Section 5

The Internet of Money

Source: The Economist (Cover)

Blockchain Technology

Everything will be tokenized and connected by a blockchain one day.

Origin

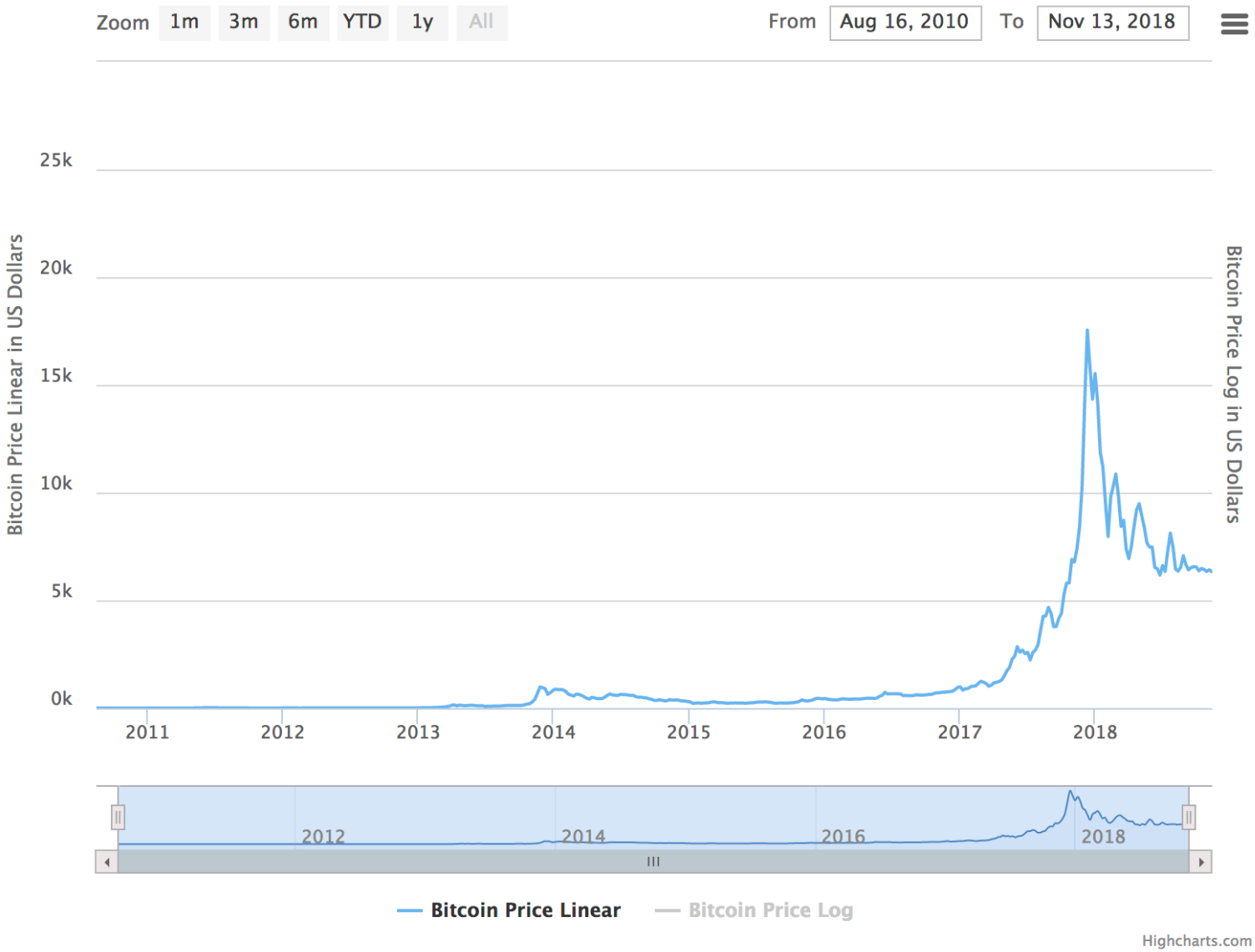
The core ideas behind blockchain technology emerged in 1991. A signed chain of information was used as an *electronic ledger* for **digitally signing documents** in a way that could easily show none of the signed documents in the collection had been changed. In 2008, the first application to digital cash was described in the original paper on Bitcoin, *Bitcoin: A Peer to Peer Electronic Cash System*[1].

