Final Report: Advanced Topics in Blockchain

1. Introduction

Blockchain technology has come a long way since the introduction of Bitcoin. Developed by Satoshi Nakamoto in 2008, Bitcoin assembled a number of previously introduced ideas in a new and exciting way which allowed for the creation of something that had only been talked about before - digital money. Unlike previous attempts, Nakamoto was successful in creating a public, secure system built on a peer-to-peer network that did not rely on any central authority for verification or accounting purposes. By combining a collection of transactions into a block and then chaining them together with hash pointers, Bitcoin has provided the world with an immutable digital ledger that can be potentially used for much more than just recording monetary exchange.

After Bitcoin, the second most popular blockchain network is Ethereum. Ethereum was created by Vitalik Buterin at a time when people were trying to add functionality to Bitcoin so that it could do more than just act as vehicle for transmitting value via simple transactions. Bitcoin uses a stack-based scripting language and although many interesting ideas have been implemented on the Bitcoin blockchain there are a number of reasons why Bitcoin did not facilitate these up and coming use-cases very well. Vitalik was excited about the possibilities of blockchain technology and he theorized a new type of blockchain that could facilitate, amongst other things, a relatively new and important idea at the time – smart contracts. To accomplish his goals, Vitalik created a new blockchain that featured a Turing complete programming language. Ethereum also introduced other improvements over Bitcoin including those related to speed and throughput, and security.

From the very beginning, it was recognized that the Bitcoin blockchain was not going to be able to scale to the lofty heights that many in the community dreamed of. Numerous people embraced Bitcoin and saw Bitcoin as a means to end the reign of fiat currency and private money creation which rested in the hands of the world's economic overlords – central banks. For Bitcoin, or even later iterations like Ethereum, to become a world currency it would need to be able to handle a much higher throughput, most notably measured in transactions per second. Private, centralized networks handling payment processing like Visa could handle thousands of times the number of transactions per second compared to either Bitcoin or Ethereum. The first iteration of blockchains, which include both Bitcoin and Ethereum, are peer to peer networks that implement the gossip protocol. The whole network must essentially know about and agree on a valid set of transactions before consensus can proceed. If a new block is created before part of the network has had time to settle on a new state, then multiple versions of the blockchain will begin to proliferate. Miners in the network work to solve cryptographic puzzles with

the difficulty of these puzzles set to prevent forking from happening in order to allow consensus to be reached across the entire network. Because of this, the consensus model in both Bitcoin and Ethereum is called proof-of-work.

Many ideas have been introduced in the blockchain space in the hopes of creating a new type of decentralized peer-to peer network that can scale well beyond the 7 transactions per second offered by Bitcoin. One popular theme in many of these ideas relates to the notion of consensus. Because validators or miners in the Bitcoin network work to solve cryptographic puzzles, numerous blocks can potentially be produced in the same time frame. It takes time therefore for the network to vote on which block it will add to the blockchain. If there are two valid blocks, Miners will choose which block remains in the blockchain by voting or dedicating their hash power to that block. The network at large will reach consensus via the longest chain rule. The longest chain will be chosen as the definitive chain because it offers the most security. Individual transactions in proof-of-work blockchains usually need to wait for a number of new blocks to be added before a transaction can be determined to have been carried out successfully. If someone accepts a transaction before waiting, by offering a service in exchange, they may lose out, if the bitcoin they received ends up being contained in a discarded block.

Consensus in proof-of-work blockchains is reached slowly for a number of reasons. One of the most popular ideas related to replacing the proof-of-work consensus model is called - proof-of-stake. A proof-of-stake network will generally reach consensus faster than one using proof-of-work. In proof-of-stake, a validator is chosen at random with the odds of that validator being chosen weighted by the number of tokens that a potential validator holds. The higher the stake you have in the network, the higher the chance you have to be chosen as a validator. Being chosen as a validator means inflation is created in the network and newly minted tokens are rewarded to you. The network only needs to ensure that the consensus protocol was correctly followed and that all of the transactions in the block are valid before it can proceed to allow for a new set of transactions to be looked at. Theoretically in a proof-of stake blockchain, blocks are being created at a much quicker rate and there is not the same need to wait for a prescribed number of blocks before having confidence that a transaction will remain valid. Consensus is reached quickly. A validator is chosen by an agreed upon algorithm and there is not this uncertainty period where the network is wondering which way the blockchain will grow. The Ethereum foundation has been looking to move Ethereum from proof-of-work to proof-of-stake for a number of years now.

Another popular idea looking at scaling proof-of-work involves re-tooling these first-generation blockchains instead of replacing them with new proof-of-stake networks. This idea encompasses what are referred to as off-chain transactions and it is often called a layer two solution. A layer one solution looks at altering the main blockchain itself, like maybe changing its consensus model. During some periods, the Bitcoin network for instance has taken hours to send a transaction because it is so congested.

