KTU NOTES

The learning companion.

KTU STUDY MATERIALS | SYLLABUS | LIVE NOTIFICATIONS | SOLVED QUESTION PAPERS

This paper is available at https://bitcoin.org/bitcoin.pdf.

It introduced the term **chain of blocks**. No one knows the actual identity of Satoshi Nakamoto. After introducing Bitcoin in 2009, he remained active in the Bitcoin developer community until 2011. He then handed over Bitcoin development to its core developers and simply disappeared. Since then, there has been no communication from him whatsoever, and his existence and identity are shrouded in mystery. The term "chain of blocks" evolved over the years into the word "blockchain."

As stated previously, blockchain technology incorporates a multitude of applications that can be implemented in various economic sectors. Particularly in the finance sector, significant improvement in the performance of financial transactions and settlements manifests as highly desirable time-and-cost reductions. Additional light will be shed on these aspects of blockchain in *Chapter 19*, *Blockchain – Outside of Currencies*, where practical use cases will be discussed in detail for various industries. For now, it is sufficient to say that parts of nearly all economic sectors have already realized the potential and promise of blockchain, and have embarked, or will do so soon, on the journey to capitalize on the benefits of blockchain technology.

Blockchain defined

A good place to start learning what blockchain is would be to see its definition. There are some different ways that blockchain may be defined; following are two of the most widely accepted definitions:

Layman's definition: Blockchain is an ever-growing, secure, shared recordkeeping system in which each user of the data holds a copy of the records, which can only be updated if all parties involved in a transaction agree to update.

Technical definition: Blockchain is a peer-to-peer, distributed ledger that is cryptographically secure, append-only, immutable (extremely hard to change), and updateable only via consensus or agreement among peers.

Now, let's examine things in some more detail. We will look at the keywords from the technical definition one by one.

Peer-to-peer

The first keyword in the technical definition is **peer-to-peer**, or **P2P** This means that there is no central controller in the network, and all participants (nodes) talk to each other directly. This property allows for transactions to be conducted directly among the peers without third-party involvement, such as by a bank.

Distributed ledger

Dissecting the technical definition further reveals that blockchain is a "distributed ledger," which means that a ledger is spread across the network among all peers in the network, and each peer holds a copy of the complete ledger.

Cryptographically secure

Next, we see that this ledger is "cryptographically secure," which means that cryptography has been used to provide security services that make this ledger secure against tampering and misuse. These services include non-repudiation, data integrity, and data origin authentication. You will see how this is achieved later in *Chapter 4*, *Public Key Cryptography*, which introduces the fascinating world of cryptography.

Append-only

Another property that we encounter is that blockchain is "append-only," which means that data can only be added to the blockchain in *time-sequential order*. This property implies that once data is added to the blockchain, it is almost impossible to change that data and it can be considered practically immutable. In other words, blocks added to the blockchain cannot be changed, which allows blockchain to become an immutable and tamper-proof ledger of transactions.

However, remember that it can be changed in rare scenarios wherein collusion against the blockchain network by bad actors succeeds in gaining more than 51 percent of the power. Otherwise, the blockchain is practically immutable.

There may be some legitimate reasons to change data in the blockchain once it has been added, such as the "right to be forgotten" or "right to erasure" (also defined in the GDPR ruling: https://gdpr-info.eu/art-17-gdpr/).

However, those are individual cases that need to be handled separately and that require an elegant technical solution. For all practical purposes, blockchain is indeed immutable and cannot be changed.

Updatable via consensus

The most critical attribute of a blockchain is that it is updateable only via consensus. This is what gives it the power of decentralization. In this scenario, no central authority is in control of updating the ledger. Instead, any update made to the blockchain is validated against strict criteria defined by the blockchain protocol and added to the blockchain only after a consensus has been reached among all participating peers/nodes on the network. To achieve consensus, there are various consensus facilitation algorithms that ensure all parties agree on the final state of the data on the blockchain network and resolutely agree upon it to be true. Consensus algorithms are introduced later in this chapter, and then in more detail in *Chapter 5*, *Consensus Algorithms*.

Blockchain architecture

Having detailed the primary features of blockchain, we are now in a position to begin to look at its actual architecture. We'll begin by looking at how blockchain acts as a layer within a distributed peer-to-peer network.

Blockchain by layers

Blockchain can be thought of as a layer of a distributed peer-to-peer network running on top of the internet, as can be seen in the following diagram. It is analogous to SMTP, HTTP, or FTP running on top of TCP/IP:

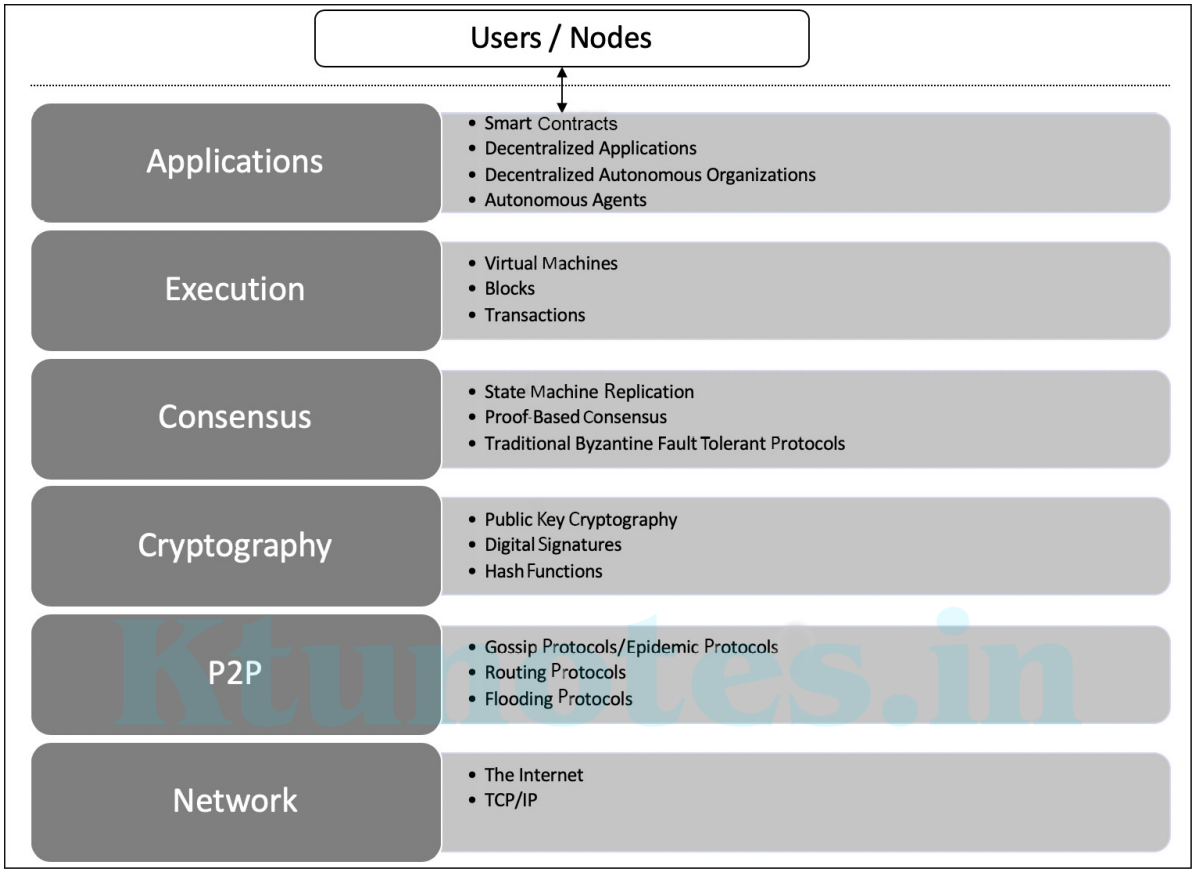

Figure 1.5: The architectural view of a generic blockchain

Now we'll discuss all these elements one by one:

- In the preceding diagram, the lowest layer is the **Network**, which is usually the internet and provides a base communication layer for any blockchain.
- A peer-to-peer network runs on top of the **Network** layer, which consists of information propagation protocols such as gossip or flooding protocols.
- After this comes the **Cryptography** layer, which contains crucial cryptographic protocols that ensure the security of the blockchain. These cryptographic protocols play a vital role in the integrity of

blockchain processes, secure information dissemination, and blockchain consensus mechanisms. This layer consists of public key cryptography and relevant components such as digital signatures and cryptographic hash functions. Sometimes, this layer is abstracted away, but it has been included in the diagram because it plays a fundamental role in blockchain operations.

- Next comes the **Consensus** layer, which is concerned with the usage of various consensus mechanisms to ensure agreement among different participants of the blockchain. This is another crucial part of the blockchain architecture, which consists of various techniques such as SMR, proof-based consensus mechanisms, or traditional (from traditional distributed systems research) Byzantine fault-tolerant consensus protocols.
- Further to this, we have the **Execution** layer, which can consist of virtual machines, blocks, transaction, and smart contracts. This layer, as the name suggests, provides executions services on the blockchain and performs operations such as value transfer, smart contract execution, and block generation. Virtual machines such as **Ethereum Virtual Machine** (**EVM**) provide an execution environment for smart contracts to execute.
- Finally, we have the **Applications** layer, which is composed of smart contracts, decentralized applications, DAOs, and autonomous agents. This layer can effectively contain all sorts of various user level agents and programs that operate on the blockchain. Users interact with the blockchain via decentralized applications. We will discuss more about decentralized applications in *Chapter 2*, *Decentralization*.

All these concepts will be discussed in detail later in this book in various chapters. Next, we'll look at blockchain from more of a business-oriented perspective.

Blockchain in business

From a business standpoint, a blockchain can be defined as a platform where peers can exchange value/e-cash using transactions without the need for a centrally trusted arbitrator. For example, for cash transfers, banks act

as a trusted third party. In financial trading, a central clearing house acts as a trusted third party between two or more trading parties. This concept is compelling, and, once you absorb it, you will realize the enormous potential of blockchain technology. This disintermediation allows blockchain to be a decentralized consensus mechanism where no single authority is in charge of the database. Immediately, you'll see a significant benefit of decentralization here, because if no banks or central clearing houses are required, then it immediately leads to cost savings, faster transaction speeds, and more trust.

We've now looked at what blockchain is at a fundamental level. Next, we'll go a little deeper and look at some of the elements that comprise a blockchain.

