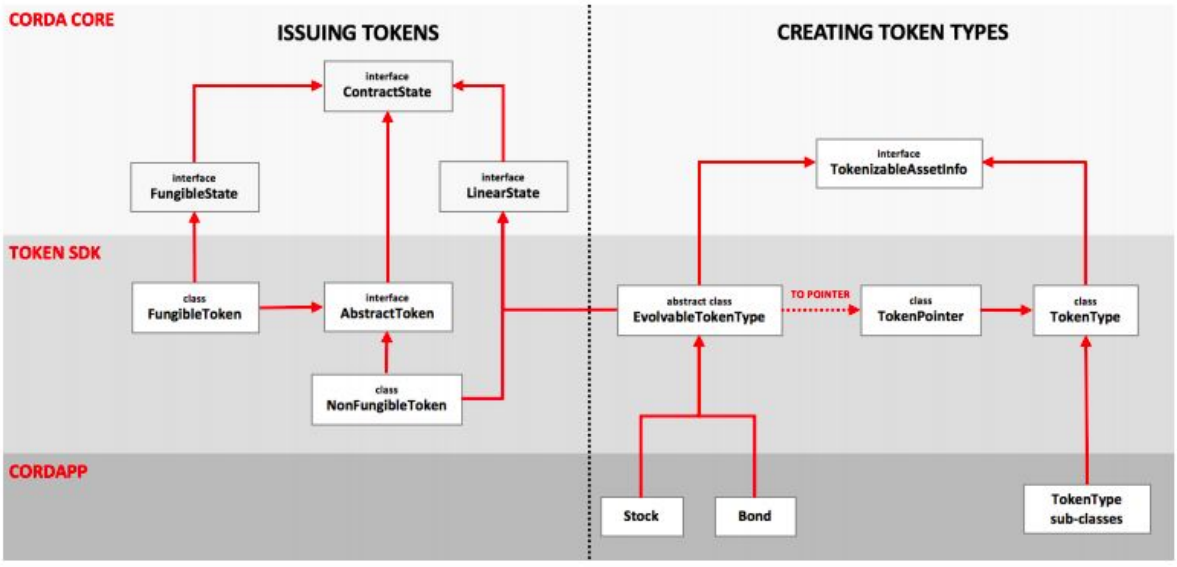

Fig. 4.1: Token State Hierarchy in Corda[39]

There are two token types in Corda: Fixed Token Type and Evolvable Token Type. Since we are representing a fiat currency with our tokens, we use the Fixed Token Type that does not change over time. All Fixed Tokens implement the *TokenType* interface. Corda also provides two classes - the *FungibleToken* class and the *NonfungibleToken* class to represent fungible and non-fungible assets respectively. Once again, because a currency is a fungible asset, we use the *FungibleToken* class to issue the tokens. To do this we first create an instance of the *IssuedTokenType* wrapper class that contains a token type and the issuing party of the token and then use this to create an instance of the *FungibleToken* class. We use Corda's Flow Framework to issue, transfer and redeem the tokens.

International Transfer through the XRP Ledger Network

The University is required to have an XRP account on the XRP Ledger Network in order to temporarily hold funds in XRP that is sent from another country. The XRP payment is made by initiating a Corda flow which calls out to a Ripple node. The International sender node, after querying the Conversion Oracle, creates a transaction with the following details:

- The specified amount
- The University's XRP address
- The Student ID of the student to whom they are sending the funds to.

The transaction is then signed and sent to a ripple node on the XRP Ledger Network. The XRP Ledger sends back a confirmation of the payment received in the form of a transaction hash to the sender.

The sender adds this transaction hash with all the initial data and creates a Payment Submission Receipt, obtains a signature for validation from a validating notary and sends it to the *Payment Confirmation Oracle* and the Student Node. The *Confirmation Oracle* queries the XRP Ledger about the payment using the Transaction Hash. Once the payment is confirmed, it obtains a signature from a validating notary and sends the information to the F.O Node (the admin node run by the Finance Office of the University) and the sender node.

Once the F.O Node receives a payment notification from the Confirmation Oracle, it initiates a flow to the XRP Ledger to transfer funds from its XRP Account to the University Bank Account and then issues the appropriate amount of tokens to the corresponding student.

Proposed Network Structure

The proposed network structure is illustrated in Fig 4.2 below.