The original paper on Bitcoin was published under the name *Satoshi Nakamoto*—a pseudonym. The actual author(s) and owner of the first Bitcoins remain a mystery. Nakamoto's paper provided the blueprint that most modern digital cash schemes follow, with many variations.

- Bitcoin is implemented in a distributed fashion so that no single user controls the currency and no single point of failure exists.
- By using a distributed blockchain and consensus-based maintenance, a self-regulating mechanism was created that ensures that only valid transactions are added to the blockchain.
- Blockchain technology has become tightly linked to Bitcoin and and other crypto currencies, although it is **not restricted to simple fund transfers**.

[1] Satoshi Nakamoto, satoshin@gmx.com, www.bitcoin.org, https://bitcoin.org/bitcoin.pdf.

Bitcoin Price History Chart



Current Price: $5,688.43 **Today:** -9.81% -$618.81

Since 2015, hundreds of cryptocurrency specifications exist; most are similar to and derived from the first **fully implemented decentralized cryptocurrency**, Bitcoin.
Examples: Ethereum and Ripple.

- Decentralized cryptocurrency is produced by the entire cryptocurrency system collectively, at a rate which is defined when the system is created and which is publicly known.
- The security of cryptocurrency ledgers is based on the assumption that the majority of *miners* are honestly trying to maintain the ledger, having **financial incentive to do so**.
- Bitcoin's decentralized control verifies and records transactions using a blockchain transaction database in the role of a public distributed ledger.[2] The system is peer-to-peer, and transactions take place between users directly, without an intermediary.
- Blockchains are **immutable digital ledger systems** implemented in a distributed fashion, without a central repository and normally also without a central authority.

Decentralized cryptocurrencies such as bitcoin (BTC or XBT) now provide an outlet for personal wealth that is beyond restriction and confiscation.

[2] A ledger is the principal book or computer file for recording and totalling economic transactions measured in terms of a monetary unit of account by account type, with debits and credits in separate columns and a beginning monetary balance and ending monetary balance for each account.

Reference

Blockchain Technology Overview, Draft NISTIR 8202, National Institute of Standards and Technology, U.S. Department of Commerce, January, 2018

Blockchain Architecture

Blockchains utilize computer science mechanisms such as linked lists and distributed networking as well as cryptographic primitives like hashing, digital signatures, and public/private keys.

Hashes

An important component of blockchain technology is the use of cryptographic hash functions for many operations, such as hashing the content of a block. Even the smallest change of input (e.g., a single bit) will result in a completely different ***output digest***.

With the hashing standards used for blockchains it is computationally infeasible to

- find any input that maps to any pre-specified output
- find two or more inputs that produce the same output.

A hashing algorithm used in many blockchain technologies is the **Secure Hash Algorithm (SHA)** with an output size of 256 bits (SHA-256). This algorithm has an output of 32 (8-bit) characters, meaning that there are $2^{256} \approx 10^{77}$, or

> 115,792,089,237,316,195,423,570,985,008,687,907,853,269,984,665,640,564,039,457,584,007,913,129,639,936 **possible digest values**.

SHA-256 Digest Value(1) = `0x6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b`

SHA-256 Digest Value(2) = `0xd4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35`

Examples of inputs mapped to SHA-256 digest values shown as a 64-character hexadecimal string

Blockchain Architecture

Blockchains utilize computer science mechanisms such as linked lists and distributed networking as well as cryptographic primitives like hashing, digital signatures, and public/private keys.

Hashes

An important component of blockchain technology is the use of cryptographic hash functions for many operations, such as hashing the content of a block. Even the smallest change of input (e.g., a single bit) will result in a completely different ***output digest***.

With the hashing standards used for blockchains it is computationally infeasible to

- find any input that maps to any pre-specified output
- find two or more inputs that produce the same output.

A hashing algorithm used in many blockchain technologies is the **Secure Hash Algorithm (SHA)** with an output size of 256 bits (SHA-256). This algorithm has an output of 32 (8-bit) characters, meaning that there are $2^{256} \approx 10^{77}$, or

> 115,792,089,237,316,195,423,570,985,008,687,907,853,269,984,665,640,564,039,457,584,007,913,129,639,936 possible digest values.