Generic elements of a blockchain

Now, let's walk through the generic elements of a blockchain. You can use this as a handy reference section if you ever need a reminder about the different parts of a blockchain. More precise elements will be discussed in the context of their respective blockchains in later chapters, for example, the Ethereum blockchain. The structure of a generic blockchain can be visualized with the help of the following diagram:

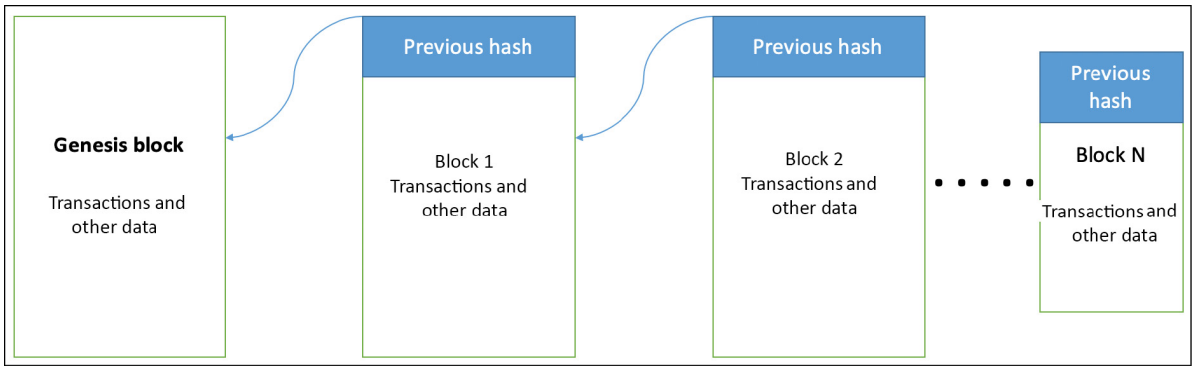

Figure 1.6: Generic structure of a blockchain

Elements of a generic blockchain are described here one by one. These are the elements that you will come across in relation to blockchain:

- **Address**: Addresses are unique identifiers used in a blockchain transaction to denote senders and recipients. An address is usually a public key or derived from a public key.
- **Transaction**: A transaction is the fundamental unit of a blockchain. A transaction represents a transfer of value from one address to another.
- **Block**: A block is composed of multiple transactions and other elements, such as the previous block hash (hash pointer), timestamp, and nonce. A block is composed of a block header and a selection of transactions bundled together and organized logically. A block contains several elements, which we introduce as follows:
    - A reference to a previous block is also included in the block unless it is a genesis block. This reference is the hash of the header of the previous block. A **genesis block** is the first block in the blockchain that is hardcoded at the time the blockchain was first started. The structure of a block is also dependent on the type and design of a blockchain.
    - A **nonce** is a number that is generated and used only once. A nonce is used extensively in many cryptographic operations to provide replay protection, authentication, and encryption. In blockchain, it's used in PoW consensus algorithms and for transaction replay protection. A block also includes the nonce value.
    - A **timestamp** is the creation time of the block.
    - **Merkle root** is a hash of all of the nodes of a Merkle tree. In a blockchain block, it is the combined hash of the transactions in the block. Merkle trees are widely used to validate large data structures securely and efficiently. In the blockchain world, Merkle trees are commonly used to allow efficient verification of transactions. Merkle root in a blockchain is present in the block header section of a block, which is the hash of all transactions in a block. This means that verifying only the Merkle root is required to verify all transactions present in the Merkle tree instead of verifying all transactions one by one. We will elaborate further on these concepts in *Chapter 4*, *Public Key Cryptography*.

- In addition to the block header, the block contains transactions that make up the block body. A **transaction** is a record of an event, for example, the event of transferring cash from a sender's account to a beneficiary's account. A block contains transactions and its size varies depending on the type and design of the blockchain.

The following structure is a simple block diagram that depicts a block. Specific block structures relative to their blockchain technologies will be discussed later in the book with greater in-depth technical detail:

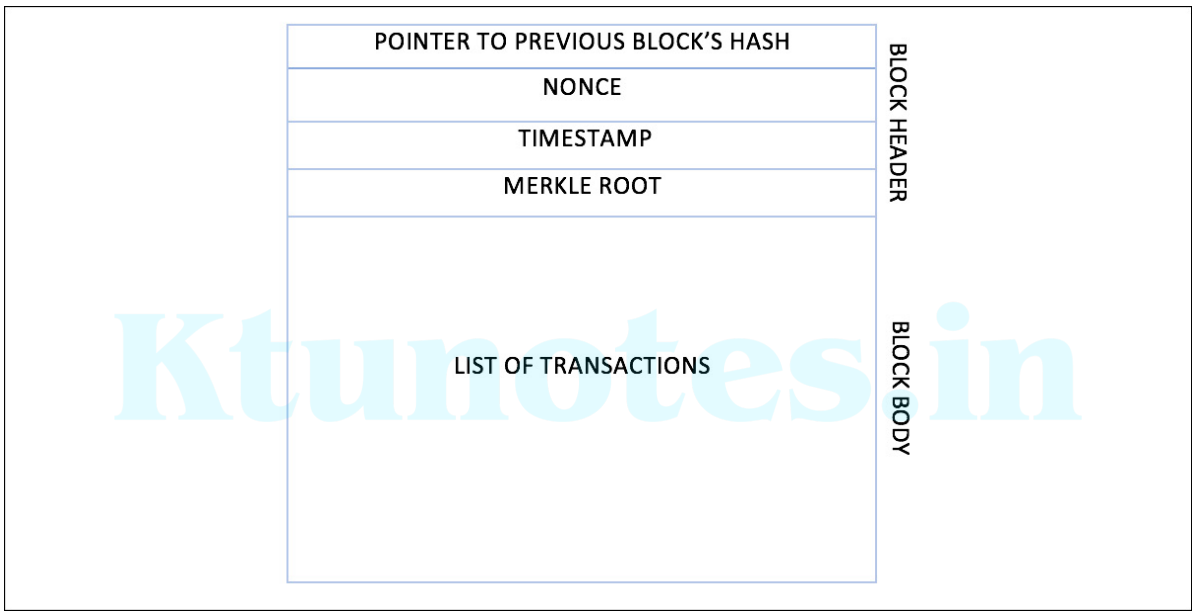

Figure 1.7: The generic structure of a block

Generally, however, there are just a few attributes that are essential to the functionality of a block: the block header, which is composed of the hash of the previous block's header, the timestamp, nonce, Merkle root, and the block body that contains the transactions. There are also other attributes in a block, but generally, the components introduced in this section are usually available in a block.

- **Peer-to-peer network**: As the name implies, a *peer-to-peer network* is a network topology wherein all peers can communicate with each other and send and receive messages.

- **The scripting or programming language**: *Scripts* or *programs* perform various operations on a transaction in order to facilitate various functions. For example, in Bitcoin, transaction scripts are predefined in a language called **Script**, which consists of sets of commands that allow nodes to transfer bitcoins from one address to another. Script is a limited language, in the sense that it only allows essential operations that are necessary for executing transactions, but it does not allow for arbitrary program development.

Think of the scripting language as a calculator that only supports standard preprogrammed arithmetic operations. As such, the Bitcoin Script language cannot be called "Turing complete." In simple words, a Turing complete language means that it can perform any computation. It is named after Alan Turing, who developed the idea of a Turing machine that can run any algorithm however complex. Turing complete languages need loops and branching capability to perform complex computations. Therefore, Bitcoin's scripting language is not Turing complete, whereas Ethereum's Solidity language is.

To facilitate arbitrary program development on a blockchain, a Turing complete programming language is needed, and it is now a very desirable feature to have for blockchains. Think of this as a computer that allows the development of any program using programming languages. Nevertheless, the security of such languages is a crucial question and an essential and ongoing research area. We will discuss this in greater detail in *Chapter 6*, *Introducing Bitcoin*, *Chapter 10*, *Smart Contracts*, and the chapters on *Ethereum Development*, later in this book.

- **Virtual machine**: This is an extension of the transaction script introduced previously. A *virtual machine* allows Turing complete code to be run on a blockchain (as smart contracts); whereas a transaction script is limited in its operation. However, virtual machines are not available on all blockchains. Various blockchains use virtual machines to run programs such as **Ethereum Virtual Machine** (**EVM**) and **Chain Virtual Machine** (**CVM**). EVM is used in the Ethereum blockchain, while CVM is a virtual machine developed for and used in an enterprise-grade blockchain called "Chain Core."

- **State machine**: A blockchain can be viewed as a state transition mechanism whereby a state is modified from its initial form to the next one by nodes on the blockchain network as a result of transaction execution.
- **Smart contracts**: These programs run on top of the blockchain and encapsulate the business logic to be executed when certain conditions are met. These programs are enforceable and automatically executable. The *smart contract* feature is not available on all blockchain platforms, but it is now becoming a very desirable feature due to the flexibility and power that it provides to blockchain applications. Smart contracts have many use cases, including but not limited to identity management, capital markets, trade finance, record management, insurance, and e-governance. Smart contracts will be discussed in more detail in *Chapter 10*, *Smart Contracts*.
- **Node**: A *node* in a blockchain network performs various functions depending on the role that it takes on. A node can propose and validate transactions and perform mining to facilitate consensus and secure the blockchain. This goal is achieved by following a **consensus protocol** (most commonly PoW). Nodes can also perform other functions such [redacted]. This asset is usually a token or virtual currency, such as Bitcoin, but it can also be any real-world asset represented on the blockchain by using tokens. There are also now standards related to tokens; for example, on Ethereum, there are ERC20, ERC721, and a few others that define the interfaces and semantics of tokenization. We will cover these in *Chapter 12*, *Further Ethereum*.