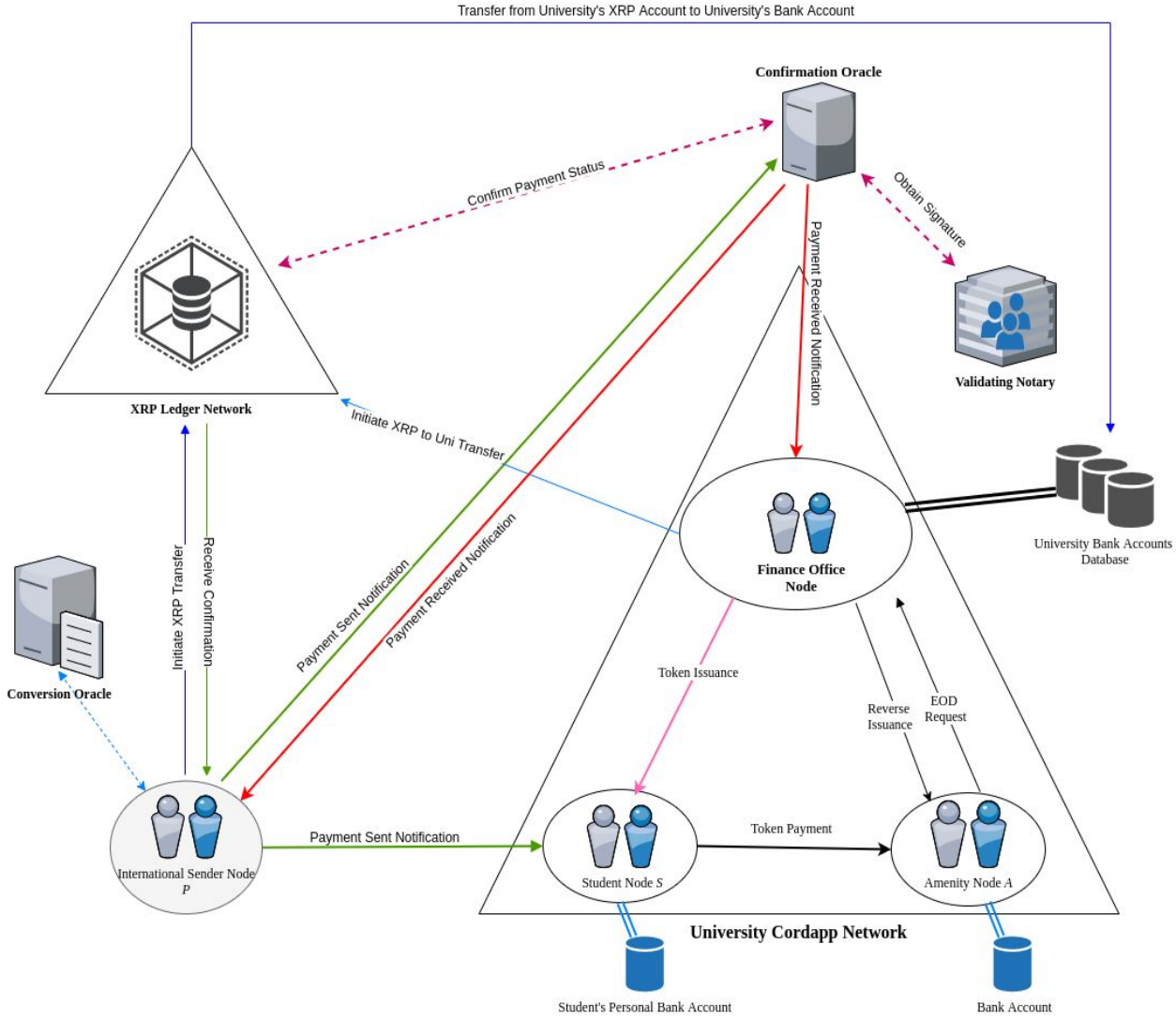

Fig 4.2: Network Diagram of Proposed Model

Nodes in our Network:

1. Student nodes denoted by *S.* - The CorDapp Node designed for university students through which they access and manage tokens.
2. Service nodes, *A* - The various amenities available at the university campus where token payment is accepted. Can include Accounts Payable and University Bookstore, cafeterias, cafes. And also establishments such as Starbucks and Subway.
3. International nodes denoted by *P*. - International Currency Sender CorDapp Node, usually parents or guardians of International Students.
4. Finance Office (F.O) - the administrator node under the authority of the Finance Office of the university.
5. Validating Notary - Corda Notary that validates and signs transactions.
6. Conversion Oracle - Corda Oracle that checks the currency exchange rate and calculates the equivalent amount in Canadian Dollar for International transfer.
7. Confirmation Oracle - Corda Oracle that Confirms the receival of funds by checking with the XRP Ledger Network.

Transactions on the Network:

1. **International Transfer**

   International transfers are broken down into two parts that are initiated by two different nodes:

   1) The International sender node transfers the amount to the XRP Ledger account of the Finance office. Thus, converting the amount in a foreign currency to XRP currency.