SHA-256 Digest Value(**1**) = `0x6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b`

SHA-256 Digest Value(**2**) = `0xd4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35`

SHA-256..(**Hello, World!**) = `0xdffd6021bb2bd5b0af676290809ec3a53191dd81c7f70a4b28688a362182986f`

Blockchain technologies take a list of transactions and create a ***hash "fingerprint"*** (the digest) for the list. If a single value in a transaction within the list changes, the digest for that block changes, making it easy to discover even minor one bit changes.

What potential problem do you see here?

Blockchain technologies take a list of transactions and create a ***hash "fingerprint"*** (the digest) for the list. If a single value in a transaction within the list changes, the digest for that block changes, making it easy to discover even minor one bit changes.

Collision Resistance

Given the finite number of possible output digest values and an extremely large number of possible input values and, it is possible to have **a collision** where hash(x) = hash(y). However, it is highly unlikely for any such input x and y that produce the same digest to both be valid in the context of the blockchain system (in this case, both being valid blockchain transactions) as well as be computed reasonably close to each other in time.

The hashing algorithm used (SHA-256) is said to be ***collision resistant***, since to find a collision in SHA-256, one would have to execute the algorithm, on average, about 2^{128} times.

Secure Hash Algorithms

A cryptographic hash is like a ***signature*** for a text or a data file—a one way function that cannot be decrypted back—suitable for password validation, anti-tamper, digital signatures etc.

The SHA family of cryptographic hash functions is published by the **National Institute of Standards and Technology**, or NIST, as a U.S. Federal Information Processing Standard.

All SHA-family algorithms, as FIPS-approved security functions, are subject to official validation by the CMVP (Cryptographic Module Validation Program), a joint program run by the American NIST and the Canadian CSE (Communications Security Establishment).

Comparison of SHA functions view · talk · edit

Algorithm and variant		Output size (bits)	Internal state size (bits)	Block size (bits)	Rounds	Operations	Security (in bits) against collision attacks	Capacity against length extension attacks	Performance on Skylake (median cpb)[1]		First published
									long messages	8 bytes	
MD5 (as reference)		128	128 (4 × 32)	512	64	And, Xor, Rot, Add (mod 2^{32}), Or	≤18 (collisions found)[2]	0	4.99	55.00	1992
SHA-0		160	160 (5 × 32)	512	80	And, Xor, Rot, Add (mod 2^{32}), Or	<34 (collisions found)	0	≈ SHA-1	≈ SHA-1	1993
SHA-1							<63 (collisions found[3])		3.47	52.00	1995
SHA-2	*SHA-224*	224	256 (8 × 32)	512	64	And, Xor, Rot, Add (mod 2^{32}), Or, Shr	112	32	7.62	84.50	2004
	SHA-256	256					128	0	7.63	85.25	2001
	SHA-384	384	512 (8 × 64)	1024	80	And, Xor, Rot, Add (mod 2^{64}), Or, Shr	192	128 (≤ 384)	5.12	135.75	2001
	SHA-512	512					256	0	5.06	135.50	
	SHA-512/224	224					112	288	≈ SHA-384	≈ SHA-384	2012
	SHA-512/256	256					128	256			
SHA-3	*SHA3-224*	224	1600 (5 × 5 × 64)	1152	24[4]	And, Xor, Rot, Not	112	448	8.12	154.25	2015
	SHA3-256	256		1088			128	512	8.59	155.50	
	SHA3-384	384		832			192	768	11.06	164.00	
	SHA3-512	512		576			256	1024	15.88	164.00	
	SHAKE128	*d* (arbitrary)		1344			min(*d*/2, 128)	256	7.08	155.25	
	SHAKE256	*d* (arbitrary)		1088			min(*d*/2, 256)	512	8.59	155.50	

Transactions

A transaction is a recording of a transfer of assets (digital currency, units of inventory, etc.) between parties. An analog to this would be a record in a checking account for each time money was deposited or withdrawn.

	Input	Output	Amount	Total
Transaction ID: 0xa1b2c3	Account A	Account B	0.0321	
		Account C	2.5000	
				2.5321

A single transaction typically requires at least the following information fields:

- *Amount* – The total amount of the digital asset to transfer.
- *Inputs* – A list of the digital assets to be transferred (their total value equals the amount).
- *Outputs* – The accounts that will be the recipients of the digital assets.
- *Transaction ID/Hash* – A unique identifier for each transaction. Some blockchains use an ID, and others take a hash of the specific transaction as a unique identifier.

