1 Background

In 2009, a developer (or developers) by the name of Satoshi Nakamoto created an electronic payment system named Bitcoin (Nakamoto, 2009). The goal of Bitcoin was simple: allow two parties to directly exchange a digital currency without the need for a trusted 3rd party (e.g., a bank) to mediate the transfer. Over the past eight years, Bitcoin has grown to be one of the most successful cryptocurrencies. It has also proved to be one of the first public-facing, fault-tolerant distributed systems capable of operating in an adversarial environment.

As of 2017, there are over 1000 different cryptocurrencies, many based on Bitcoin, with a combined market capitalization approaching $1 trillion. (CoinMarket Cap, 2017). The success of Bitcoin and the emergence of new platforms such as Ethereum that offer programmable smart contracts, is pushing the technology beyond the cryptocurrency use case and driving the growing interest in blockchain technology.

The level of interest is especially high in enterprise applications. The recently formed Ethereum Enterprise Alliance (Enterprise Ethereum Alliance, 2016) and the open-source Hyperledger project (Hyperledger, 2017) now have hundreds of members across a broad spectrum of Fortune 500 companies. Virtually every major software vendor is offering services and consulting on blockchain technology.

2 What is a Blockchain?

A blockchain can be compared to a bank ledger containing transactions. It provides information about the date, time, and amount of money or other property of interest changing ownership. Once transactions are written to the ledger, they are permanent; they can't be changed or deleted. Transactions are bundled into blocks and these blocks are linked to form the ledger, which is called blockchain. A blockchain is distributed over multiple nodes using an underlying peer-to-peer (P2P) network protocol for node discovery and communication.

The components of blockchain are discussed in detail in Section 3, but are briefly described below for context.

- Cryptography: hash functions that link blocks together providing integrity of the chain and digital signatures providing integrity for the transactions.
- Consensus Algorithm: The process by which parties to a blockchain decide on the ordering and presence of transactions on the ledger.
- Distributed Ledger: A distributed, replicated, representation of all transactions
- P2P Protocol: The protocol that manages the peer nodes of the network that support blockchain. Performs communication between nodes, flow control, node discovery, and framing.
- Smart Contracts: business rules or logic that can extend the functionality of a blockchain

2.1 What a Blockchain Does

At its core, a blockchain enables a network of peer computers (or nodes) to validate, settle, and agree on a record of transactions. It establishes a form of trust between parties that may not otherwise trust each other, and does so without relying on traditional centralized services, or trusted 3rd parties. This new form of decentralized trust has generated interest with users across a wide range of domains. Companies are investing research and development efforts into

blockchain technology, with the hopes of revamping existing processes to reduce cost while improving security, accountability, and transparency.

2.2 How it Operates

A blockchain is a replicated state machine. A given state is synchronized across a set of machines, or nodes, such that these nodes function as a single machine, despite the potential that some will fail either through normal faults or malicious activity.

The state of a blockchain is driven by incoming transactions, each transaction causing a state transition. What the state represents depends on the goal of the blockchain. In the case of cryptocurrency, the state is the balance of accounts, while in an enterprise blockchain, the state may represent other forms of information.

A transaction can be thought of as the fundamental unit of work in a blockchain. It is the input to the system, an atomic operation, that drives a state transition. It either succeeds and updates the state of the system, or it fails and is ignored. A transaction is redundantly verified by every blockchain node in the network and multiple transactions are batched into blocks for efficient processing.

The consensus protocol enables all nodes to agree on the transactions in a block and the order of the blocks ensuring identical copies of the blockchain are stored on every machine. These new blocks are cryptographically linked to the prior block to prevent them from being altered once agreed upon. The combination of a tamper-proof transaction log and a deterministic state transition ensures that all machines can compute the same state given the same transaction log.

3 Key Components of a Blockchain

3.1 Cryptography

A blockchain relies on two cryptographic primitives for many of its security properties: cryptographic hash functions and digital signatures.

3.1.1 Hash Function

A cryptographic hash function is a mathematical function that takes in an input string of any size and produces a fixed sized output. Cryptographic hash functions have two main properties that are used to secure blockchains.