   ➔ Node *P* queries the *Conversion Oracle* to obtain details on current conversion rates and to calculate the amount that will be received in the destination currency.

- ➔ Node *P*'s CorDapp initiates a transfer transaction flow to the XRP Ledger Network. This includes information such as the XRP Address of the university, Student ID of the student the sender is sending funds to, etc.
- ➔ The XRP Network sends back a confirmation of payment submission with a Payment Transaction ID.
- ➔ Node *P* attains signature from the Validating Notary and distributes this info to the Confirmation Oracle and the corresponding Student Node *S*.
- ➔ The Confirmation Oracle checks the status of payment with the XRP Ledger Network and once it is confirmed, obtains a signature from the *Validation Notary* as proof of validation and sends the info to the F.O Node and the sender node *P*.

2) The F.O Node upon getting notified from the Confirmation Oracle initiates the transfer of funds from the XRP Ledger to the University Bank Account. Effectively converting the XRP amount to $CAD. And then the F.O Node consequently issues the appropriate amount of tokens to the corresponding student.

- ➔ F.O Node initiates transfer of funds from its XRP Account to University Bank Account.
- ➔ F.O Node issues tokens to the corresponding student node *S*.

2. **Token Issuance** - F.O to students
   - ➔ The F.O Node receives a confirmation of receival of funds from the *Confirmation Oracle.*
   - ➔ Issues tokens to corresponding student node *S*.

3. **Reverse issuance**(cash out) - F.O to students
   - ➔ Student node *S* sends a Cash-Out Request along with the tokens to the F.O Node.
   - ➔ F.O marks the tokens as "historic" and cannot be used again.
   - ➔ F.O Node initiates a local payment to the corresponding student's bank account.

4. **EOD reverse issuance** - Between F.O and services
    - ➔ Service *A*, sends an EOD request to F.O Node along with the tokens.
    - ➔ F.O marks the tokens as “historic” and cannot be used again.
    - ➔ F.O Node initiates a local payment to the corresponding bank account of the service.

5. **Payment** (using tokens) - Student to Services
    - ➔ Any student node *S* is able to issue payment to any service node *A*.

6. **Transfer** - Student to student
    - ➔ Tokens can be transferred freely between student nodes.

Feasibility

How does this help?

- Parents can send money directly to the University’s reserve instead of through multiple banks and save a lot in terms of transaction fees and better conversion process.
- University is able to keep track of the incoming International currency while establishing constraints as needed.
- International students do not have to rely on external banks on alternate methods such as GIC to transfer and access funds from their home country.

Why XRP Network?

It is a trusted global system that has been proven to work. Numerous big banks with high credibility are part of the network and through them, the issue of legality is solved. Because Ripple is committed to monitoring and reporting any Anti-Money Laundering (AML) flags detected across the XRP Ledger network, as well as to reporting suspicious activity to FinCEN as applicable. And KYC is taken care of by the trusted credible parties in the network.

Why R3 Corda?

Corda was created for enterprise settings with a governance model. The modular architecture is optimal for building our distributed network with a large number of nodes, governed by one or two authority nodes. The Flow architecture of Corda helps in the easy automation of tasks such as validating token transfer transactions in between students and as well as between students and services. The Flow design also emulates multithreading, resulting in very high throughput.

Vs Ethereum & Hyperledger Fabric - Cordapps have the edge to work within the legal framework of the real world. They are not based on "code is law" as in Ethereum, where smart contract code rules over transaction disputes. The explicit linking between code and legal prose establishes "law is law" as in the real world. Hyperledger Fabric now supports Ehtereum smart contracts and even has had a reported 20,000 TPS[41]. But Fabric has a number of design issues and has gone through a number restructurings due to the same. Another aspect is interoperability, whereas Corda was built from step one with interoperability in mind, Hyperledger displays a distinct lack of it. The interoperability of Corda's Flow framework is what enables us to communicate with the XRP distributed network.

Scalability

Following this model, any educational institution will be able to establish their own private cryptotoken system, facilitating a micro-economy within their campus. Any additional rules or laws regarding International transfer of funds can be integrated into the prose section of Corda contracts.

Future Work

Future work involves developing the Corda distributed network for a University model according to their administrative structure. The finance or accounts department can be in charge of the system according to their model. Or the University can also set up a new department collaborating IT and finance departments to moderate the network. Setting up the token system for the University and enabling various services within the campus to accept token payments is a

crucial first step. Familiarizing the students with the system and promoting the usage of the tokens are also critical. An initial soft launch of the system for a short duration can be used to collect data and analyze it, to make consequent modifications in the smart contracts.