A high-level diagram of blockchain architecture highlighting the key elements mentioned previously is shown as follows:

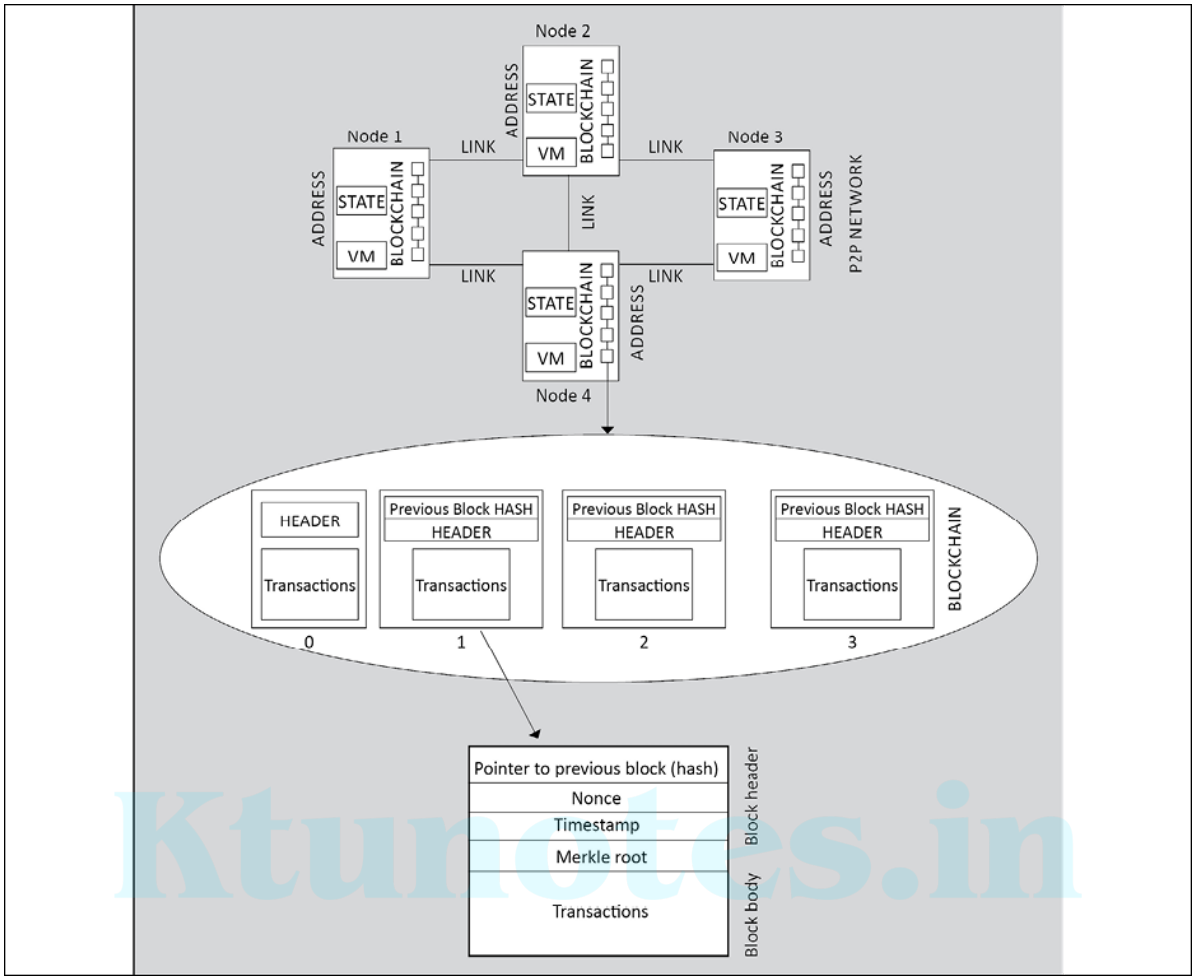

Figure 1.8: Generic structure of a blockchain network

The preceding diagram shows a four-node blockchain network (top), each ote that this is a generic structure of a blockchain; we will see specific blockchains structures in detail in the context of Ethereum and Bitcoin blockchains later in this book.

How blockchain works

We have now defined and described blockchain. Now, let's see how a blockchain actually works. Nodes are either *miners* who create new blocks

and mint cryptocurrency (coins) or *block signers* who validate and digitally sign the transactions. A critical decision that every blockchain network has to make is to figure out which node will append the next block to the blockchain. This decision is made using a *consensus mechanism*. The consensus mechanism will be described later in this chapter. For now, we will look at how a blockchain validates transactions and creates and adds blocks to grow the blockchain.

We will look at a general scheme for creating blocks. This scheme is presented here to give you a general idea of how blocks are generated and what the relationship is between transactions and blocks:

1. **Transaction is initiated**: A node starts a transaction by first creating it and then digitally signing it with its private key. A transaction can represent various actions in a blockchain. Most commonly, this is a data structure that represents the transfer of value between users on the blockchain network. The transaction data structure usually consists of some logic of transfer of value, relevant rules, source and destination addresses, and other validation information. Transactions are usually either a cryptocurrency transfer (transfer of value) or smart contract invocation that can perform any desired operation. A transaction occurs between two or more parties. This will be covered in more detail in specific chapters on Bitcoin and Ethereum later in the book.
2. **Transaction is validated and broadcast**: A transaction is propagated (broadcast) usually by using data-dissemination protocols, such as *Gossip protocol*, to other peers that validate the transaction based on preset validity criteria. Before a transaction is propagated, it is also verified to ensure that it is valid.
3. **Find new block**: When the transaction is received and validated by special participants called miners on the blockchain network, it is included in a block, and the process of mining starts. This process is also sometimes referred to as "finding a new block." Here, nodes called miners race to finalize the block they've created by a process known as mining.
4. **New block found**: Once a miner solves a mathematical puzzle (or fulfills the requirements of the consensus mechanism implemented in a

blockchain), the block is considered "found" and finalized. At this point, the transaction is considered confirmed. Usually, in cryptocurrency blockchains such as Bitcoin, the miner who solves the mathematical puzzle is also rewarded with a certain number of coins as an incentive for their effort and the resources they spent in the mining process.

5. **Add new block to the blockchain**: The newly created block is validated, transactions or smart contracts within it are executed, and it is propagated to other peers. Peers also validate and execute the block. It now becomes part of the blockchain (ledger), and the next block links itself cryptographically back to this block. This link is called a hash pointer.

This process can be visualized in the diagram as follows:

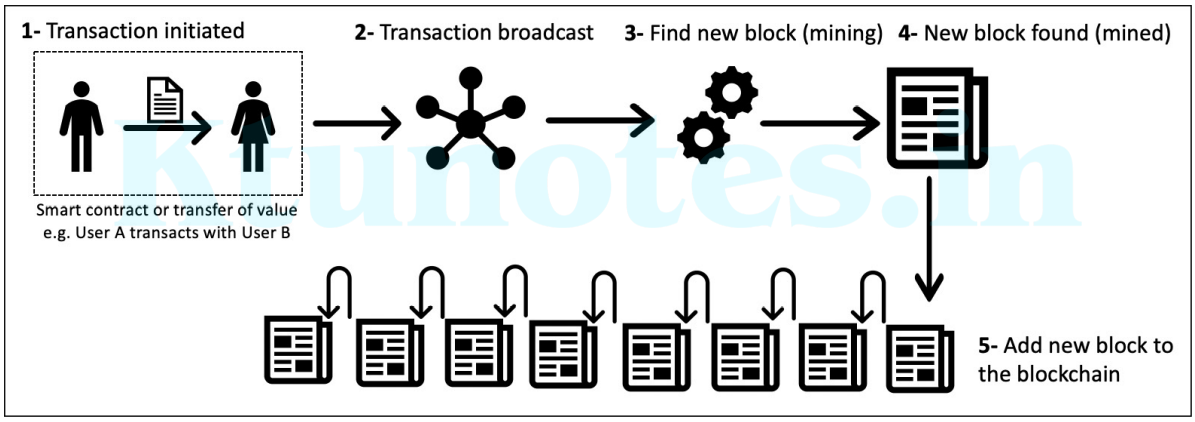

Figure 1.9: How a block is generated

This completes the basic introduction to blockchain. In the next section, you will learn about the benefits and limitations of this technology.

Benefits, features, and limitations of blockchain

Numerous advantages of blockchain technology have been discussed in many industries and proposed by thought leaders around the world who are

participating in the blockchain space. The notable benefits of blockchain technology are as follows:

1. **Decentralization**: This is a core concept and benefit of blockchain. There is no need for a trusted third party or intermediary to validate transactions; instead, a consensus mechanism is used to agree on the validity of transactions.
2. **Transparency and trust**: As blockchains are shared and everyone can see what is on the blockchain, this allows the system to be transparent. As a result, trust is established. This is more relevant in scenarios such as the disbursement of funds or benefits where personal discretion in relation to selecting beneficiaries needs to be restricted.
3. **Immutability**: Once the data has been written to the blockchain, it is extremely difficult to change it back. It is not genuinely immutable, but because changing data is so challenging and nearly impossible, this is seen as a benefit to maintaining an immutable ledger of transactions.
4. **High availability**: As the system is based on thousands of nodes in a peer-to-peer network, and the data is replicated and updated on every node, the system becomes highly available. Even if some nodes leave the network or become inaccessible, the network as a whole continues to work, thus making it highly available. This redundancy results in high availability.
5. **Highly secure**: All transactions on a blockchain are cryptographically secured and thus provide network integrity. Any transactions posted from the nodes on the blockchain are verified based on a predetermined set of rules. Only valid transactions are selected for inclusion in a block. The blockchain is based on proven cryptographic technology that ensures the integrity and availability of data. Generally, confidentiality is not provided due to the requirements of transparency. This limitation is the leading barrier to its adoption by financial institutions and other industries that require the privacy and confidentiality of transactions. As such, the privacy and confidentiality of transactions on the blockchain are being researched very actively, and advancements are already being made. It could be argued that, in many situations, confidentiality is not needed and transparency is preferred. For example, with Bitcoin, confidentiality is not an absolute

requirement; however, it is desirable in some scenarios. A more recent example is Zcash (https://z.cash), which provides a platform for conducting anonymous transactions. Other security services, such as non-repudiation and authentication, are also provided by blockchain, as all actions are secured using private keys and digital signatures.

6. **Simplification of current paradigms**: The current blockchain model in many industries, such as finance or health, is somewhat disorganized. In this model, multiple entities maintain their own databases and data sharing can become very difficult due to the disparate nature of the systems. However, as a blockchain can serve as a single shared ledger among many interested parties, this can result in simplifying the model by reducing the complexity of managing the separate systems maintained by each entity.
7. **Faster dealings**: In the financial industry, especially in post-trade settlement functions, blockchain can play a vital role by enabling the quick settlement of trades. Blockchain does not require a lengthy process of verification, reconciliation, and clearance because a single version of agreed-upon data is already available on a shared ledger between financial organizations.
8. **Cost-saving**: As no trusted third party or clearing house is required in the blockchain model, this can massively eliminate overhead costs in the form of the fees, which are paid to such parties.
9. **Platform for smart contracts**: A blockchain is a platform on which programs can run that execute business logic on behalf of the users. This is a very useful feature but not all blockchains have a mechanism to execute *smart contracts*; however, this is a very desirable feature. It is available on newer blockchain platforms such as Ethereum and MultiChain, but not on Bitcoin.

Smart contracts

Blockchain technology provides a platform for running smart contracts. These are automated, autonomous programs that reside on the blockchain network and encapsulate the business logic and code needed to execute a required function when certain conditions are met. For example, think about an insurance contract where a claim is paid to the traveler if the flight is canceled. In the real world, this

process normally takes a significant amount of time to make the claim, verify it, and pay the insurance amount to the claimant (traveler). What if this whole process were automated with cryptographically-enforced trust, transparency, and execution so that as soon as the smart contract received a feed that the flight in question has been canceled, it automatically triggers the insurance payment to the claimant? If the flight is on time, the smart contract pays itself.