When the network gets this congested, transaction fees can skyrocket as well. Many transactions in the network might be sent between a small number of participants. Two cryptocurrency exchanges for instance may be sending each other thousands of transactions per day. This traffic can slow down the network and it is thought that these transactions do not need the full security offered by the main chain as they are being sent between two semi-trusted parties. By creating a contract that essentially locks away funds in a multi-signature agreement, multiple parties can send funds back and forth off-chain and not add to network congestion. Transactions carried out off-chain should also be faster and would cost less money. Essentially there is no reason for the entire network to validate every transaction between Alice and Bob if Alice and Bob are both satisfied with their agreed upon arrangement. In a situation like this, the blockchain is only used when opening or closing the account or in a situation where a dispute arises.

The class of blockchains we have looked at so far are called, public and permission-less. In a public network anyone can participate in consensus and fully utilize the network, meaning that they are able to read from and write to the blockchain. Other types of blockchains however have been developed that change some or all of these conditions. Private and permissioned blockchains may restrict participation to a semi-trusted collection of organizations and individuals. These types of blockchains are ideal for numerous applications. Because the network is smaller and more centralized, performance is usually increased. There are many use-cases for blockchains in situations where there are data accessibility restraints.

In this report we will be looking at four major topics. In the first, we will look more closely at the two most popular proof-of-work, public, permission-less blockchain networks - Bitcoin and Etheruem. In the second chapter, we will shift our focus to proof-of-stake consensus. The two proof-of-stake models being looked at here are Algorand and Casper. The third section covers off-chain protocols namely the Lightning Network and Plasma. Finally, in the fourth, we look at a popular private blockchain network called Hyperledger Fabric by studying two papers produced by IBM describing classes of applications being envisioned for Fabric. One of these papers talks about a system for health care data and the other talks about one for supply chain management. This report covers requirements associated with a directed studies course in blockchain taken with Sean Chester at the University of Victoria.

2. Proof-of-Work: Bitcoin and Ethereum

A. Bitcoin

In the paper, “Bitcoin: A Peer-to-Peer Electronic Cash System” [1], Satoshi Nakamoto introduces his ideas towards facilitating the exchange of digital value over a public, peer-to-peer network without the need for trusted intermediaries. In a trusted model,

Nakamoto talks about how transactions can never really be non-reversible. There is always the need for a ‘trusted’ third party to mediate disputes. This introduces extra costs to participants trying to use the system and because of this it does not allow for very small transactions. It also allows for fraud and potentially reversible transactions, and it therefore increases the need for trust throughout the system even causing merchants to require extra information before providing services. Uncertainty can be mitigated by using physical currency but as Nakamoto states; “no mechanism exists to make payments over a communications channel without a trusted party” [1].

Nakamoto’s Bitcoin protocol has been designed to be built on cryptographic proof instead of trust. In Bitcoin, it is computationally infeasible to reverse transactions once they have been included in a block and then accepted by the network. As other blocks are added to any block of transactions, the odds of re-building the blockchain for personal gain becomes even more and more infeasible. As long as honest nodes hold more than 50% of the network’s hashing power the Bitcoin network is supposedly secure [1]. Bitcoin solves the double-spending problem which had frustrated previous attempts to create a similar system. If Alice sends Bob a bitcoin and Carl the same bitcoin, only one of those transactions should count.

Satoshi Nakamoto defines an electronic coin as a chain of digital signatures [1]. By digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of a coin, a coin can be transferred from one user to another. By verifying the chain of ownership, someone receiving the coin can be comfortable knowing that the coin was indeed sent to them. To ensure that double spending did not take place without the use of a trusted central authority a new system was needed. Consensus must be reached in the network regarding the order of all transactions. To do accomplish this in a public, peer-to-peer setting, Nakamoto theorized that all transactions needed to be publicly announced. Every participant needed to agree to the exact same history of all transactions.

The solution introduced by Bitcoin utilizes what is called a timestamp server to achieve a total ordering of all interactions. Hashing a block of transactions along with a timestamp and publishing that hash across the network proves that the data contained in the block must have existed at that time. By including a hash of the previous block in the hash of any block, a chain is formed. The only block without a previous block is the genesis block.

To provide security for the blockchain, Nakamoto implemented a proof-of-work system based on work done by Adam Back in developing Hashcash [3]. To implement a distributed timestamp server, a proof-of-work is discovered for a block by finding a hash for that block that begins with a number of leading zeros. To achieve this, a nonce is included in a block and this nonce value is incremented until the hash value for the block meets the pre-determined criteria. This process can take a lot of CPU power and helps prevent adversaries from spamming the network. As blocks are chained together,

re-spending previous transactions means that the hashes for all blocks from the time the transaction was spent into the future need to be re-calculated. A devious organization would need to have more hash power than all honest nodes combined as the Bitcoin consensus rules stipulate that the longest chain is the canonical one. Nakamoto makes the point that if voting in the network was carried out with one vote per IP address, one entity could simply create numerous addresses. Proof-of-work provides security, authenticity and stability. The difficulty of proof-of-work is determined by a moving average looking to maintain a constant number of blocks produced every hour.