Determining the validity of a transaction: Just because someone claims a transaction took place does not mean it really happened. Transactions are signed and can be verified with public/private key pairs at any time.

Asymmetric-Key Cryptography

A fundamental technology utilized by blockchain technologies is referred to as *public/private key cryptography*[3] using a **pair of keys**: a public key and a private key that are mathematically related to each other

- The public key may be made public without reducing the security of the process, but the private key must remain secret if the data is to **retain its cryptographic protection**.
- Despite the relationship between the two keys, the private key cannot efficiently be determined based on knowledge of the public key.

Asymmetric-key cryptography utilization in blockchain systems:

- Private keys are used to **digitally sign transactions**.
- Public keys are used to
  - **derive addresses**, allowing for a one-to-many approach for pseudonymity (one public key pair can yield multiple addresses),
  - **verify signatures** generated with private keys.
- Verification that the user transferring value to another user is in possession of the private key capable of signing the value.

[3] Public-key cryptography, or asymmetric cryptography, is any cryptographic system that uses pairs of keys: public keys which may be disseminated widely, and private keys which are known only to the owner.

The Public and Private key pair comprise of two uniquely related cryptographic keys (basically long random numbers). Example of a Public Key:

3048 0241 00C9 18FA CF8D EB2D EFD5 FD37 89B9 E069 EA97 FC20 5E35 F577 EE31 C4FB C6E4 4811 7D86 BC8F BAFA 362F 922B F01B 2F40 C744 2654 C0DD 2881 D673 CA2B 4003 C266 E2CD CB02 0301 0001

- The Public Key is made available to everyone via a publicly accessible repository or directory, whereas the Private Key must remain confidential to its respective owner.
- Because the key pair is mathematically related, whatever is encrypted with a Public Key may only be decrypted by its corresponding Private Key and vice versa.

  *E.g., if **Bob** wants to send sensitive data to **Alice**, and wants to be sure that only Alice may be able to read it, he will encrypt the data with **Alice's Public Key**. Only Alice has access to her corresponding Private Key and as a result is the only person with the capability of decrypting the encrypted data back into its original form.*

  Even if someone else gains access to the encrypted data, it will remain confidential as they should not have access to Alice's Private Key.

- Public key cryptography can achieve confidentiality. Another important aspect of this technology is its ability to create a Digital Signature.

Addresses and Address Derivation

A user's address is a short, alphanumeric string derived from the user's public key using a hash function (along with some additional data used to detect errors). Addresses are used to send and receive digital assets.

- Most blockchain systems make use of addresses as the "to" and "from" endpoints in a transaction.
- Addresses are shorter than the public keys and are not secret.
- An address is typically generated by taking a public key, hashing it, and converting the hash to text: *public key* $\Rightarrow$ *hash function* $\Rightarrow$ *address*

Users can generate as many private/public key pairs, and therefore addresses as desired, allowing for a varying degree of pseudo-anonymity. Addresses act as the public-facing "identity" on a blockchain for a user, and often will be converted into a ***QR code*** for easier use.

A blockchain distributes digital assets by assigning them to an address. To spend a digital asset, the user must prove possession of the address's corresponding private key. By digitally signing a transaction with the private key, the transaction can be verified with the public key.

Private Key Storage

Most users of a blockchain system do not record their private keys manually, rather, software commonly called a wallet securely stores them. The wallet can store private keys, public keys, and associated addresses. The wallet software can also calculate the total number of assets a user may have.

A private key is usually generated using a secure random function, meaning that reconstructing it is difficult, if not impossible. If a user loses a private key, then any asset associated with that key is lost. If a private key is stolen, the attacker will have full access to all assets controlled by that private key. The **security of private keys** is so important that many users use special secure hardware to store it.

Private key storage is an extremely important aspect of blockchain technology. When it is reported in the news that "Bitcoin was stolen from..", it almost certainly means the private keys were found and used to sign a transaction sending the money to a new account, not that the system was compromised. Note that because blockchain data cannot generally be changed, once a criminal steals a private key and publicly moves the associated funds to another account, **it cannot be undone**.