This is indeed a revolutionary feature of blockchain, as it provides flexibility, speed, security, and automation for real-world scenarios that can lead to a completely trustworthy system with significant cost reductions. Smart contracts can be programmed to perform any actions that blockchain users need and according to their specific business requirements.

10. **Smart property**: It is possible to link a digital or physical asset to the blockchain in such a secure and precise manner that it cannot be claimed by anyone else. You are in full control of your asset, and it cannot be double-spent or double-owned. Compare this with a digital music file, for example, which can be copied many times without any controls. While it is true that many **Digital Rights Management** (**DRM**)schemes are being used currently along with copyright laws, none of them are enforceable in the way a blockchain-based DRM can be. Blockchain can provide digital rights management functionality in such a way that it can be enforced fully. There are famously broken DRM schemes that looked great in theory but were hacked due to one limitation or another. One example is the Oculus hack: http://www.wired.co.uk/article/oculus-rift-drm-hacked. Another example is the PS3 hack; also, copyrighted digital music, films, and e-books are routinely shared on the internet without any limitations. We have had copyright protection in place for many years, but digital piracy refutes all attempts to fully enforce the law. On a blockchain, however, if you own an asset, no one else can claim it unless you decide to transfer it. This feature has far-reaching implications, especially in DRM and e-cash systems where double-spend detection is a crucial requirement. The double-spend problem was first solved without the requirement of a trusted third party in Bitcoin.

As with any technology, some challenges need to be addressed in order to make a system more robust, useful, and accessible. Blockchain technology is no exception. In fact, much effort is being made in both academia and industry to overcome the challenges posed by blockchain technology. The most sensitive blockchain problems are as follows:

- **Scalability**: Currently, blockchain networks are not as scalable as, for example, current financial networks. This is a known area of concern and a very ripe area for research.
- **Adoption**: Often, blockchain is seen as a nascent technology. Even though this perspective is rapidly changing, there is still a long way to go before the mass adoption of this technology. The challenge here is to allow blockchain networks to be easier to use so that adoption can increase. In addition, several other challenges such as scalability (introduced previously) exist, which must be solved in order to increase adoption.
- **Regulation**: Due to its decentralized nature, regulation is almost impossible on blockchain. This is sometimes seen as a barrier toward adoption because, traditionally, due to the existence of regulatory authorities, consumers have a certain level of confidence that if something goes wrong they can hold someone accountable. However, in blockchain networks, no such regulatory authority and control exists, which is an inhibiting factor for many consumers.
- **Relatively immature technology**: As compared to traditional IT systems that have benefited from decades of research, blockchain is still a new technology and requires a lot of research to achieve maturity.
- **Privacy and confidentiality**: Privacy is a concern on public blockchains such as Bitcoin where everyone can see every single transaction. This transparency is not desirable in many industries such as the financial, law, or medical sectors. This is also a known concern and a lot of valuable research with some impeccable solutions has already been developed. However, further research is still required to drive the mass adoption of blockchain.

All of these issues and possible solutions will be discussed in detail in *Chapter 21*, *Scalability and Other Challenges*.

You now know the basics of blockchain and its benefits and limitations. Now, let's take a look at the various types of blockchain that exist.

Types of blockchain

Based on the way that blockchain has evolved over the last few years, it can be divided into multiple categories with distinct, though sometimes partially overlapping attributes. You should note that the tiers described earlier in the chapter are a different concept, whereby the logical categorization of blockchain, based upon its evolution and usage, is presented.

In this section, we will examine the different types of blockchains from a technical and business use perspective. These blockchain types can occur on any blockchain tier, as there is no direct relationship between those tiers mentioned earlier and the various types of blockchain.

In this section, we'll examine:

- Distributed ledgers
- Distributed Ledger Technology (DLT)
- Blockchains
- Ledgers

Distributed ledgers

First, I need to clarify an ambiguity. It should be noted that a *distributed ledger* is a broad term describing shared databases; hence, all blockchains technically fall under the umbrella of shared databases or distributed ledgers. Although all blockchains are fundamentally distributed ledgers, all distributed ledgers are not necessarily blockchains.

A critical difference between a distributed ledger and a blockchain is that a distributed ledger does not necessarily consist of blocks of transactions to keep the ledger growing. Rather, a blockchain is a special type of shared database that is comprised of blocks of transactions. An example of a distributed ledger that does not use blocks of transactions is R3's Corda (https://www.corda.net). Corda is a distributed ledger that is developed to record and manage agreements and is especially focused on the financial services industry. On the other hand, more widely known blockchains like Bitcoin and Ethereum make use of blocks to update the shared database.

As the name suggests, a distributed ledger is distributed among its participants and spread across multiple sites or organizations. This type of ledger can be either private or public. The fundamental idea here is that, unlike many other blockchains, the records are stored contiguously instead of being sorted into blocks. This concept is used in Ripple, which is a blockchain- and cryptocurrency-based global payment network.

Distributed Ledger Technology

It should be noted that over the last few years, the terms distributed ledger or DLT have grown to be commonly used to describe blockchain in the finance industry. Sometimes, blockchain and DLT are used interchangeably. Though this is not entirely accurate, it is how the term has evolved recently, especially in the finance sector. In fact, DLT is now a very active and thriving area of research in the financial sector. From a financial sector point of view, DLTs are permissioned blockchains that are used by consortiums. DLTs usually serve as a shared database, with all participants known and verified. They do not have a cryptocurrency and do not require mining to secure the ledger.

At a broader level, DLT is an umbrella term that represents Distributed Ledger Technology as a whole, comprising of blockchains and distributed ledgers of different types.

Public blockchains

As the name suggests, public blockchains are not owned by anyone. They are open to the public, and anyone can participate as a node in the decision-making process. Users may or may not be rewarded for their participation. All users of these "permissionless" or "un-permissioned" ledgers maintain a copy of the ledger on their local nodes and use a distributed consensus mechanism to decide the eventual state of the ledger. Bitcoin and Ethereum are both considered public blockchains.

Private blockchains

As the name implies, private blockchains are just that—private. That is, they are open only to a consortium or group of individuals or organizations who have decided to share the ledger among themselves. There are various blockchains now available in this category, such as Kadena and Quorum. Optionally, both of these blockchains can also run in public mode if required, but their primary purpose is to provide a private blockchain.

Semi-private blockchains

With semi-private blockchains, part of the blockchain is private and part of it is public. Note that this is still just a concept today, and no real-world proofs of concept have yet been developed. With a semi-private blockchain, the private part is controlled by a group of individuals, while the public part is open for participation by anyone.

This hybrid model can be used in scenarios where the private part of the blockchain remains internal and shared among known participants, while the public part of the blockchain can still be used by anyone, optionally allowing mining to secure the blockchain. This way, the blockchain as a whole can be secured using PoW, thus providing consistency and validity for both the private and public parts. This type of blockchain can also be called a "semi-decentralized" model, where it is controlled by a single entity but still allows for multiple users to join the network by following appropriate procedures.

Sidechains

More precisely known as "pegged sidechains," this is a concept whereby coins can be moved from one blockchain to another and then back again. Typical uses include the creation of new *altcoins* (alternative cryptocurrencies) whereby coins are burnt as a proof of an adequate stake. "Burnt" or "burning the coins" in this context means that the coins are sent to an address that is un-spendable, and this process makes the "burnt" coins irrecoverable. This mechanism is used to bootstrap a new currency or introduce scarcity, which results in the increased value of the coin.

This mechanism is also called "Proof of Burn" and is used as an alternative method for distributed consensus to PoW and **Proof of Stake** (**PoS**). The example provided previously for burning coins applies to a **one-way pegged sidechain**. The second type is called a **two-way pegged sidechain**, which allows the movement of coins from the main chain to the sidechain and back to the main chain when required.

This process enables the building of smart contracts for the Bitcoin network. Rootstock is one of the leading examples of a sidechain, which enables smart contract development for Bitcoin using this paradigm. It works by allowing a two-way peg for the Bitcoin blockchain, and this results in much faster throughput.

Permissioned ledger

A *permissioned ledger* is a blockchain where participants of the network are already known and trusted. Permissioned ledgers do not need to use a distributed consensus mechanism; instead, an agreement protocol is used to maintain a shared version of the truth about the state of the records on the blockchain. In this case, for verification of transactions on the chain, all verifiers are already preselected by a central authority and, typically, there is no need for a mining mechanism.

By definition, there is also no requirement for a permissioned blockchain to be private, as it can be a public blockchain but with regulated access control. For example, Bitcoin can become a permissioned ledger if an access control layer is introduced on top of it that verifies the identity of a user and then allows access to the blockchain.

Shared ledger

This is a generic term that is used to describe any application or database that is shared by the public or a consortium. Generally, all blockchains fall into the category of a shared ledger.

Fully private and proprietary blockchains

There is no mainstream application of these types of blockchains, as they deviate from the core concept of decentralization in blockchain technology. Nonetheless, in specific private settings within an organization, there could be a need to share data and provide some level of guarantee of the authenticity of the data.

An example of this type of blockchain might be to allow for collaboration and the sharing of data between various government departments. In that case, no complex consensus mechanism is required, apart from simple SMR and an agreement protocol with known central validators. Even in private blockchains, tokens are not really required, but they can be used as a means of transferring value or representing some real-world assets.

Tokenized blockchains

These blockchains are standard blockchains that generate cryptocurrency as a result of a consensus process via mining or initial distribution. Bitcoin and Ethereum are prime examples of this type of blockchain.

Tokenless blockchains

These blockchains are designed in such a way that they do not have the basic unit for the transfer of value. However, they are still valuable in situations where there is no need to transfer value between nodes and only the sharing of data among various trusted parties is required. This is similar to fully private blockchains, the only difference being that the use of tokens is not required. This can also be thought of as a shared distributed ledger used for storing and sharing data between the participants. It does have its

benefits when it comes to immutability, tamper proofing, security, and consensus-driven updates but is not used for a common blockchain application of value transfer or cryptocurrency. Most of the permissioned blockchains can be seen as an example of tokenless blockchains, for example, Hyperledger Fabric or Quorum. Tokens can be built on these chains as an application, but intrinsically these blockchains do not have a token associated with them.