Given the effort it takes to create and verify new blocks in the network, block production was incentivized. The first transaction in any block is a special transaction that is awarded to the node that created the block. This process brings new coins into the system and is the only source of inflation. Another means of incentive is with transaction fees which are optionally added to any transaction. Nakamoto talks about how any node may find it more profitable to play by the rules rather than attack the system given the hash power that would be needed to be successful. He also talks about an end to inflation with block rewards at some point

Transactions in a Bitcoin block are stored in a Merkle Tree. The hash of every transaction is hashed with the hash of one other transaction to produce a tree like structure that ends in the root hash which is a component of the block header. In this way the network can ensure that individual transaction in the block have not been tampered with and some nodes can choose to store truncated versions of the blockchain that do not carry every transaction. Individual transactions can be checked for validity simply by examining the branch of the Merkle Tree that contains that transaction. This is called ‘simplified payment verification’ by Nakamoto. For maximum security, businesses or individuals concerned about authenticity should store a full history of a blockchain.

B. Ethereum

Ethereum is a public, peer-to-peer blockchain protocol proposed in 2013 [2] that is primarily known for adding a Turing complete programming language to a Bitcoin-like network. This is done through what is called the ‘Ethereum Virtual Machine’ or EVM for short. Through the EVM, Ethereum allows for fully operational applications and smart contracts to be deployed on a blockchain. The author of the Ethereum white paper [2], Vitalik Buterin, talks about various ideas for applications that people had been trying to implement on Bitcoin extending beyond simple payment processing. For Buterin, Bitcoin was ill-suited to being used for more complex applications for a number of reasons. According to Buterin, Ethereum was intended to, “provide a blockchain with a built-in, fully fledged Turing-complete programming language that can be used to ‘create’ contracts that can be used to encode arbitrary state transition functions” [2].

Bitcoin uses a model called UTXO or 'Unspent Transaction Output'. A UTXO can be owned not just by a public key but also by a more, "complicated script expressed in a simple stack-based programming language" [2]. One example of this would be with what is called 'multisig', where a script is that is created that depends on signatures from more than one public key in order to be activated. Script is also used to verify an elliptic curve signature against a transaction and the address that owns a UTXO. One of the main accomplishments of Ethereum is in changing the UTXO model to an account-based model. Bitcoin allowed for some programming functionality and various schemes had been proposed and even employed using it. However, the scripting functionality that could be employed in Bitcoin using unspent transactions or UTXO only allowed for a weak version of smart contracts. There were three major reasons why Buterin believed that Bitcoin was not suitable for the new class of applications being thought about at this time.

The stack-based scripting language called Script which was used by Bitcoin was not ideal for developing decentralized applications according to Buterin. For one thing, Script is not Turing-complete in that it does not allow for loops. This was at one time thought to be a feature in Bitcoin as it would help avoid errors with infinite loops. Script can theoretically provide the same level of functionality without using loops but would need to be many times more cumbersome for certain applications that required repeated functionality.

In order to prevent infinite loops, Ethereum relies on two simple concepts. These are called, STARTGAS and GASPRICE and are both represented by values that are included with each transaction. STARTGAS gives a limit to how many steps any piece of code can execute. Running code on the Ethereum blockchain uses what is called gas and even without STARTGAS specified any code would stop executing as any account would eventually run out of currency. GASPRICE indicates to the network how much the sender of a transaction is willing to pay for every computational step. If any transaction runs out of gas, all of the state changes associated with that transaction will be reversed. Fees paid however will not be returned. If a transaction executes without using all of the gas assigned to it via what is specified in STARTGAS, the remaining gas will be returned to the Sender's account. Ethereum allows for the existence of two different types of accounts. The first type is what is called and externally owned account. An externally owned account has no code but can send message by signing a transaction. The other type of account, a contract account, activates every time it receives a message. According to Buterin, "this allows it to read and write to internal storage and send other messages or create contracts in turn" [2]. This is the type of account that is used to store application code and smart contracts.

The second limitation of scripting relates to what is called 'value-blindness'. This refers to the fact that the UTXO model cannot provide fine-grained control over amounts, being only able to produce full amounts. A transaction has one or more inputs and each input contains a, "reference to an existing UTXO and a cryptographic signature

produced by the private key associated with the owner's address, and one or more outputs, with each output containing a new UTXO to be added to the state" [2]. In Bitcoin, if you have 100 BTC in your account and you want to send someone 1 BTC, you have to send the remaining 99 BTC to a new address you control. In this way you are essentially burning accounts every time you want to send a transaction. Ethereum is different because you can keep the same address and send a portion of your balance anytime you want without losing it. In this way, Ethereum chose to what is called the account/ balance model. Ethereum can provide incremental updates to states and this provides much better functionality, allowing for smart contracts and decentralized applications.

The third and final limitation provided by Buterin relates to what he refers to as the lack of state in Bitcoin because of the UTXO model [2]. UTXO is either spent or unspent, therefore it cannot be used for multi-stage contracts or scripts which keep any other internal state. This means that UTXO can only be used to build simple contracts that can only be used once. Ethereum contracts on the other hand were to be stateful and Ethereum was specifically designed to allow for the kind of functionality that could allow for creating things like the DAO (Digital Autonomous Organization), which was to be an organization whose rules existed solely as series of inter-related smart contracts.