5. BLOCKCHAIN TAX LEDGER

Problem

In April 2016, a dedicated unit was established at the Canadian Revenue Agency (CRA) to analyze Canada's GST/HST Tax Gap. They released a report titled "Estimating and Analyzing the Tax Gap: Related to the Goods and Services/Harmonized Sales Tax (GST/HST)"[42] in June 2016, analyzing the tax gap throughout a 15 year period from 2000 to 2014. The GST/HST gap was estimated to be an average of 5.6% of potential GST/HST revenues in this period.

The CRA reports that the tax gap, even though frequently associated with fraud and tax evasion, is the result of both intentional and unintentional actions. Non-compliance can be due to deliberate choices to hide income, over-claim deductions or under-report income. But it can also be due to mistakes regarding ignorance in filing and reporting of taxes too.[42]

The CRA also spends a lot of resources to check the tax returns and carry out audits every year. However, this is a difficult and long process that takes up a lot of manpower of the CRA. By introducing immutable blockchain ledgers and its features to capture and keep track of sales and related tax details, we leave no window for either intentional or unintentional non-compliance.

In the following, we initialize a proposal for CRA's GST/HST handing. This is an ongoing project. We have discussed with the CRA members during our work and got useful suggestions and feedback from them.

Proposed Model

This proposal serves the purpose of establishing an E-service of GST/HST handling for CRA. This model borrows basic techniques from blockchain, however, the design concepts are very different from other blockchains. The following draft is a brief outline of a private blockchain

system designed after some preliminary discussions with the CRA. Further discussions are required to streamline the system.

Note: After further discussions with the CRA, due to legality and confidentiality concerns they were unable to provide us with the data model they work with. As a result, this draft is currently incomplete. However, CRA did give us the main ideas of the requirements of the system.

The CRA pointed us towards the Sales Recording Module (SRM) systems, also dubbed as "blackboxes", recently introduced by Revenue Québec in the province of Québec. This is a mandatory billing system that has to be used by all restaurant establishments including catering services and bars. It comes as a tamper-proof microcomputer that has a security seal from IBM Canada [47] and it records all sales of the establishment and generates receipts with unique barcodes. The blackbox system is used in conjecture with a Sales Reporting System (SRS) which is the cash register or point-of-sales (POS) system [43] that is compatible with the SRM. The module must also be connected to an SRM-compatible receipt printer. Every month the establishment is obliged to produce and send a *Sommaire Périodique des Ventes* (periodic sales summary or SPV) to Revenue Québec. The SPV summarizes all transaction data of that period and can be sent electronically or by mail.

Assumptions

Our proposal is based on the following assumptions:

The CRA has a powerful computer server which has enough storage space. CRA has a public key system with public key CK_{pub} and secret key CK_{sec}. In what follows, we will use RSA signature algorithm $Sig_K()$ and hash function SHA-256. But this is just for the purpose of demonstration. Any secure asymmetric key system and hash function can be used in their stead.

The business vendors are denoted as B_1, B_2,..., B_n. Each business has some computer device D to use, which can perform signature and hash calculations and has the ability to store certain data.

In our proposal, the communication is only between CRA and B_i. So we just use B to denote a business vendor. Optionally, CRA and B_i share a secret master key K_{master}. If a special device D is designed and distributed by CRA, then the K_{master} can be pre-installed in D. D could also be a software implementation. In that case, B may use a regular computer. There is a communication channel between CRA and B. Since B has an account in CRA's computer, SSH may be used to secure the communication. Another option is TLS. To simplify our protocol, we assume that the communication channel is secure, that means no errors will be added from the communication.

We also assume that the CRA is always honest and will follow the protocol. B might not be honest. In this way, B can use HMAC instead of the signature if a K_{master} is used to create session keys.

B2C Sales Tax

In this section, we consider the Business-to-Consumer or B2C sales records. When B sells a product or service to a customer, B needs to give the customer a receipt which includes the value of GST/HST the customer paid. The receipt should be created through our computer device D. The receipt includes a truncated Merkle tree root.

Initialization

CRA creates an account for B. This account will form an e-block, that will be recorded in CRA's computer and one copy will be sent to B. This block consists of the following information: account number, account name, create time, valid timestamp, block hash, CRA's signature, CRA's public key, B's public key, B's acknowledgement. The input of the block hash is all the data except the block hash and signature. Initially, B's acknowledgement is 0.