All the aforementioned terminologies are used in literature, but fundamentally all these blockchains are distributed ledgers and fall under the top-level category of DLTs. We can view these different types in the simple chart as follows:

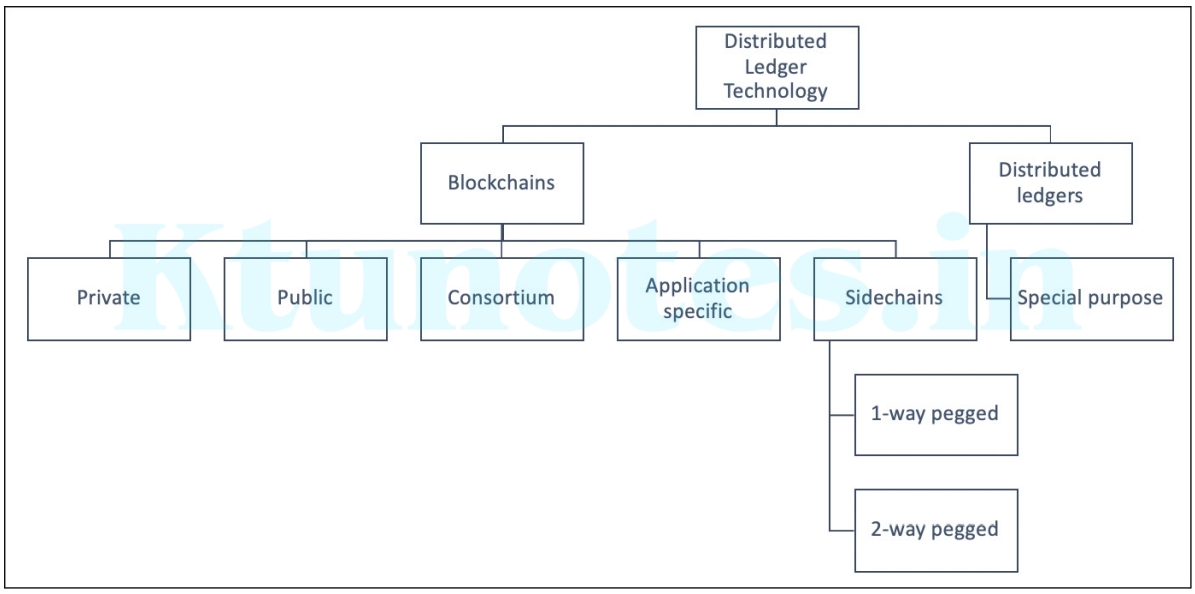

Figure 1.10: DLT hierarchy

This ends our examination of the various types of blockchain. We'll now move on to the next section to discuss the concept of consensus.

Consensus

Consensus is the backbone of a blockchain, as it provides the

The choice of the **consensus algorithm** to utilize is governed by the type of blockchain in use; that is, not all consensus mechanisms are suitable for all types of blockchains. For example, in public permissionless blockchains, it would make sense to use PoW instead of mechanisms that are more suitable for permissioned blockchains, such as **Proof of Authority** (**PoA**) or traditional Byzantine fault-tolerant consensus mechanisms. Therefore, it is essential to choose an appropriate consensus algorithm for a particular blockchain project.

Consensus mechanism

For more than three decades, this concept has been researched by computer scientists in industry and academia. With the advent of blockchain and Bitcoin, consensus mechanisms have come into the limelight again and gained considerable popularity.

- **Agreement**: All honest nodes decide on the same value.
- **Integrity**: This is a requirement that no node can make the decision more than once in a single consensus cycle.

- **Validity**: The value agreed upon by all honest nodes must be the same as the initial value proposed by at least one honest node.
- **Fault tolerant**: The consensus algorithm should be able to run correctly in the presence of faulty or malicious nodes (Byzantine nodes).
- **Termination**: All honest nodes terminate the execution of the consensus process and eventually reach a decision.

Having seen these general requirements, we'll now look at the different types of consensus mechanisms.

Types of consensus mechanisms

All consensus mechanisms are developed to deal with faults in a distributed system and to allow distributed systems to reach a final state of agreement. There are two general categories of consensus mechanisms. These categories deal with all types of faults (fail-stop types or arbitrary). These common types of consensus mechanisms are as follows:

- **Proof-based consensus mechanisms**: This arrangement requires nodes to compete in a leader-election lottery, and the node that wins proposes the final value. The algorithm works on the principle of providing proof of some work and the possession of some authority or tokens to win the right of proposing the next block. For example, the PoW mechanism used in Bitcoin falls into this category, where a miner who solves the computational puzzle as proof of computational effort expended wins the right to add the next block to the blockchain.
- **Traditional fault tolerance-based**: With no compute-intensive operations, such as partial hash inversion (as in Bitcoin PoW), this type of consensus mechanism relies on a simple scheme of nodes that publish and verify signed messages in a number of phases. Eventually, when a certain number of messages are received over a period of rounds (phases), then an agreement is reached.

To achieve fault tolerance, replication is used. This is a standard and widely used method to achieve fault tolerance. In general, there are two types of faults that a node can experience:

- **Fail-stop faults**: This type of fault occurs when a node merely has crashed. Fail-stop faults are the easier ones to deal with of the two fault types. Paxos or the RAFT protocol, introduced earlier in this chapter, are normally used to deal with this type of fault. These faults are simpler to deal with.
- **Byzantine faults**: The second type of fault is one where the faulty node exhibits malicious or inconsistent behavior arbitrarily. This type is difficult to handle since it can create confusion due to misleading information. This can be a result of an attack by adversaries, a software bug, or data corruption. SMR protocols such as **Practical Byzantine Fault Tolerance** (**PBFT**) was developed to address this second type of faults.

Many other implementations of consensus protocols have been proposed in

It was introduced by **Leslie Lamport** in 1989. With

An alternative to Paxos is RAFT, which works by assigning any of three states; that is, *Follower*, *Candidate*, or *Leader* to the nodes.

We will briefly touch on some aspects of consensus in blockchain now, but more detail on the theory of consensus mechanisms from a distributed system point of view and also from the blockchain perspective will be presented in *Chapter 5*, *Consensus Algorithms*.

Consensus in blockchain

Consensus is a distributed computing concept that has been used in This concept was previously discussed in the *distributed systems* section of this chapter. In this section, we will address consensus in the context of blockchain technology. Some concepts presented following are still relevant to the distributed systems theory, but they are explained from a blockchain perspective.

Roughly, the following describes the two main categories of consensus mechanisms:

1. .
2. .

As there is significant research being conducted in this area, new types of consensus mechanisms are also emerging, such as the , which is used in the Ripple network. The Ripple network will be discussed in detail in this book's online content pages, here: https://static.packt-cdn.com/downloads/Altcoins_Ethereum_Projects_and_More_Bonus_Content.pdf. There are also various other proposals out there, which are trying to find the right balance between scalability and performance. Some notable projects include PBFT, Hybrid BFT, BlockDAG, Tezos, Stellar, and GHOST.

The consensus algorithms available today, or that are being researched in the context of blockchain, are presented as follows. The following is not an exhaustive list, but it includes all notable algorithms:

- **Proof of Work** (**PoW**): This type of consensus mechanism relies on proof that adequate computational resources have been spent before proposing a value for acceptance by the network. This scheme is used in Bitcoin, Litecoin, and other cryptocurrency blockchains. Currently, it is the only algorithm that has proven to be astonishingly successful against any collusion attacks on a blockchain network, such as the Sybil attack. The Sybil attack will be discussed in *Chapter 6*, *Introducing Bitcoin*.
- **Proof of Stake** (**PoS**): This algorithm works on the idea that a node or user has an adequate stake in the system; that is, the user has invested enough in the system so that any malicious attempt by that user would outweigh the benefits of performing such an attack on the network. This idea was first introduced by Peercoin, and it is going to be used in the Ethereum blockchain version called *Serenity*. Another important concept in PoS is **coin age**, which is a criterion derived from the amount of time and number of coins that have not been spent. In this
- **Delegated Proof of Stake** (**DPoS**): This is an innovation over standard PoS, whereby each node that has a stake in the system can delegate the validation of a transaction to other nodes by voting. It is used in the BitShares blockchain.
- **Proof of Elapsed Time** (**PoET**): Introduced by Intel in 2016, PoET uses a **Trusted Execution Environment** (**TEE**) to provide randomness and safety in the leader-election process via a guaranteed wait time. It requires the Intel **SGX** (**Software Guard Extensions**) processor to provide the security guarantee for it to be secure. This concept is discussed in more detail in *Chapter 17*, *Hyperledger*, in the context of the Intel Sawtooth Lake blockchain project.
- **Proof of Deposit** (**PoD**): In this case, nodes that wish to participate in the network have to make a security deposit before they can mine and propose blocks. This mechanism is used in the Tendermint blockchain.

- **Proof of Importance** (**PoI**): This idea is significant and different from PoS. PoI not only relies on how large a stake a user has in the system, but it also monitors the usage and movement of tokens by the user in order to establish a level of trust and importance. It is used in the NEM coin blockchain. More information about this coin is available from NEM's website (https://nem.io).
- **Federated consensus or federated Byzantine consensus**: This mechanism is used in the stellar consensus protocol. Nodes in this protocol retain a group of publicly-trusted peers and propagate only those transactions that have been validated by the majority of trusted nodes.
- **Reputation-based mechanisms**: As the name suggests, a leader is elected by the reputation it has built over time on the network. It is based on the votes of other members.
- **Practical Byzantine Fault Tolerance** (**PBFT**): This mechanism achieves SMR, which provides tolerance against Byzantine nodes. Various other protocols, including PBFT, PAXOS, RAFT, and **Federated Byzantine Agreement** (**FBA**), are also being used or have been proposed for use in many different implementations of distributed systems and blockchains.
- **Proof of Activity** (**PoA**): This scheme is a combination of PoS and PoW, which ensures that a stakeholder is selected in a pseudorandom but uniform fashion. This is a comparatively more energy-efficient mechanism as compared to PoW. It utilizes a new concept called "Follow the Satoshi." In this scheme, PoW and PoS are combined together to achieve consensus and a good level of security. This scheme is more energy efficient as PoW is used only in the first stage of the mechanism; after the first stage, it switches to PoS, which consumes negligible energy. We will discuss these ideas further in *Chapter 7*, *Bitcoin Network and Payments*, where protocols are reviewed in the context of advanced Bitcoin protocols.
- **Proof of Capacity** (**PoC**): This scheme uses hard disk space as a resource to mine the blocks. This is different from PoW, where CPU resources are used. In PoC, hard disk space is utilized for mining and, as such, is also known as *hard drive mining*. This concept was first introduced in the BurstCoin cryptocurrency.

- **Proof of Storage**: This scheme allows for the outsourcing of storage capacity. This scheme is based on the concept that a particular piece of data is probably stored by a node, which serves as a means to participate in the consensus mechanism. Several variations of this scheme have been proposed, such as Proof of Replication, Proof of Data Possession, Proof of Space, and Proof of Space-time.
- **Proof of Authority** (**PoA**): This scheme utilizes the identity of the participants called validators as a stake on the network. Validators are known and have the authority to propose new blocks. Validators propose the new blocks and validate them as per blockchain rules. Commonly used PoA algorithms are Clique and Aura.