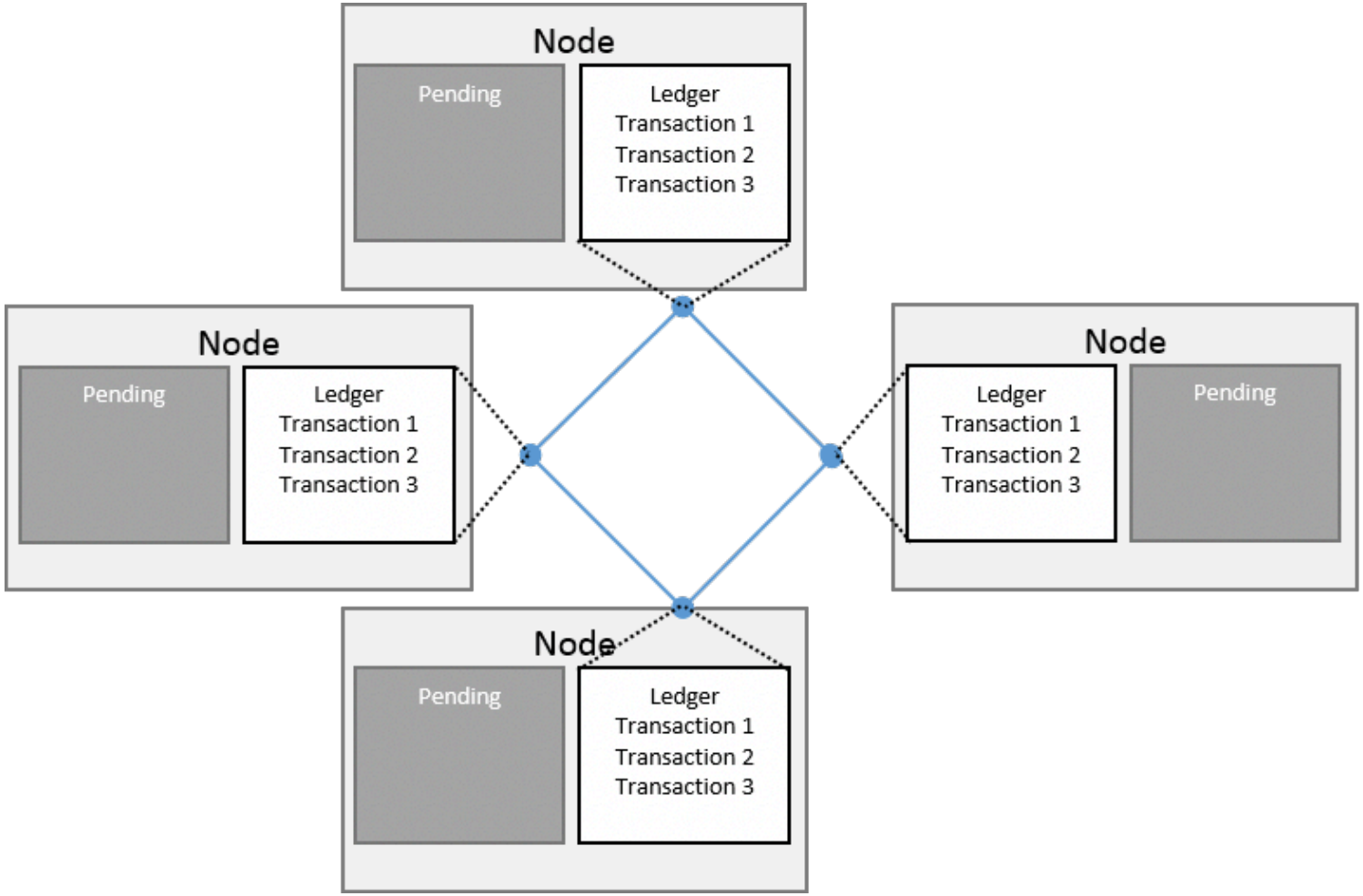
Ledgers

A ledger is a collection of transactions. Historically, paper ledgers have been used to keep track of the exchange of goods and services. More recently, ledgers have been stored digitally, often in large databases owned and operated solely by centralized "trusted" third parties on behalf of a community of users (i.e., the third party is the owner of the ledger).