Ethereum sought to improve scripting and provide a home for on-chain meta protocols which were developed to exist on the Bitcoin blockchain but could not do that well because of Bitcoin's design. According to Buterin, Ethereum would, "allow developers to create arbitrary consensus-based applications that have the scalability, standardization, feature-completeness, ease of development and interoperability offered by these three paradigms all at the same time" [2]. Another intended use for Ethereum was in allowing for so-called altcoins to be deployed via smart contracts on the Ethereum blockchain rather than needing to build their own blockchains.

Missing: GHOST protocol

3. Proof-of-Stake:

A. Casper

Unlike proof-of-work (PoW), proof-of-stake does not rely on solving difficult cryptographic puzzles in order for the network to reach consensus. There are many proposed ways to implement proof-of-stake but the general idea is that validators are chosen to lead any round with the odds stacked in their favor via the number of tokens that they hold. One popular offshoot is delegated proof-of-stake (DPoS). In DPoS there are only a certain number of validators in the network and token holders can vote for those validators by delegating their tokens to them.

One PoS proposal receiving a lot of attention is [4] as it being developed by the team behind Ethereum. Their PoS scheme – Casper - has been partially derived from Byzantine fault tolerance (BFT) literature. BFT algorithms can generally be proven to reach consensus if more than 2/3 of all validators are honest, regardless of network latency. Interestingly, [4] is designed as an overlay for PoW systems like Ethereum in order to offer PoS based security improvements. Casper is initially responsible for finalizing blocks while the underlying PoW consensus mechanism is responsible for proposing blocks. If attackers fully control the proposal mechanism, Casper can prevent the network from finalizing two conflicting checkpoints. Attackers however could prevent Casper from finalizing future checkpoints.

Implementing Casper on Ethereum is intended to eventually lead to fully replacing the underlying PoW proposal mechanism. Casper introduces the idea of slashing to combat undesirable behaviour in the network. PoS validators would have their stake locked away and their balances could be either diminished or even fully eliminated by the protocol. Punishing nodes is supposed to allow for things like accountability which is something that BFT algorithms generally don't support. The authors in [4] suggest that other PoS implementations are plagued by the 'nothing at stake' problem where attackers can keep attacking the system from within without fear.

By punishing stakeholders, it is claimed that Casper provides stronger security incentives than PoW even. Volunteering to lock away one's funds however might be a risky proposition. Having an attacker trick your computer into following the wrong chain could have catastrophic consequences. Validators would have to have complete trust in the system and this could affect the overall security of the network as many honest participants may choose to not participate. A discovered security flaw may lead to a large number of dishonest attackers joining the system. Bitcoin-like architectures are famously insecure in the face of a 51% attack, when over 50% of network validators are controlled by a malicious actor. PoS as described here is claiming to be secure only if 2/3 of validators are honest. Therefore, a malicious entity would only need a 1/3 stake to take over the network.

B. Algorand

Algorand is a blockchain protocol headed by former Turing award winner Silvio Micali. Algorand uses an alternate version of proof-of-stake called pure proof-of-stake (PPoS). Although Algorand has evolved since the paper, 'Algorand: Scaling Byzantine Agreements for Cryptocurrencies' was published in 2017 the basics surrounding the protocol can be found in it [2].

Algorand implements a new Byzantine Agreement protocol to establish consensus. Verifiable Random Functions are used which allow network nodes to privately check if they have been selected to participate in assembling a new block. This same mechanism allows for nodes to include a proof of their selection in their network

messages. Users therefore do not need to keep any private state except for their private keys which helps improve general security.

The protocol allows Algorand to replace participants immediately after they send a message. This helps PPoS to differentiate itself from most regular proof-of-stake schemes. In many implementations of PoS, prospective validators need to lock their funds into the system. Algorand allows Users to have free access to their funds while also enjoying the benefit of potentially receiving rewards for being chosen as a block validator. Having a larger pool of validators to choose from via the utilization of Verifiable Random Functions improves security because attacks carried out on a chosen group of validators is not as concerning anymore.

Results published [2] state that Algorand achieves a throughput over 125x greater than what can be achieved by Bitcoin with much faster block confirmation times to boot. Furthermore, the authors state that the network receives almost no penalty for scaling. The Algorand MainNet was launched in June 5 of 2019. At first glance, Algorand seems to have solved most of the issues surrounding previous PoS iterations. The network is very new however and attackers have not fully had the opportunity to exploit the system. Bitcoin is battle-tested, as it was launched in 2008. Any competitor, even if it solves the issues that plague Bitcoin relating to both throughput and energy usage, would still not have the public's trust in the way that Bitcoin does.

4. Layer 2 Solutions:

A. Lightning Network

'The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments" is a paper written by Joseph Poon and Thaddeus Dryja. A draft of their paper was first published in early 2015. Like other layer 2 solutions, the lightning network seeks to address scalability issues in public blockchain networks. Another goal of the lightning network is to create a payment system with very low transaction fees thus enabling the idea of micropayments. Transaction fees in networks like Bitcoin and Ethereum are usually too high for these very small transactions because of the large numbers of people who use them regularly. Bitcoin was originally envisioned as a solution for micropayments.