Some prominent protocols in blockchain will be discussed in detail in *Chapter 5*, *Consensus Algorithms*. In this chapter, a light introduction is presented only.

Remember, earlier in this chapter we said that distributed systems are difficult to build and a distributed system cannot have consistency, availability, and partition tolerance at the same time. This is a proven result; however, in blockchain, it seems that this theorem is somehow violated. In the next section, we will introduce the CAP theorem formally and discuss why blockchain appears to achieve all three properties simultaneously.

CAP theorem and blockchain

The **CAP theorem**, also known as Brewer's theorem, was introduced by Eric Brewer in 1998 as a conjecture. In 2002, it was proven as a theorem by Seth Gilbert and Nancy Lynch. The theorem states that any distributed system cannot have consistency, availability, and partition tolerance simultaneously:

- **Consistency** is a property that ensures that all nodes in a distributed system have a single, current, and identical copy of the data.

2

Decentralization

Decentralization is not a new concept. It has been in use in strategy, management, and government for a long time. The basic idea of decentralization is to distribute control and authority to the peripheries of an organization instead of one central body being in full control of the organization. This configuration produces several benefits for organizations, such as increased efficiency, expedited decision making, better motivation, and a reduced burden on top management.

In this chapter, the concept of decentralization will be discussed in the context of blockchain. The fundamental basis of blockchain is that no single central authority is in control of the network. This chapter will present examples of various methods of decentralization and ways to achieve it. Furthermore, the decentralization of the blockchain ecosystem, decentralized applications, and platforms for achieving decentralization will be discussed in detail. Many exciting applications and ideas emerge from the decentralized blockchain technology, such as decentralized finance and decentralized identity, which will be introduced in this chapter.

Decentralization using blockchain

Decentralization is a core benefit and service provided by blockchain . This model

allows anyone to compete to become the decision-making authority. A consensus mechanism governs this competition, and the most famous method is known as **Proof of Work** (**PoW**).

Decentralization is applied in varying degrees from a semi-decentralized model to a fully decentralized one depending on the requirements and circumstances. Decentralization can be viewed from a blockchain perspective as a mechanism that provides a way to remodel existing applications and paradigms, or to build new applications, to give full control to users.

Information and communication technology (**ICT**) has conventionally been based on a centralized paradigm whereby database or application servers are under the control of a central authority, such as a system administrator. With Bitcoin and the advent of blockchain technology, this model has changed, and now the technology exists to allow anyone to start a decentralized system and operate it with no single point of failure or single trusted authority. It can either be run autonomously or by requiring some human intervention, depending on the type and model of governance used in the decentralized application running on the blockchain.

The following diagram shows the different types of systems that currently exist: central, distributed, and decentralized. This concept was first published by Paul Baran in *On Distributed Communications: I. Introduction to Distributed Communications Networks* (Rand Corporation, 1964):

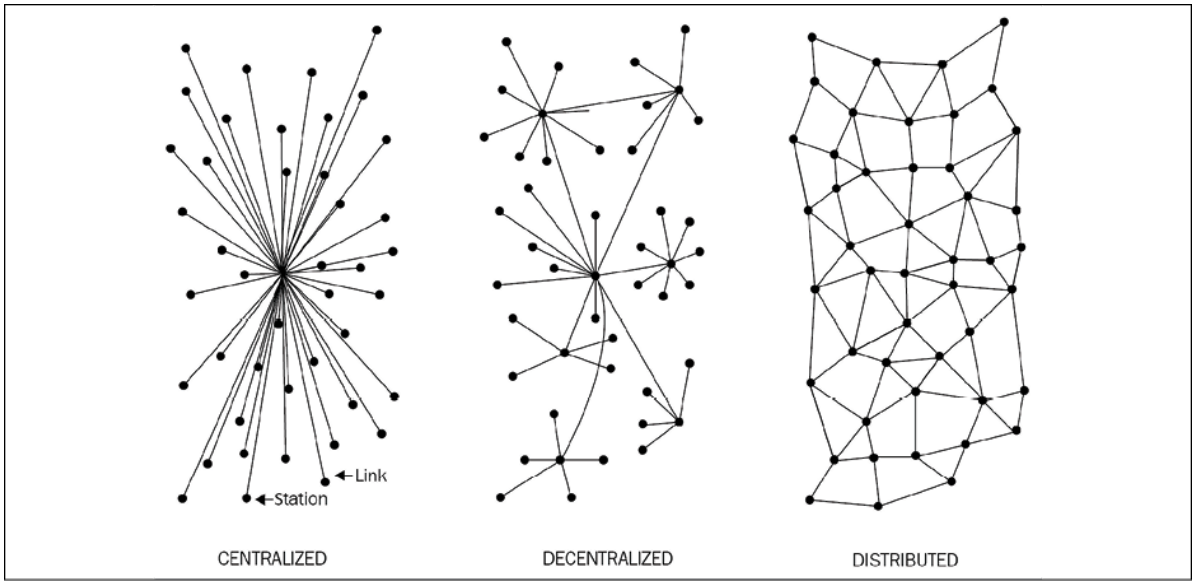

Figure 2.1: Different types of networks/systems

Centralized systems are conventional (client-server) IT systems in which there is a single authority that controls the system, and who is solely in charge of all operations on the system. All users of a centralized system are dependent on a single source of service. The majority of online service providers, including Google, Amazon, eBay, and Apple's App Store, use this conventional model to deliver services.

In a **distributed system**, data and computation are spread across multiple nodes in the network. Sometimes, this term is confused with *parallel computing*. While there is some overlap in the definition, the main difference between these systems is that in a parallel computing system, computation is performed by all nodes simultaneously in order to achieve the result; for example, parallel computing platforms are used in weather research and forecasting, simulation, and financial modeling. On the other hand, in a distributed system, computation may not happen in parallel and data is replicated across multiple nodes that users view as a single, coherent system. Variations of both of these models are used to achieve fault tolerance and speed. In the parallel system model, there is still a central authority that has control over all nodes and governs processing. This means that the system is still centralized in nature.

The critical difference between a decentralized system and distributed system is that in a distributed system, there is still a central authority that governs the entire system, whereas in a decentralized system, no such authority exists.

A **decentralized system** is a type of network where nodes are not dependent on a single master node; instead, control is distributed among many nodes. This is analogous to a model where each department in an organization is in charge of its own database server, thus taking away the power from the central server and distributing it to the sub-departments, who manage their own databases.

A significant innovation in the decentralized paradigm that has given rise to this new era of decentralization of applications is **decentralized consensus**. This mechanism came into play with Bitcoin, and it enables a user to agree on something via a consensus algorithm without the need for a central, trusted third party, intermediary, or service provider.

We can also now view the different types of networks shown earlier from a different perspective, where we highlight the controlling authority of these networks as a symbolic hand, as shown in the following diagram. This model provides a clearer understanding of the differences between these networks from a decentralization point of view:

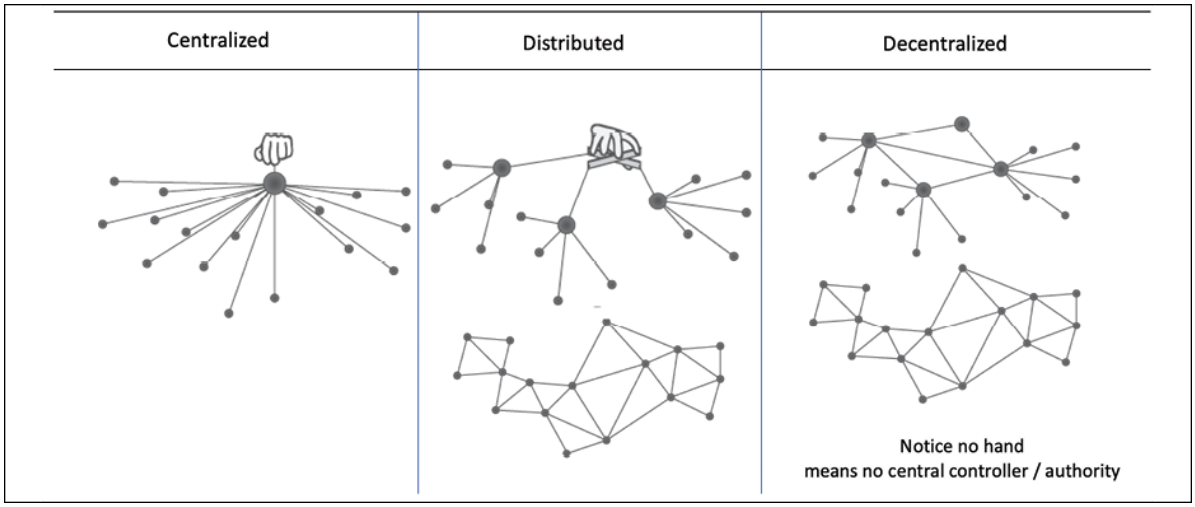

Figure 2.2: Different types of networks/systems depicting decentralization from a modern perspective

The preceding diagram shows that the centralized model is the traditional one in which a central controller exists, and it can be viewed as a depiction of the usual client/server model. In the middle we have distributed systems, where we still have a central controller but the system comprises many dispersed nodes. On the right-hand side, notice that there is no hand/controller controlling the networks.

This is the key difference between decentralized and distributed networks. A decentralized system may look like a distributed system from a topological point of view, but it doesn't have a central authority that controls the network.