B has its own public key system, which can be created by CRA or other public-key authorities. Before CRA created the account, B should send its public key to CRA. If CRA is always honest then B and CRA can use the K_{master} to establish a session key and then B uses HMAC instead of a signature to authenticate messages. After B receives the e-block, B verifies the CRA's signature

and puts this block as the first block in a blockchain. B then computes a hash value of the block and signs it. B sends the hash value and signature to CRA and CRA records the information as B's acknowledgement.

Sale or return of an item

When B sells an item to a customer, B creates an e-block of invoice with a block header. The block is then put in a blockchain called saleChain. The invoice contains regular data about the sale. The block header consists of the following: B's account number, block hash, previous block hash, B's signature, invoice number, sale time, tax due time, GST/HST customer paid or received, B's payable tax amount, Merkle tree root, A flag which indicate this is for selling or returning an item, returning information.

B stores the e-block in the Merkle tree. Before B stores the block, B needs to update the Merkle tree and update the Merkle tree root. B prints a copy of the invoice to the customer and sends the block header to CRA. (if CRA wants to keep all the sales information for later data analysis, then B will send the whole block instead of just the header). Upon receiving, CRA updates B's Merkle tree and checks the correctness of the received B's Merkle tree root. If there are any mistakes, then CRA will check with B.

The e-block is explained in Figure 5.1. The top part (excluding the invoice) is the header of the block. The size of the block header is about 0.5 - 0.6 KB. At this moment, we haven't defined the size of the invoice. But the size of the invoice may be flexible. The invoice contains a truncated signature which establishes the connection from the invoice to the block header.

Tax return

AccountNum	B's account number
blockHash	256 bits hash value of the block. The input of the hash function includes: all the data in this block except blockHash and Signature

Signature	B's signature on the blockHash, 2048 bits RSA signature
PrevBlockHash	The hash value of the previous block, 256 bits
InvoiceNum	Invoice number of this sale
SaleTime	The time of this sale
DueDate	Tax due date
GST/HST	The amount of GST/HST paid or received by the customer
TaxSum	The amount of accumulated payable GST/HST of B including this sale
MerkleRoot	The hash value stored in the root of the Merkle tree which includes this block, 256 bits
Flag	indicate if this block is for returning (01), or if there are tax credits claim change (02), or for both (03). If this is just for a sale, the flag will be (00) and the next two fields are omitted.
ReturnInfo (optional)	Returning item's InvoiceNum and blockHash. This field will be omitted if this is not returning
Credit claim (optional)	Information about tax credit claims, including smaller subfields. (See B2B)
Invoice	Truncated signature This area includes the details of the invoice data. The invoice must contain the amount of GST/HST paid by the customer, B's account number, the invoice number and the sale's time. The size of the invoice is flexible. If this is a return, then some detailed return information will be in this location.

Figure 5.1: B2C block fields

We assume that B needs to pay tax periodically. By the due date, B sends its blockchain along with its payment to CRA. The paid tax equals the sum in the TaxSum of the last block. CRA and B then start a new chain for sales. After checking the correctness of B's tax return, CRA sends a

receipt to B. After B receives the CRA's receipt, B's old blockchain may be deleted. CRA will store the old blockchain of B somewhere for a certain time.

Verification

If verification is required, CRA checks the Merkle tree. If something is flagged as incorrect, then CRA can trace it to the corresponding block to find out the cause of the mistake. Since CRA's Merkle tree just stores the block headers, the storage and the checking time will be efficient.

Notes: Main ideas about the design of the blockchain are as follows. Both CRA and B use digital signatures to authenticate their communication. All the sales within a tax period form a blockchain so that a sale cannot be changed without changing all the other sales. The previous block hash is used to connect blocks as a chain. Merkle trees are used for fast verification. The information about each sale will be sent to CRA instantly so that it will be difficult to change it after the item sale or return.

For n sales, the Markle tree needs $2^{\log_2 n+1}$ nodes, each node contains a 256-bit hash value except the n leave nodes which contains block headers each of about 0.6 KB. For a correct blockchain of size n, the verification involves computing $\log_2 n$ hash values.

B2B Sales Tax

For the B2B case, we start from a simple idea of a complex transaction. For example, B is a manufacturer of clothes. Suppose B needs to buy fabrics, buttons, etc. from different sellers. We use Inv1, Inv2, ..., to denote different inventories. B will receive invoices (assume all these invoices are e-invoices). When B sells clothes or other products, B also gives customer invoices which can be used to claim tax credits. We denote the sales of the product as S1, S2, ..., etc. B records all this information in the blockchain system.

For each kind of inventory, B creates and maintains a blockchain as follows. For example, a blockchain of fabric. A block contains 3 parts: header, buying and selling. The header contains hash values, signature etc. The buying part contains the information about fabric bought with the amount of tax paid. The selling part contains the sales-related tax claim for the fabric. Each time when B buys fabric or sells a product related to fabric, a new block is created and put into the blockchain.