Poon and Dryja notably propose a decentralized system, "whereby transactions are sent over a network of micropayment channels whose transfer of value occurs off-blockchain" [7]. Keeping Bitcoin decentralized was a key requirement in the design of the lightning network as the authors note that, "centralization would then defeat aspects of network decentralization that make Bitcoin secure, as the ability for entities to validate the chain is what allows Bitcoin to ensure ledger accuracy and security" [7]. In their system, they wish to be able sign Bitcoin transactions with a new sighash type to address transaction malleability. This fix was implemented later in 2017 when the

Bitcoin network upgraded to Segregated Witness (SegWit). Malleability is a property in cryptography that makes it possible to transform a ciphertext into another ciphertext which decrypts to a related plaintext.

With payment channels, participants can transfer money to each other off-chain, thereby decreasing the overall load on the main Bitcoin blockchain. Because of this, these transactions are private unlike regular transactions. To open a channel, two participants commit an amount into a two-signature address with what is called a funding transaction and this transaction is published on-chain. Afterwards participants can make any number of transactions between themselves. Closing a payment channel is achieved by broadcasting the final account balances back to the blockchain.

Once a payment channel is established between two participants, each party must maintain a latest commitment transaction. A commitment transaction reflects the distribution of funds that were initially locked away in the original funding transaction. Commitment transactions are a pair of asymmetrical transactions and must be signed by both participants. A commitment transaction contains two outputs, one pays the other participant and the other is a time-locked and revocable output that eventually pays the Sender. If the Sender has the revocation key, the Sender may revoke the second output. The second transaction acts in reverse with the Receiver of the first becoming the Sender.

Closing a channel can be done in any one of three ways: collaboratively, unilaterally or because of a breach remedy. Closing a channel collaboratively is the best solution. One participant will initiate the closure and the other will agree without any dispute. If another party does not agree to closing a channel a time-lock will occur. After the time-lock expires, the funds will be free to use, the channel will close and this is how a channel is closed unilaterally. With a breach remedy, one party may try to cheat by unilaterally closing a channel with an older transaction. This results in a time-lock and the aggrieved party can claim all of the funds in the channel by using a Breach Remedy transaction. According the Dryja and Poon, “one should periodically monitor the blockchain to see if one’s counterparty has broadcast an invalidated Commitment Transaction, or delegate a third party to do so” [7]. By giving a third party (Watchtower) the Breach Remedy transaction, the third party will review the evidence and close the channel unilaterally if one party tried to publish an invalid state. This is how dishonest actors are punished.

Payment channels are created between two entities and ultimately scalability is achieved via a network of these channels. The only transactions that would need to be broadcast on-chain are those where participants disagree. Bitcoin transaction outputs are given a hash-lock and a time-lock and this makes it impossible for any participant to steal funds from another. Dryja and Poon also suggest that by using staggered timeouts, funds can be sent between numerous intermediaries. If a channel is not open between any two participants payments can be routed through the network and this can

be done with an onion routing technique if a Sender and a Receiver both have mutual peers in common.

B. Plasma

Plasma is a layer two scaling solution introduced primarily for Ethereum by Vitalik Buterin and Joseph Poon in August of 2017. Plasma is different from the Lighting Network in that it is organized as a framework and series of secondary blockchains that would exist with a tree-like structure on top of the main Ethereum blockchain. These secondary chains are called Plasma chains. Like with the Lightning Network, the idea is to use the main chain as little as possible as a means to achieve scalability, increasing throughput and lowering transaction fees to power micro-transactions. The goals of Plasma are to enable, “the blockchain to represent a significant amount of decentralized financial applications worldwide” [6]. Another goal of Plasma is to provide, “globally persistent data farms” via economic incentives.

Secondary or Plasma chains would be built with smart contracts and organized inside of a Merkle tree. Potentially, an infinite number of plasma chains can exist. Additionally, on any chain, an unlimited number of plasma chains can be created as copies of the parent chain. Plasma chains allow for flexibility because they can be theoretically tailored for unique purposes and even use their own consensus algorithms. A plasma chain could potentially use proof-of-work, proof-of-stake or even an alternative Byzantine fault-tolerant consensus algorithm like Tendermint.

Because Ethereum is essentially a distributed computing solution, Buterin and Poon sought, “to design a system whereby computation can occur off-blockchain but ultimately enforceable on-chain which is scalable to billions of computations per second with minimal on-chain updates” [6]. Proof-of-stake validation on Plasma chains is enforced by the way of fraud proofs. Ultimately, Plasma chains would rely on the present Ethereum blockchain, “to enforce transactional state transitions” [6]. A tree of Plasma chains in seen by the authors [6] to be like a court system with the final verdict resting on the main Ethereum proof-of-work blockchain. Participants could look for a resolution on respective root chains below the base layer. A system of fraud proofs is used to enforce balances and state transitions. Fraud proofs are central to Plasma and are generally used to ensure that participants can report dishonest nodes, protect their funds, and exit a transaction. Through a fraud proof, a participant on a Plasma child chain can file a complaint to the parent or root chain.