The differences between distributed and decentralized systems can also be viewed at a practical level in the following diagrams:

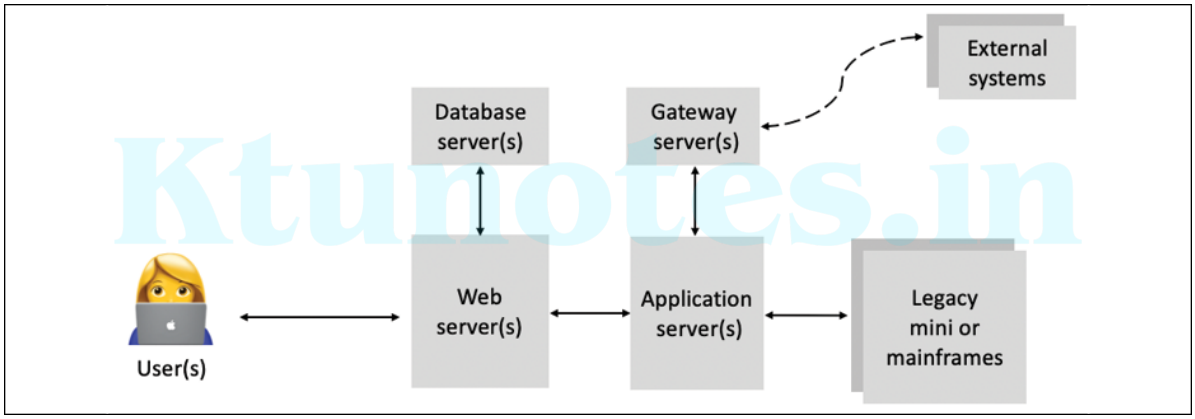

Figure 2.3: A traditional distributed system comprises many servers performing different roles

The following diagram shows a decentralized system (based on blockchain) where an exact replica of the applications and data is maintained across the entire network on each participating node:

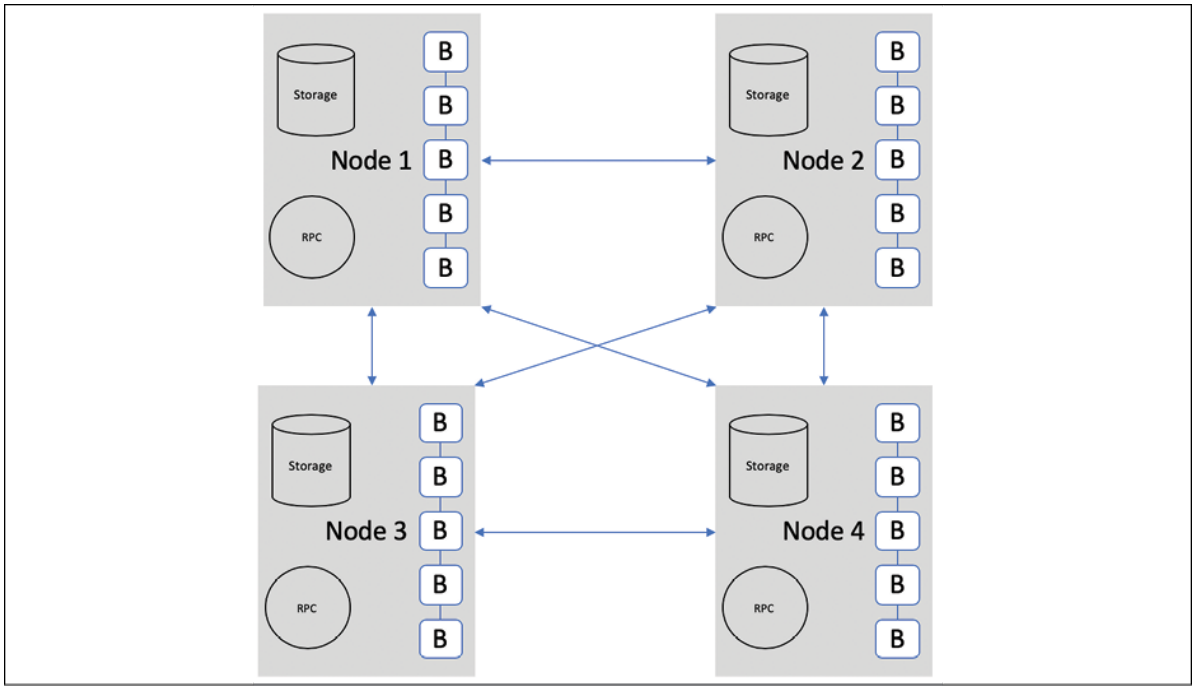

Figure 2.4: A blockchain-based decentralized system (notice the direct P2P connections and the exact replicas of blocks)

A comparison between centralized and decentralized systems (networks/applications) is shown in the following table:

Feature	**Centralized**	**Decentralized**
Ownership	Service provider	All users
Architecture	Client/server	Distributed, different topologies
Security	Basic	More secure
High availability	No	Yes
Fault tolerance	Basic, single point of failure	Highly tolerant, as service is replicated

Collusion resistance	Basic, because it's under the control of a group or even single individual	Highly resistant, as consensus algorithms ensure defense against adversaries
Application architecture	Single application	Application replicated across all nodes on the network
Trust	Consumers have to trust the service provider	No mutual trust required
Cost for consumer	Higher	Lower

The comparison in the table only covers some main features and is not an exhaustive list of all features. There may be other features of interest that can be compared too, but this list should provide a good level of comparison.

Now we will discuss what methods can be used to achieve decentralization.

Methods of decentralization

Two methods can be used to achieve decentralization: disintermediation and competition. These methods will be discussed in detail in the sections that follow.

Disintermediation

The concept of **disintermediation** can be explained with the aid of an example. Imagine that you want to send money to a friend in another country. You go to a bank, which, for a fee, will transfer your money to the bank in that country. In this case, the bank maintains a central database that is updated, confirming that you have sent the money. With blockchain technology, it is possible to send this money directly to your friend without

the need for a bank. All you need is the address of your friend on the blockchain. This way, the intermediary (that is, the bank) is no longer required, and decentralization is achieved by disintermediation. It is debatable, however, how practical decentralization through disintermediation is in the financial sector due to the massive regulatory and compliance requirements. Nevertheless, this model can be used not only in finance but in many other industries as well, such as health, law, and the public sector. In the health industry, where patients, instead of relying on a trusted third party (such as the hospital record system) can be in full control of their own identity and their data that they can share directly with only those entities that they trust. As a general solution, blockchain can serve as a decentralized health record management system where health records can be exchanged securely and directly between different entities (hospitals, pharmaceutical companies, patients) globally without any central authority.

Contest-driven decentralization

In the method involving **competition**, different service providers compete with each other in order to be selected for the provision of services by the system. This paradigm does not achieve complete decentralization. However, to a certain degree, it ensures that an intermediary or service provider is not monopolizing the service. In the context of blockchain technology, a system can be envisioned in which smart contracts can choose an external data provider from a large number of providers based on their reputation, previous score, reviews, and quality of service.

This method will not result in full decentralization, but it allows smart contracts to make a free choice based on the criteria just mentioned. This way, an environment of competition is cultivated among service providers where they compete with each other to become the data provider of choice.

In the following diagram, varying levels of decentralization are shown. On the left side, the conventional approach is shown where a central system is in control; on the right side, complete disintermediation is achieved, as intermediaries are entirely removed. Competing intermediaries or service providers are shown in the center. At that level, intermediaries or service

providers are selected based on reputation or voting, thus achieving partial decentralization:

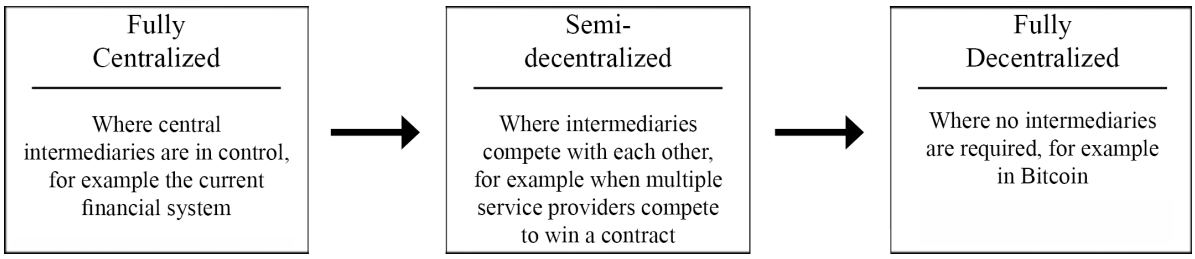

Figure 2.5: Scale of decentralization

There are many benefits of decentralization, including transparency, efficiency, cost saving, development of trusted ecosystems, and in some cases privacy and anonymity. Some challenges, such as security requirements, software bugs, and human error, need to be examined thoroughly.

For example, in a decentralized system such as Bitcoin or Ethereum where security is normally provided by private keys, how can we ensure that an asset or a token associated with these private keys cannot be rendered useless due to negligence or bugs in the code? What if the private keys are lost due to user negligence? What if due to a bug in the smart contract code the decentralized application becomes vulnerable to attack?

Before embarking on a journey to decentralize everything using blockchain and decentralized applications, it is essential that we understand that not everything can or needs to be decentralized.

This view raises some fundamental questions. Is a blockchain really needed? When is a blockchain required? In what circumstances is blockchain preferable to traditional databases? To answer these questions, go through the simple set of questions presented below:

Question	**Yes/No**	**Recommended solution**

Is high data throughput required?	Yes	Use a traditional database.
	No	A central database might still be useful if other requirements are met. For example, if users trust each other, then perhaps there is no need for a blockchain. However, if they don't or trust cannot be established for any reason, blockchain can be helpful.
Are updates centrally controlled?	Yes	Use a traditional database.
	No	You may investigate how a public/private blockchain can help.
Do users trust each other?	Yes	Use a traditional database.
	No	Use a public blockchain.
Are users anonymous?	Yes	Use a public blockchain.
	No	Use a private blockchain.
Is consensus required to be maintained within a consortium?	Yes	Use a private blockchain.
	No	Use a public blockchain.
Is strict data immutability required?	Yes	Use a blockchain.

	No	Use a central/traditional database.

Answering all of these questions can help you decide whether or not a blockchain is required or suitable for solving the problem. Beyond the questions posed in this model, there are many other issues to consider, such as latency, choice of consensus mechanisms, whether consensus is required or not, and where consensus is going to be achieved. If consensus is maintained internally by a consortium, then a private blockchain should be used; otherwise, if consensus is required publicly among multiple entities, then a public blockchain solution should be considered. Other aspects, such as immutability, should also be considered when deciding whether to use a blockchain or a traditional database. If strict data immutability is required, then a public blockchain should be used; otherwise, a central database may be an option.

As blockchain technology matures, there will be more questions raised regarding this selection model. For now, however, this set of questions is sufficient for deciding whether a blockchain-based solution is suitable or not.

Now we understand different methods of decentralization and have looked at how to decide whether a blockchain is required or not in a particular scenario. Let's now look at the process of decentralization, that is, how we can take an existing system and decentralize it. First, we'll briefly look at the different ways to achieve decentralization.

Routes to decentralization

There are systems that pre-date blockchain and Bitcoin, including BitTorrent and the Gnutella file-sharing system, which to a certain degree could be classified as decentralized, but due to a lack of any incentivization mechanism, participation from the community gradually decreased. There wasn't any incentive to keep the users interested in participating in the growth of the network. With the advent of blockchain technology, many

initiatives are being taken to leverage this new technology to achieve decentralization. The Bitcoin blockchain is typically the first choice for many, as it has proven to be the most resilient and secure blockchain and has a market cap of nearly $166 billion at the time of writing. Alternatively, other blockchains, such as Ethereum, serve as the tool of choice for many developers for building decentralized applications. Compared to Bitcoin, Ethereum has become a more prominent choice because of the flexibility it allows for programming any business logic into the blockchain by using **smart contracts**.