1. It is possible to efficiently compute the hash function in the forward direction; however, it's nearly impossible to compute the inverse of the hash function. In other words, it's infeasible to learn any valid input from the output hash.
2. Small changes in the input to a hash function result in large and unpredictable changes in the output. As an example of the above two properties, using the SHA-256 hash function, we can create a unique fingerprint of the word "HELLOWORLD":

   ```
   sha256("HELLOWORLD") =
   '0x0b21b7db59cd154904fac6336fa7d2be1bab38d632794f281549584068cdcb74'
   ```

   The hash of the word results in a 256-bit value and it will always be the same value given the same input. However, given the 256-bit output value, it is computationally infeasible to discover the original input "HELLOWORLD" in our example. Furthermore, changing

even one character (dropping the ‘H’) in the original input results in a completely different hash value:

```
sha256(“ELLOWORLD”) =
'0x7d26a27cec234907afe7ce36266858446f4b8eebb7026982c8713d3c5d1c100e'
```

This makes it easy for hash functions to detect even the slightest change in an input string. Although it’s easy to detect changes in our simple example by looking at the text, hash functions can help to guarantee the authenticity of more complex content where changes may not be as easily noticeable.

Hash functions are used to both ensure an individual block of transactions cannot be altered and that the order of blocks in the overall blockchain remains consistent. Once a block is created it cannot be altered, and one cannot remove blocks or insert blocks into the middle of the blockchain. This is further explained in Section 3.3.

3.1.2 Digital Signature

Another cryptographic primitive used by blockchain technology is the digital signature. Cryptographic digital signatures are based on public key cryptography. They are the digital equivalent of a traditional signature but are much more secure. Essentially a digital signature provides a way for anyone to mathematically verify that a party is willing to attest to some digital content.

A digital signature requires a cryptographic key pair. The key pair consist of a private key, sometimes referred to as the signing key, and a public key, also known as the verification key. The signing key is a securely generated random value, while the verification key is generated from the signing key. The math behind the signature scheme ensures that it is computationally infeasible to reverse the process–you cannot learn the signing key from the verification key. The signing key should be securely controlled by the owner and never shared. On the other hand, the verification key can be shared with anyone and is used along with the signed content to verify the validity and authenticity of the signature.

For example, Bob is the owner of a key pair and uses his private key to sign the cryptographic hash of some content to generate a signature on that content[1]. Given the corresponding public key, the content, and the generated signature, anyone can verify that Bob (uniquely) signed the content.

In the context of the blockchain, signature verification of a transaction is a critical step. If the signature is invalid, the transaction is rejected. Signatures are primarily used to ensure that all data and state on the blockchain cannot be illegitimately modified.

Most blockchain implementations use Elliptic Curve Digital Signature Algorithm (ECDSA); ECDSA is a U.S. government standard. The algorithms behind ECDSA have undergone considerable cryptographic analysis and are considered secure and more efficient than other perhaps better known cryptographic algorithms such as RSA/Digital Signature Algorithm (DSA). (Narayanan, Bonneau, Felten, Miller, & Goldfeder, 2016)

[1] Quite often for efficiency a cryptographic hash of the content is signed, and not the content itself. If correctly implemented this does not introduce any weaknesses in security.

3.2 Consensus

Consensus makes it possible for a decentralized network of machines to agree upon and share the state of the system. It is critical in ensuring participants can trust the transactions processed on the blockchain–even when they may not trust each other. For example, say Alice transfers 10 digital tokens to Bob. What prevents Alice from transferring the same 10 digital tokens to Carl? In the cryptocurrency world, this is known as the Double Spend Problem (Double Spend Problem, 2017). Somehow the system must reconcile and share account information across a network of independent nodes to ensure Carl is not cheated out of his payment from Alice.

Before Bitcoin, it was impossible to electronically transfer digital money without relying on a centralized authority to manage the state of the system. Bitcoin cleverly solved the double spend problem and eliminated the need for a middleman by simply distributing a copy of the ledger to every node on the network, so anyone can check the state of an account.

However, this creates a new problem. How does a blockchain node know it's being sent valid information? What if messages are lost? What if a malicious actor is falsifying transactions or blocks? This problem is best illustrated by the well-known Byzantine Generals Problem (Byzantine Fault Tolerance, 2017).