We use an example to explain the idea. Suppose B bought fabric two times: InF1, InF2 and button two times: InB1, InB2. Suppose B also had 3 sales: S1, S2, S3. S1 used fabric and button, S2 used button, S3 used fabric. B maintains two inventory blockchains: one for fabric and another for buttons. When S1 was created (S1 is sold), a new block in fabric blockchain was

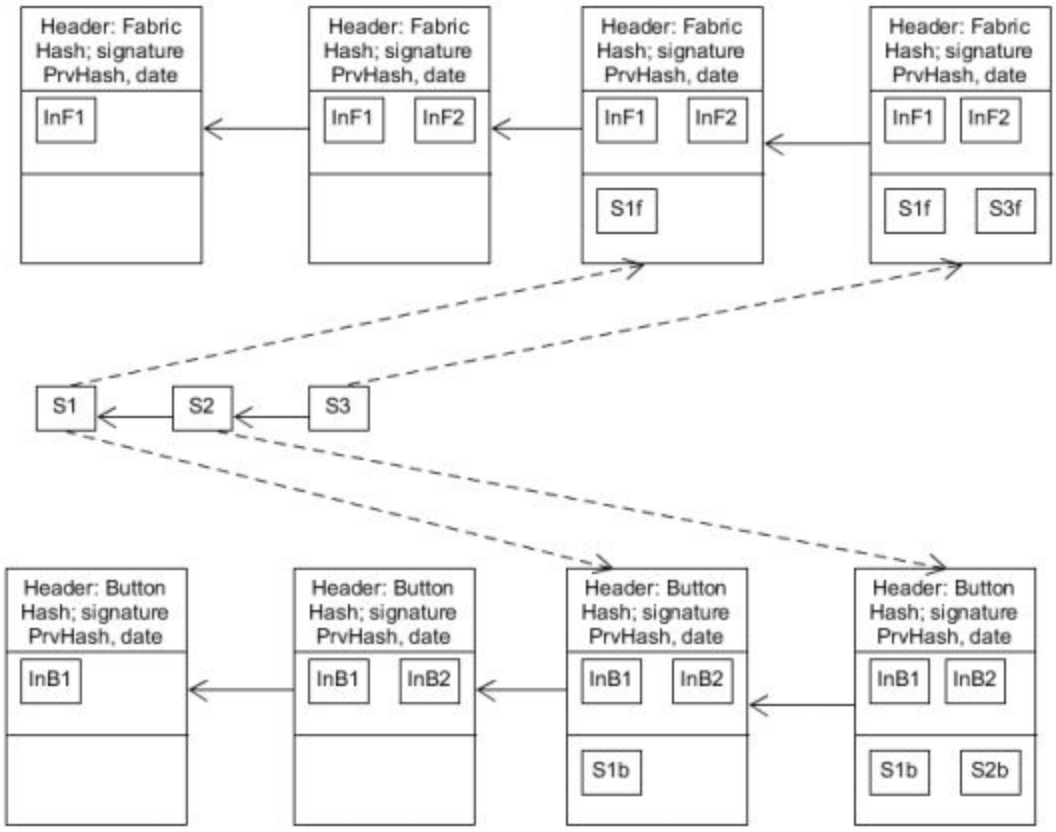

Fig 5.2: Sample B2B chain structure

added and a new block in button blockchain was added. The tax credits are claimed in S1f, S1b for different inventory respectively. The total claims are recorded in a sales blockchain, which is a B2C blockchain, in which a field of tax claim is added in the corresponding block. The correspondent block hashes are recorded in S1. S1 recorded the tax the buyer paid, and the tax credits claimed for fabric and button, and the remaining payable tax. Since all the block hashes are recorded, they cannot be changed after added. The sales chain will adjust the accumulated payable tax. We outline the chains in Figure 5.2. For the details of the B2B case, we need more information to consider the data model. The example gives us only a basic idea on the structure.

Feasibility

The usage of a private permissioned blockchain ledger to record sales and tax information is very appropriate for the use-case. Classic properties of a blockchain are favorable for the system. Permanent and irreversible transaction records prevent underreporting of sales. Any flags raised at any point in the blockchain is easily traceable to the error point. Easy and fast validation makes auditing processes faster too. And auditing does not even need to be on-site and can be decentralized.

Implementing a whole new blockchain system in all sectors for the whole country might take some time and should be done in phases. The CRA could follow the model of Quebec and implement it one sector at a time for the B2C model. And once the whole system is stable, the B2B model can be commenced.