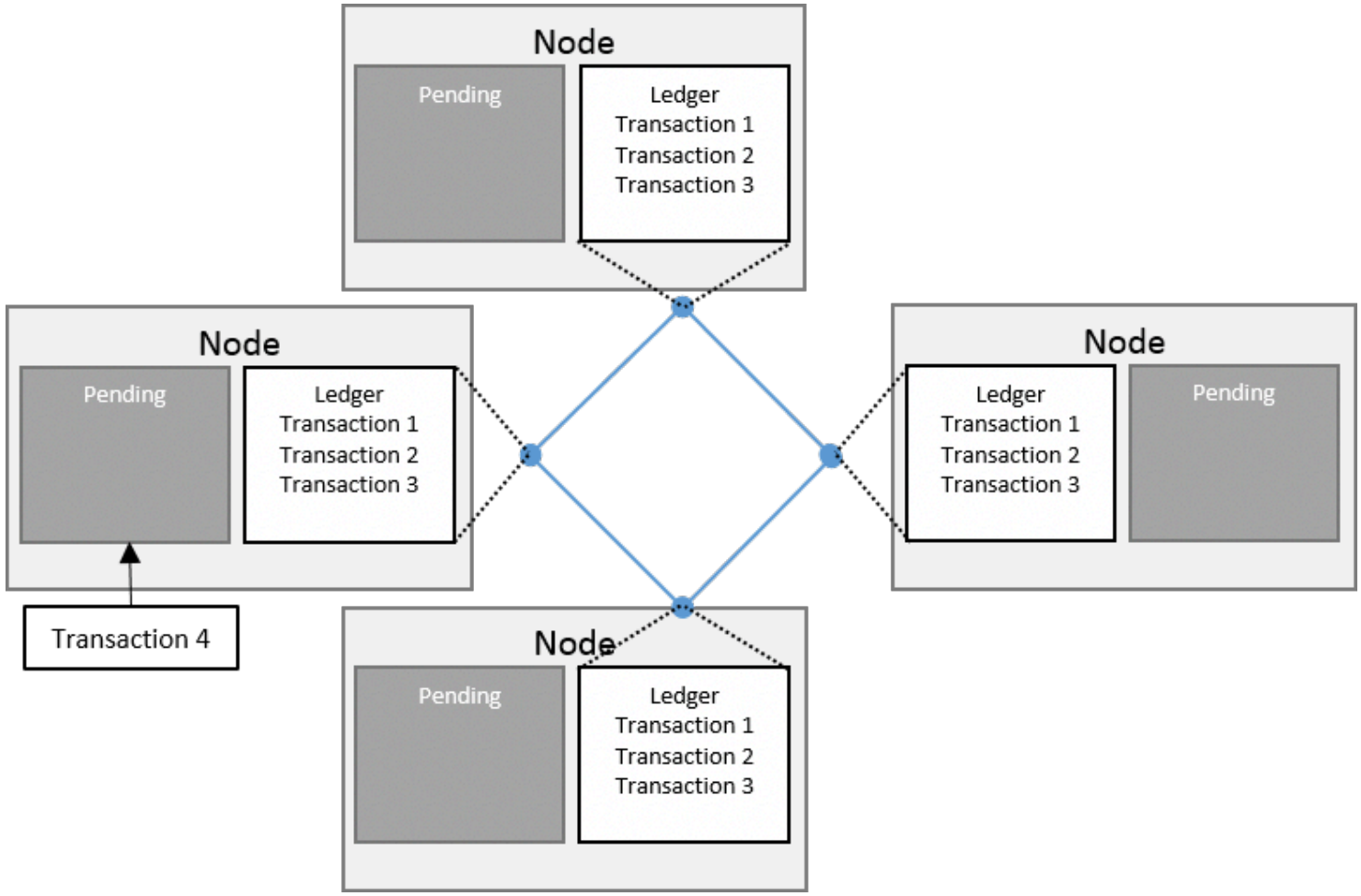
Centralized ledgers may have shortcomings, such as:

- They may be **lost or destroyed**; a user must trust that the owner is properly backing up the system.
- Transactions may **not be valid**; a user must trust that the owner is validating each received transaction.
- A transaction list may **not be complete**; a user must trust that the owner is including all valid transactions that have been received.
- Transaction data may **have been altered**; a user must trust that the owner is not altering past transactions.