Fraud proofs use what is called an interactive funds-withdrawal protocol. To withdraw funds, Plasma requires an exit-time. Output must be confirmed using the UTXO model. Other network participants would then have the opportunity to submit a bonded proof during the exit-time period. Periodically, commitments from Plasma chains which are organized in a Merkle tree to save space, are broadcast to the root blockchain. This allows for an enormous amount if scalability as the load of the Ethereum blockchain would be minimal under normal operating conditions.

There are two key components of [6] according to Poon and Buterin. One is to reframe all blockchain computation into a set of MapReduce functions. MapReduce can be used over a tree of chains to verify data and increase overall efficiency. The other key component is optional and would allow for, “proof-of-stake token bonding on top of existing blockchains with the understanding that the Nakamoto Consensus incentives discourage block withholding” [6]. Nakamoto consensus here is another way to describe the proof-of-work consensus model introduced in [1] by Satoshi Nakamoto. This is the consensus model that was adopted by Ethereum with minor modifications [2]. The greatest challenges identified in [6] are providing data availability and countering block withholding attacks.

By facilitating exit strategies for Users to exit faulty chains and both incentivizing and enforcing correct behaviour, Buterin and Poon believe that their solution can mitigate block withholding. Block withholding refers to an attack vector where a party solves the proof-of-work challenge to build a valid block of transactions but then chooses to not broadcast that block. That same party can then build more blocks on top of their valid block in secret that may contain invalid transactions. Later, when they publish their chain and broadcast it to the network their chain will replace other work that has been done in the meantime and it therefore becomes part of the canonical version of the blockchain. This is because of the longest chain rule that forms a chief component of most proof-of-work consensus models. The authors of [6] believed that proof-of-work mitigated these attacks because there is always the possibility that other miners in the system will discover blocks faster than an attacker can, thus mitigating the attack. To successfully carry out this type of attack on a proof-of-work blockchain, an attacker would probably need to hold some 50% of all network hash power.

In the event of a block-withholding attack on a Plasma chain, “participants can rapidly and cheaply do a mass-exit, with substantial cost saving versus other previous off-chain proposals” [6]. It is also claimed that in achieving this there is no need to place any trust in so-called 3rd party validator nodes. By broadcasting a proof-of-funds on the root chain, a participant can exit the compromised chain following a delay period. The block before the withheld block will then be considered the last valid block thus rolling back the entire chain. One main issue involved in [6] relates to what is called the mass-exit problem. In the mass-exit problem, many participants may choose to exit their Plasma chain at the same time and therefore flood the root tree. In the event of any security failure, assets should fall back on to the root chain.

Ultimately for the authors of [6], the five key components of Plasma are; the incentive layer for computing contracts, the structure for arranging Plasma chains in a tree structure, the MapReduce computing framework for constructing fraud proofs, the consensus mechanism dependent on the root chain and a bitmap UTXO commitment structure for ensuring accurate state transitions to minimize mass-exit costs [6]. Unlike the Lightning Network paper which presents an almost immediately implementable system, the Plasma paper is very scattered. It mostly reads like sketches of various thoughts. At 47 pages in length, this equates to a lot of rambling. Three years after this paper was written, there have been numerous failed attempts to create a working implementation and despite this, there is still new work being done to try to realize the initial goals of the project. The Lightning Network was designed to handle payment processing and not computation and data storage. Plasma would also be a much more extensive solution, involving thousands of blockchains working together. The authors of [6] even see solutions like the Lightning Network working inside of Plasma on various Plasma chains to further help increase transaction speed and throughput. In the conclusion to their paper, Buterin and Poon even theorize that their system could handle, “all financial computation worldwide” which would be something.

5. Hyperledger Applications

Relative to other large technology companies, IBM has been fairly active in the blockchain space. Their dedication to blockchain is reflected in a number of papers that have been published in the IBM Journal of Research and Development. This report will look at four such papers, all published in 2019.

Probably the most active area of development within IBM related to blockchain is contained within the Hyperledger project. Hyperledger is an open source, cross-industry collaboration hosted by the Linux Foundation. IBM was one of the pioneers of this initiative and their stamp is most notably seen on Hyperledger Fabric. Hyperledger Fabric is a plug and play blockchain solution and is the framework behind IBM's blockchain platform. Components in Fabric like consensus and memebership services can be replaced and it is Fabric's modular architecture which has helped the project garner a considerable amount of attention.

Unlike public blockchain networks like Bitcoin and Ethereum, Hyperledger Fabric is meant to be deployed in a permissioned, semi-trusted environment. Fabric is seen as a good solution for cross-industry applications like those related to providing bank to bank financial services. In an environment like this you have participants that may not completely trust each other with concerns relating to privacy, accountability and security. Only semi-trusted entities approved by other members in the network would be allowed to join and view the shared, distributed ledger or blockchain. Permissioned

blockchain solutions are ideal for these types of scenarios and the benefits of using a dedicated blockchain platform cannot be replicated by any type of shared database.