Future Work

Future work is to define the data model for the B2B sales transactions. Building the blockchain structure to include the various elements involved during Business to Business transactions and the subsequent tax calculations. Another aspect for future work is automating the preparation of tax return information for businesses.

For audit purposes, various algorithms can be written using the current ones CRA uses to set up flags to notify the CRA if some suspicious activity has been detected. Smart contracts can be added to the system that detect patterns or unusual activity. Random checks can also be done. More information, that might be confidential, is needed for this work.

6. CONCLUSION

Through this research, we illustrate the rapid evolution of blockchain technology since the advent of Bitcoin in 2008. From permissionless, public and anonymous blockchain systems to permissioned distributed global networks. And even the concept of the *Internet of Value* that might be achieved in the near future making monetary transactions as seamless as the current transmission of data.

Another common feature appearing in the newer open-source blockchain platforms is the inclusion of some kind of governance in the distributed system. This "progress" might be anti-intuitive to the initial principle of a decentralized blockchain. The very driving point of Blockchain or Bitcoin's initial boom of success was, after all, the absence of a centralized authority. Also, the fact that every single node in the network contributed to the decision making process.

Even though Bitcoin holds and deserves to hold all the credit for starting the explosion of blockchain research and development, the blockchain community also has witnessed and learned a lot from the growth of Bitcoin through the years. We have seen how unprogrammable bitcoin's underlying blockchain is, we achieved turing-completeness with ethereum and from ethereum and the second generation of blockchain technologies, the community has now come to the realization that there is no need to throw away existing code each time a new blockchain technology has to come to play. Corda built its platform on existing JVM code. Providing a tested, well-supported ecosystem for building your own blockchain networks and nodes. And platforms like Ripple are trying to break the pattern of multiple individual blockchain networks to one interoperable global distributed network. As the technology expands further, we can expect more changes in its structural and fundamental principles, as it transforms to facilitate new use-cases that go beyond currency, economics and markets.

CITATIONS

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System.” Oct. 2008.

[2] S. Haber and W. S. Stornetta, “How to time-stamp a digital document,” *J. Cryptology*, vol. 3, no. 2, pp. 99–111, Jan. 1991, doi: 10.1007/BF00196791.

[3] A. Back, “Hashcash - A Denial of Service Counter-Measure,” Aug. 2002.

[4] W. Dai, “B-money,” 1998.

[5] M. Swan, *Blockchain: Blueprint for a New Economy*. “O’Reilly Media, Inc.,” 2015.

[6] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, “Hash Functions and Data Integrity,” in *Handbook of Applied Cryptography*, 1996, pp. 321–383.

[7] C. Percival and S. Josefsson, “The scrypt Password-Based Key Derivation Function.” 2016, doi: 10.17487/rfc7914.

[8] R. C. Merkle, “A Certified Digital Signature,” *Advances in Cryptology — CRYPTO’ 89 Proceedings*. pp. 218–238, doi: 10.1007/0-387-34805-0_21.

[9] “Corda OS 4.4.” [Online]. Available: https://docs.corda.net/docs/corda-os/4.4.html. [Accessed: 10-Apr-2020].

[10] D. Yaga, P. Mell, N. Roby, and K. Scarfone, “Draft NISTIR 8202, Blockchain Technology Overview.” .

[11] Hyperledger Architecture Working Group and Others, “Hyperledger Architecture Volume 1: Introduction to Hyperledger Business Blockchain Design Philosophy and Consensus,” 2017.

[12] “Sawtooth v1.2.4 documentation,” *Hyperledger.org*. [Online]. Available: https://sawtooth.hyperledger.org/docs/core/releases/latest/. [Accessed: 10-Apr-2020].

[13] C. Walter, “Proof of authority (PoA),” *tokens-economy.gitbook*. [Online]. Available: https://tokens-economy.gitbook.io/consensus/chain-based-hybrid-models/proof-of-authority-poa. [Accessed: 10-Apr-2020].

[14] J. R. Willett, “The second bitcoin whitepaper,” 2013.

[15] Artillar, “Omni Layer (OMNI) Coin – Cryptocurrency,” *BitcoinWiki*, 29-Mar-2018. [Online]. Available: https://en.bitcoinwiki.org/wiki/Omni_Layer. [Accessed: 10-Apr-2020].

[16] B. Chaser and D. Man, “In-Depth Litecoin (LTC) Project Analysis,” Blockchain Whispers.

[17] “Comparison between Litecoin and Bitcoin - Litecoin Wiki,” *Litecoin Wiki*. [Online]. Available: https://litecoin.info/index.php/Comparison_between_Litecoin_and_Bitcoin. [Accessed: 10-Apr-2020].