In the Byzantine Generals Problem, there is a group of generals surrounding a city. The generals have the ability to conquer the city if they coordinate the time of attack. However, if they do not attack at the same time, they risk being defeated by the enemy. The generals can only coordinate through messengers with no way to verify the authenticity of the message. Messengers may be captured, preventing the delivery of a message. And traitors among the generals may deliberately send bad messages to disrupt coordination. How can this group successfully coordinate the attack without relying on a centralized authority? Bitcoin solved the Byzantine Generals Problem through a new form of consensus called Proof of Work (PoW).

3.2.1 Proof of Work (Nakamoto Consensus)

Public blockchains such as Bitcoin or Ethereum, allow anyone to participate in the consensus process as a miner. Miners compete (or effectively vote) to add new transactions to the blockchain with computing power by expending a certain amount of Central Processing Unit (CPU) cycles to solve a mathematical puzzle. This puzzle is intentionally computationally difficult to solve (Nakamoto, 2009), yet it is very easy to verify the answer.

To add a block of new transactions to the blockchain, a miner must solve the puzzle. The first miner to solve the puzzle sends (proposes) the block to the rest of the network for agreement. If the network agrees on the solution to the puzzle, the miner is rewarded for creating the block and the block is added to the blockchain (the miner wins this round of competition). Through a combination of game theory and economics (effectively betting CPU cycles, which cost money, to win the reward), PoW incentivizes consensus instead of attempting to enforce it. Essentially a miner is rewarded for securing the network.

When nodes synchronize to the network, there is a chance malicious actors may try to send invalid blocks in an attempt to double spend. To prevent this, nodes follow a simple rule of always synchronizing to the longest chain of blocks. This is because the longest chain reflects the majority vote by the network–it is the one miners have done the most work on.

Following the longest chain rule also reduces the number of protocol messages traditionally required to synchronize a large number of distributed nodes, making it possible for a public

blockchain to scale to thousands of nodes. However, the downside of using the longest chain rule is that blocks are never truly final. This is why public blockchains may advise waiting for 12 or more block confirmations to accept a transaction (Block Confirmation, 2016).

An example of why this is necessary is that there are rare instances when the chain will split under normal operation–when multiple miners solve a puzzle at virtually the same time. As noted, nodes choose to build on the longest chain. However, in this case, there are two valid choices. Based on PoW one chain will eventually win, becoming the longest chain (the probability of chains extending in parallel for more than one or two blocks is exceedingly small). The blocks in the shorter chain are often referred to as orphaned; and all transactions in these blocks are effectively invalid until they are packaged into a new block.

Thus, consensus is reached, but transaction settlement times are relatively long and transaction rates are limited.

3.2.2 Byzantine Fault Tolerant (BFT) Consensus

While public blockchains rely on PoW, enterprise (or permissioned) blockchains tend to use more traditional BFT consensus protocols (Byzantine Fault Tolerance, 2017). BFT consensus is based on the idea that a pre-selected, authorized group of validators[2] will create, verify, and attest to new blocks.

These validators take turns creating new blocks and submit a newly created block to other validators for verification and vote. Each validator votes for a block by cryptographically signing it. Once the network receives at least a 2/3 majority vote for a block, it is finalized and added to the blockchain. Since valid blocks will contain the digital signatures of the validators, nodes synchronizing to the blockchain need only check the validator signatures in a block to ensure they are following the correct blockchain.

BFT consensus usually requires a certain minimum number nodes to ensure the network can operate in the face of malicious actors, for example, 3F+1[3], where F is the number of faulty nodes. Compared to PoW, this approach also requires the exchange of more protocol messages to coordinate the consensus process, limiting its scalability. While PoW can support thousands of nodes, BFT is limited to at most hundreds.

Table 1. Consensus Comparison.

	BFT	Nakamoto
Speed (transactions per second)	**Potentially 1000s**	**< 20 on average**
Network Scalability	**100s of nodes**	**1000s of nodes**
Block Finality	**Instant**	**Eventual**

There is not a “one size fits all” consensus algorithm for blockchain technology. Selecting a consensus algorithm for a given use case will require a lot of thought and attention to detail both in the application itself and in the externalities involved.

[2] A validator is the term used for a “miner” in a blockchain using BFT.
[3] This may vary based on the specific protocol.