How to decentralize

Arvind Narayanan and others have proposed a framework in their book *Bitcoin and Cryptocurrency Technologies* that can be used to evaluate the decentralization requirements of a variety of issues in the context of blockchain technology. The framework raises four questions whose answers provide a clear understanding of how a system can be decentralized:

1. What is being decentralized?
2. What level of decentralization is required?
3. What blockchain is used?
4. What security mechanism is used?

The first question simply asks you to identify what system is being decentralized. This can be any system, such as an identity system or a trading system.

The second question asks you to specify the level of decentralization required by examining the scale of decentralization, as discussed earlier. It can be full disintermediation or partial disintermediation.

The third question asks developers to determine which blockchain is suitable for a particular application. It can be Bitcoin blockchain, Ethereum blockchain, or any other blockchain that is deemed fit for the specific application.

Finally, a fundamental question that needs to be addressed is how the security of a decentralized system will be guaranteed. For example, the security mechanism can be atomicity-based, where either the transaction executes in full or does not execute at all. This deterministic approach ensures the integrity of the system. Other mechanisms may include one based on reputation, which allows for varying degrees of trust in a system.

In the following section, let's evaluate a money transfer system as an example of an application selected to be decentralized.

Decentralization framework example

The four questions discussed previously are used to evaluate the decentralization requirements of this application. The answers to these questions are as follows:

1. Money transfer system
2. Disintermediation
3. Bitcoin
4. Atomicity

The responses indicate that the money transfer system can be decentralized by removing the intermediary, implemented on the Bitcoin blockchain, and that a security guarantee will be provided via atomicity. Atomicity will ensure that transactions execute successfully in full or do not execute at all. We have chosen the Bitcoin blockchain because it is the longest established blockchain and has stood the test of time.

Similarly, this framework can be used for any other system that needs to be evaluated in terms of decentralization. The answers to these four simple questions help clarify what approach to take to decentralize the system.

To achieve complete decentralization, it is necessary that the environment around the blockchain also be decentralized. We'll look at the full ecosystem of decentralization next.

Blockchain and full ecosystem decentralization

The blockchain is a distributed ledger that runs on top of conventional systems. These elements include storage, communication, and computation.

There are other factors, such as identity and wealth, which are traditionally based on centralized paradigms, and there's a need to decentralize these aspects as well in order to achieve a sufficiently decentralized ecosystem.

Storage

Data can be stored directly in a blockchain, and with this fact it achieves decentralization. However, a significant disadvantage of this approach is that a blockchain is not suitable for storing large amounts of data by design. It can store simple transactions and some arbitrary data, but it is certainly not suitable for storing images or large blobs of data, as is the case with traditional database systems.

A better alternative for storing data is to use **distributed hash tables** (**DHTs**). DHTs were used initially in peer-to-peer file sharing software, such as BitTorrent, Napster, Kazaa, and Gnutella. DHT research was made popular by the CAN, Chord, Pastry, and Tapestry projects. BitTorrent is the most scalable and fastest network, but the issue with BitTorrent and the others is that there is no incentive for users to keep the files indefinitely. Users generally don't keep files permanently, and if nodes that have data still required by someone leave the network, there is no way to retrieve it except by having the required nodes rejoin the network so that the files once again become available.

Two primary requirements here are high availability and link stability, which means that data should be available when required and network links also should always be accessible. **Inter-Planetary File System** (**IPFS**) by

Juan Benet possesses both of these properties, and its vision is to provide a decentralized World Wide Web by replacing the HTTP protocol. IPFS uses Kademlia DHT and Merkle **Directed Acyclic Graphs** (**DAGs**) to provide storage and searching functionality, respectively. The concept of DHTs and DAGs will be introduced in detail in *Chapter 4*, *Public Key Cryptography*.

The incentive mechanism for storing data is based on a protocol known as Filecoin, which pays incentives to nodes that store data using the Bitswap mechanism. The Bitswap mechanism lets nodes keep a simple ledger of bytes sent or bytes received in a one-to-one relationship. Also, a Git-based version control mechanism is used in IPFS to provide structure and control over the versioning of data.

There are other alternatives for data storage, such as Ethereum Swarm, Storj, and MaidSafe. Ethereum has its own decentralized and distributed ecosystem that uses Swarm for storage and the Whisper protocol for communication. MaidSafe aims to provide a decentralized World Wide Web. All of these projects are discussed later in this book in greater detail.

BigChainDB is another storage layer decentralization project aimed at providing a scalable, fast, and linearly scalable decentralized database as opposed to a traditional filesystem. BigChainDB complements decentralized processing platforms and filesystems such as Ethereum and IPFS.

Communication

The Internet (the communication layer in blockchain) is considered to be decentralized. This belief is correct to some extent, as the original vision of the Internet was to develop a decentralized communications system. Services such as email and online storage are now all based on a paradigm where the service provider is in control, and users trust such providers to grant them access to the service as requested. This model is based on the unconditional trust of a central authority (the service provider) where users are not in control of their data. Even user passwords are stored on trusted third-party systems.

Thus, there is a need to provide control to individual users in such a way that access to their data is guaranteed and is not dependent on a single third party. Access to the Internet (the communication layer) is based on **Internet Service Providers** (**ISPs**) who act as a central hub for Internet users. If the ISP is shut down for any reason, then no communication is possible with this model.

An alternative is to use **mesh networks**. Even though they are limited in functionality when compared to the Internet, they still provide a decentralized alternative where nodes can talk directly to each other without a central hub such as an ISP.

An example of a mesh network is Firechat, which allows iPhone users to communicate with each other directly in a peer-to-peer fashion without an Internet connection. More information is available at https://www.opengarden.com/firechat/.

Now imagine a network that allows users to be in control of their communication; no one can shut it down for any reason. This could be the next step toward decentralizing communication networks in the blockchain ecosystem. It must be noted that this model may only be vital in a jurisdiction where the Internet is censored and controlled by the government.

As mentioned earlier, the original vision of the Internet was to build a decentralized network; however, over the years, with the advent of large-scale service providers such as Google, Amazon, and eBay, control is shifting toward these big players. For example, email is a decentralized system at its core; that is, anyone can run an email server with minimal effort and can start sending and receiving emails. There are better alternatives available. For example, Gmail and Outlook already provide managed services for end users, so there is a natural inclination toward selecting one of these large centralized services as they are more convenient and free to use. This is one example that shows how the Internet has moved toward centralization.

Free services, however, are offered at the cost of exposing valuable personal data, and many users are unaware of this fact. Blockchain has revived the vision of decentralization across the world, and now concerted efforts are being made to harness this technology and take advantage of the benefits that it can provide.

Computing power and decentralization

Decentralization of computing or processing power is achieved by a blockchain technology such as Ethereum, where smart contracts with embedded business logic can run on the blockchain network. Other blockchain technologies also provide similar processing-layer platforms, where business logic can run over the network in a decentralized manner.

The following diagram shows an overview of a decentralized ecosystem. In the bottom layer, the Internet or mesh networks provide a decentralized communication layer. In the next layer up, a storage layer uses technologies such as IPFS and BigChainDB to enable decentralization. Finally, in the next level up, you can see that the blockchain serves as a decentralized processing (computation) layer. Blockchain can, in a limited way, provide a storage layer too, but that severely hampers the speed and capacity of the system. Therefore, other solutions such as IPFS and BigChainDB are more suitable for storing large amounts of data in a decentralized way. The Identity and Wealth layers are shown at the top level. Identity on the Internet is a vast topic, and systems such as bitAuth and OpenID provide authentication and identification services with varying degrees of decentralization and security assumptions:

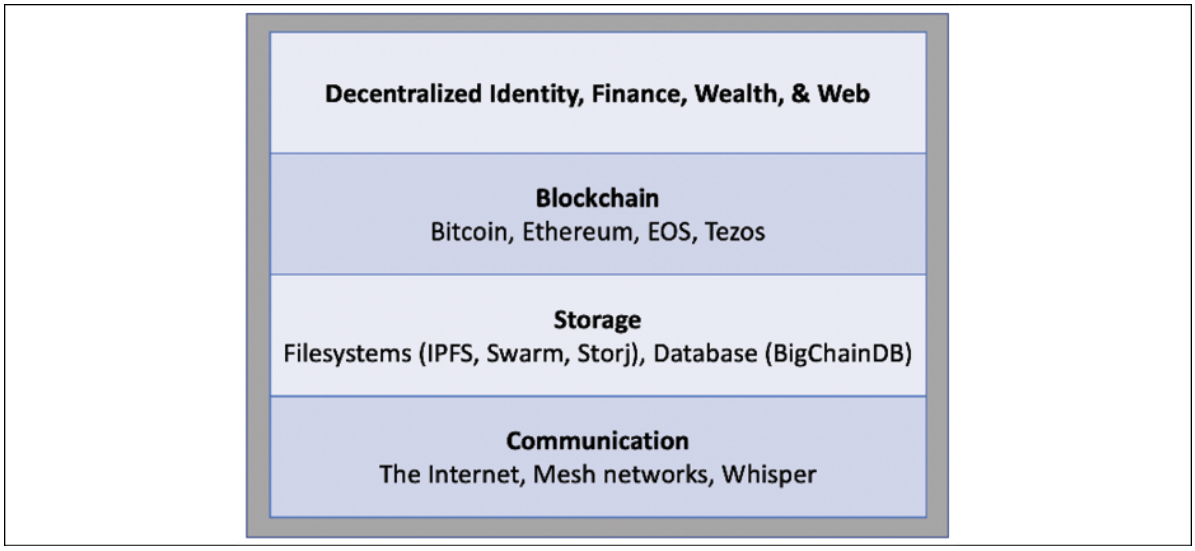

Figure 2.6: Decentralized ecosystem

The blockchain is capable of providing solutions to various issues relating to decentralization. A concept relevant to identity known as **Zooko's Triangle** requires that the naming system in a network protocol is secure, decentralized, and able to provide human-meaningful and memorable names to the users. Conjecture has it that a system can have only two of these properties simultaneously.

Nevertheless, with the advent of blockchain in the form of **Namecoin**, this problem was resolved. It is now possible to achieve security, decentralization, and human-meaningful names with the Namecoin blockchain. However, this is not a panacea, and it comes with many challenges, such as reliance on users to store and maintain private keys securely. This opens up other general questions about the suitability of decentralization to a particular problem.

Decentralization may not be appropriate for every scenario. Centralized systems with well-established reputations tend to work better in many cases. For example, email platforms from reputable companies such as Google or Microsoft would provide a better service than a scenario where individual email servers are hosted by users on the Internet.

There are many projects underway that are developing solutions for a more comprehensive distributed blockchain system. For example, Swarm and