[18] “Peercoin Docs.” [Online]. Available: https://docs.peercoin.net. [Accessed: 10-Apr-2020].

[19] S. King and S. Nadal, “PPCoin: peer-to-peer crypto-currency with proof-of-stake (2012),” 2012.

[20] S. King, “Primecoin: Cryptocurrency with prime number proof-of-work,” 2013.

[21] V. Buterin, “Primecoin: The Cryptocurrency Whose Mining is Actually Useful,” *Bitcoin Magazine*, 08-Jul-2013. [Online]. Available: https://bitcoinmagazine.com/articles/primecoin-the-cryptocurrency-whose-mining-is-actually-useful-1373298534. [Accessed: 10-Apr-2020].

[22] E. Duffield and D. Diaz, “Dash: A payments-focused cryptocurrency,” *Whitepaper, https://github. com/dashpay/dash/wiki/Whitepaper*, 2018.

[23] Dash Core Group, Inc, “Dash Documentation — Dash latest documentation,” *Dash Core Group, Inc.* [Online]. Available: https://docs.dash.org/. [Accessed: 10-Apr-2020].

[24] “A Next-Generation Smart Contract and Decentralized Application Platform,” *White paper*, vol. 3, no. 37, Jan. 2014.

[25] “What Is Ethereum?,” *bitrates*. [Online]. Available: https://www.bitrates.com/guides/ethereum. [Accessed: 10-Apr-2020].

[26] “A Blockchain Platform for the Enterprise — hyperledger-fabric,” *Read the Docs*. [Online]. Available: https://hyperledger-fabric.readthedocs.io/. [Accessed: 10-Apr-2020].

[27] E. Androulaki *et al.*, “Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains,” *arXiv [cs.DC]*, 30-Jan-2018.

[28] “A Blockchain Platform for the Enterprise — hyperledger-fabric,” *Read the Docs*. [Online]. Available: https://hyperledger-fabric.readthedocs.io/. [Accessed: 10-Apr-2020].

[29] T. Blummer, M. B. Sean, and C. Cachin, “An Introduction to Hyperledger,” Tech. rep. 2018. url: https://www. hyperledger. org/wp-content/uploads/2018 …, Aug. 2018.

[30] “What is Hyperledger? A comprehensive Guide for Innovators,” *leewayhertz.com*,

16-Apr-2019. [Online]. Available: https://www.leewayhertz.com/what-is-hyperledger/?utm_source=edureka. [Accessed: 10-Apr-2020].

[31] "Hyperledger Iroha documentation," *Read the Docs*. [Online]. Available: https://iroha.readthedocs.io/en/latest/index.html. [Accessed: 10-Apr-2020].

[32] "Hyperledger Burrow." [Online]. Available: https://hyperledger.github.io/burrow. [Accessed: 10-Apr-2020].

[33] "Hyperledger Indy 1.0 documentation," *Read the Docs*. [Online]. Available: https://hyperledger-indy.readthedocs.io/en/latest/toc.html. [Accessed: 10-Apr-2020].

[34] Hyperledger Besu community, "Hyperledger Besu Enterprise Ethereum Client." [Online]. Available: https://besu.hyperledger.org/en/stable/. [Accessed: 10-Apr-2020].

[35] "Hyperledger Besu," *Hyperledger Wiki Org*. [Online]. Available: https://wiki.hyperledger.org/display/BESU/Hyperledger+Besu. [Accessed: 10-Apr-2020].

[36] "XRP LEDGER." [Online]. Available: https://xrpl.org/. [Accessed: 10-Apr-2020].

[37] "R3 | DLT & Blockchain Software Development Company," *R3*. [Online]. Available: http://r3.com. [Accessed: 10-Apr-2020].

[38] R. G. Brown, "The corda platform: An introduction," May 2018.

[39] R. G. Brown, J. Carlyle, I. Grigg, and M. Hearn, "Corda: A Distributed Ledger," 2016.

[40] "CBIE | Global Leader in International Education," *CBIE*. [Online]. Available: https://cbie.ca/. [Accessed: 10-Apr-2020].

[41] R. Gledhill, "Choosing an Enterprise Blockchain: An exhaustive guide," *Medium*, 16-Aug-2019. [Online]. Available: https://medium.com/swlh/choosing-an-enterprise-blockchain-an-exhaustive-guide-749ba7db382c. [Accessed: 12-Apr-2020].

[42] "Estimating and Analyzing the Tax Gap Related to the Goods and Services Tax/Harmonized Sales Tax - Canada.ca," Canada Revenue Agency, Aug. 2017.

[43] "SRM User Guide," Revenu Québec, Aug. 2019.