One widely talked about use case for permissioned blockchains - like IBM's Hyperledger Fabric - relates to the area of supply chain management. Putting a class of products on a blockchain like say diamonds may have different requirements than those involved with bank to bank financial services. A supply chain blockchain may be publicly viewable as consumers' have an interest in the authenticity of a product. Bank to bank transactions, on the other hand, would most likely only be viewable by other banks and maybe some government auditing agencies. One paper that will be looked at here relates directly to the issue of supply chain management with a focus on automating what they call high-volume tasks involving reconciliations, payments, and settlements [1].

Another notable use case for blockchain technology that has received some attention is related to the distribution of health care data. Privacy concerns here are much more stringent than those involved with supply chain management. Health care data needs to be treated with the utmost care and concern as a person's health care records should only be viewable by trusted care providers. These trusted care providers should additionally only be allowed to view the portion of an individual's health care record that directly relates to any condition that is being examined and has consent associated with it. Ideally access would be time managed and not available to a health care provider after a treatment has finished. Using cryptography, a patient can digitally sign transactions that allow a provider temporary access to a portion of their medical history. Securing health care data on IBM's Hyperledger Fabric solution is looked at carefully in [3] and this is another paper being covered here.

A. Supply Chain Automation

According to Narayanaswami et al [4], there are many challenges involved with increasing blockchain adoption in a suppy chain setting. Business, management, operational, and technical challenges are all addressed in their paper. One area related to business is in designing incentive mechanisms to ensure participation. A key issue related to operational concerns is authentication outside of the blockchain, specifically before data is entered. Once a product is entered into the blockchain it can be reliably traced but what assurances do we have that it is a real diamond or a a product that has been say, certified organic. One partial solution for this is provided by the idea of 'BlockTags'. A BlockTag would be an embeddable electronic tag with optional sensors attached to physical objects to help reliably trace provenance. The authors state that BlockTags would only address provenance and introduce the idea of using analytics combined with AI to, "detect anomalies, correct for outliers, interpolate and project for missing data etc." [4]. By doing this, faulty BlockTags could also be potentially identified.

Technical challenges related to supply chains and blockchain technology identified by Narayanaswami et al [4] include solving scalability or transaction throughput issues, hardware acceleration for data encryption and dealing with the ever-increasing size of the blockchain itself. Transaction scalability is a well-know issue facing blockchain networks. Multi-enterprise loads in a blockchain setting may need to process several hundreds of transactions per minute according to the authors. This is not hard to believe. Blockchain networks are notoriously slow although setups like IBM's Hyperledger Fabric are still considerably faster than public, proof-of-work ones like Bitcoin or Ethereum. Identifying the need to accelerate data encryption is an interesting point. For many blockchain solutions, data submitted to the blockchain needs to be encrypted. Using hardware encryption is a solution but it is not quite clear from reading this paper [4] if it is an immediately viable one or not.

Blockchain size management is another idea that has been floating around the blockchain community for quite some time. Over time a blockchain could theoretically grow quite large. Solutions to this issue, identified by Narayanaswami et al [4] include truncation and, "building associated logging and indexes to enable faster access." How to implement solutions to manage this issue is still an open question in the blockchain space. One other challenge identified here [4] involves consensus. The authors state that more attention has to be paid to the algorithms that establish consensus before data is written into the blockchain" [4]. Consensus needs to be clearly defined as it relates to how participants may join or leave the network which could affect things like voting and other governance issues. How would entities be allowed access to a supply chain network? Would there be a central agency that would approve all membership changes potentially removing participants involved in certain areas of the world that conflict with another country’s political ideas?

The last idea introduced in [4] is yet another one that has garnered a fair amount of attention and for good reason. Establishing a system for blockchain interoperability could have an enormous impact. In a supply chain setting, members of one network would naturally need to be able to send and receive information to other networks. Besides for the technical issues involved with passing secure transactions between blockchain networks there is the need for data consistency to allow for this to happen in the first place. Even if two supply chains utilize the same blockchain implementation - like Hyperledger Fabric - there is still a need for cross-industry standards to be established. Another issue in this scenario, that is presented in [4], is that of authenticating membership credentials. Managing membership is one network may be difficult but how about across networks? Who authenticates membership? A federated model would probably work best here because it could be based on a quorum or circle of trust and allow for membership to be approved without central agency.

B. Blockchain for HCLS (Health Care and Life Sciences)

In [3] the authors focus on tasks related to healthcare data exchange, payments, clinical trials and the pharmaceutical supply chain. Curbera et al. believe that blockchain has the potential to disrupt current industry practices in the area of health care and life sciences (HCLS). They state that blockchain is the, "ideal technology to support major requirements for emerging healthcare and life sciences (HCLS) trends." The volume of health care data contained in electronic format is rapidy expanding. Most electronic health records (EHR) are universally available [3]. With the advent of precision medicine combined with falling costs for genome sequencing, personal genomics data is becoming more and more accesible. Another area of data is called 'exogenous' data and this type of data is harvested from, "consumer-grade wearable devices and data related to social determinants of health" [3].

In an HCLS environment, a blockchain contains a full record of consent requests, authorization, and data exchange. Data integrity, provenance, patient/owner consent, and privacy are all concerns here. Every node in the health care network as proposed in [3] would need to run in an environment enabled by the Health Insurance Portability and Accountability Act (HIPAA). There are 59 controls required by the HIPAA. The HIPAA also requires Key Management Services (KMS), User Management Services (UMS), and Log Management Services (LMS).

With HCLS data, providing patient consent is a requirement. Curbera et al state that, "Leveraging patient consent to make patient data available through the healthcare system requires a secure, globally accessible system of record" [3]. Patient consent needs to exist on very specific terms which involves the portion of a patient's health record that is being requested, the reasons for the request and the date that the authorization for access expires. In this system a patient could revoke access at any time and have a full record available to them detailing the history of all access provided. A shared, immutable ledger like that provided by a blockchain tailored specifically to this use case could theoretically provide the perfect backbone.

The authors of [3] extend the use case looked at above - related to patient consent - to 'owner-mediated health data exchange'. Owner-mediated health data could extend to the exchange of data or even datasets owned by third parties. Examples could include data from a clinical trial or de-identified patient data. Large datasets like those say related to genomics data can be exchanged via smart contracts. A hash of the data along with associated pointers are stored in the blockchain while the data itself is encrypted and split into chunks and stored elsewhere. After a contract is negotiated, metadata is released and the data requestor can then download and decrypt the data.

A system is proposed in [3] built on top of IBM's Hyperledger Fabric solution. A full description of this architecture is beyond the scope of this review. Some of the technical challenges that remain afterwards and are identified relate to privacy and security of data, scalability, and speed.

Concerning security, Curbera et al state that, "Hyperledger's native data encryption scheme is not sufficient since it does not allow defining fine-grain visibility rules per data element and per user" [3]. HIPAA regulations in the United States and similar schemes in other jurisdictions have specific data privacy requirements. The authors [3] seek to address this by adding an extra layer of data encryption. Every piece of data is encrypted with a data key for extra protection. They also use private channels which are implemented in Hyperledger Fabric to provide security during the exchange of possibly sensitive data.

With respect to scalability, the authors of [3] point out that a blockchain by design stores a copy of all data on every node in the network which naturally inhibits scalability. There solution solves this problem by splitting data into chunks and using encryption. Only metadata related to the data itself along with a hash of the data for integrity is stored on-chain. They claim that by doing this they solve the problem of scalability. There are maybe a couple of concerns here. One is that storing data off-chain is not a new solution. What they propose has been implemented before. The second concern would be that, as other blockchains have shown, eventually a blockchain will become cumbersome even without storing gigabytes of exogenous data directly on the blockchain. The Bitcoin blockchain for instance is 149 GB is size and doesn't directly store any datasets either. A secure, global system storing references to all health care data along with a full audit trail detailing all access could face similar bloat.

Speed in a blockchain setting relates to scalability. The authors of [3] believe that Hyperledger Fabric is basically fast enough on its own and any speed concerns in their system could potentially be addressed by tweaking settings in Fabric such as those that relate to the number of endorsing peers needed to approve transactions. Curbera et al. note that the extra layer of encryption could add to latency but believe that this can be addressed by hardware encryption.

A concern here might be related to the fact that Hyperledger Fabric has never been implemented in a setting similar to the one proposed for HCLS data. They claim that the amount of health care data in the world is expanding rapidly but if this trend continues in the future the number of transactions involved with exchanging this data will increase as well. Whether Hyperledger Fabric can support this type of load is an unanswered question as it had not faced any similar real-life scale yet. The security provided by Hyperledger Fabric may need to be revisited as well in this type of scenario. At some point it might be better to just host a public blockchain with stronger security guarantees like what is offered by proof-of-work blockchains based on the Bitcoin protocol. Further challenges identified by Curbera et al. relate to multi-blockchain environments, standardization of key blockchain services for HCLS data and providing building blocks to enable other industries to adopt their solution.

6. Conclusion

7. References

[1] Nakamoto, Satoshi. “Bitcoin: A peer-to-peer electronic cash system.” (2008).
[2] Buterin, Vitalik. “A next-generation smart contract and decentralized application platform.” White paper 3.37 (2017).
[3] Back, Adam. “Hashcash.” (1997).
[4] Buterin, Vitalik, and Virgil Griffith. “Casper the friendly finality gadget.” arXiv preprint arXiv:1710.09437 (2017).
[5] Gilad, Yossi, et al. "Algorand: Scaling byzantine agreements for cryptocurrencies." Proceedings of the 26th Symposium on Operating Systems Principles. 2017.
[6] Poon, Joseph, and Vitalik Buterin. "Plasma: Scalable autonomous smart contracts." White paper (2017): 1-47.
[7] Poon, Joseph, and Thaddeus Dryja. "The bitcoin lightning network: Scalable off-chain instant payments." (2016).
[8] Curbera, F., et al. "Blockchain: An enabler for healthcare and life sciences transformation." IBM Journal of Research and Development 63.2/3 (2019): 8-1. (1 citation)
[9] Narayanaswami, C., et al. "Blockchain anchored supply chain automation." IBM Journal of Research and Development 63.2/3 (2019): 7-1.