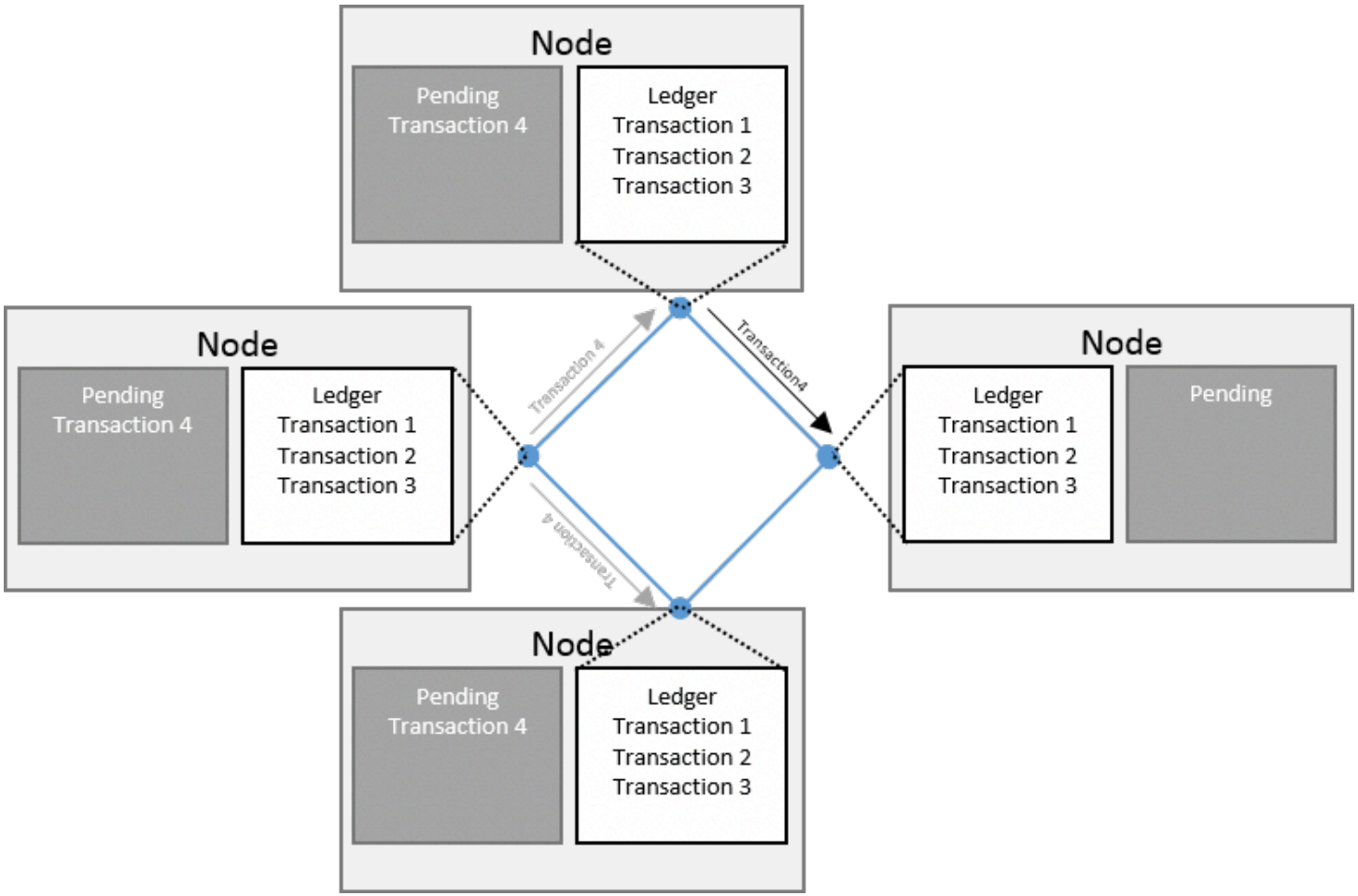
A **ledger implemented using a blockchain** can mitigate these issues through the use of a distributed consensus mechanism. The blockchain ledger will be copied and distributed amongst every node within the system.



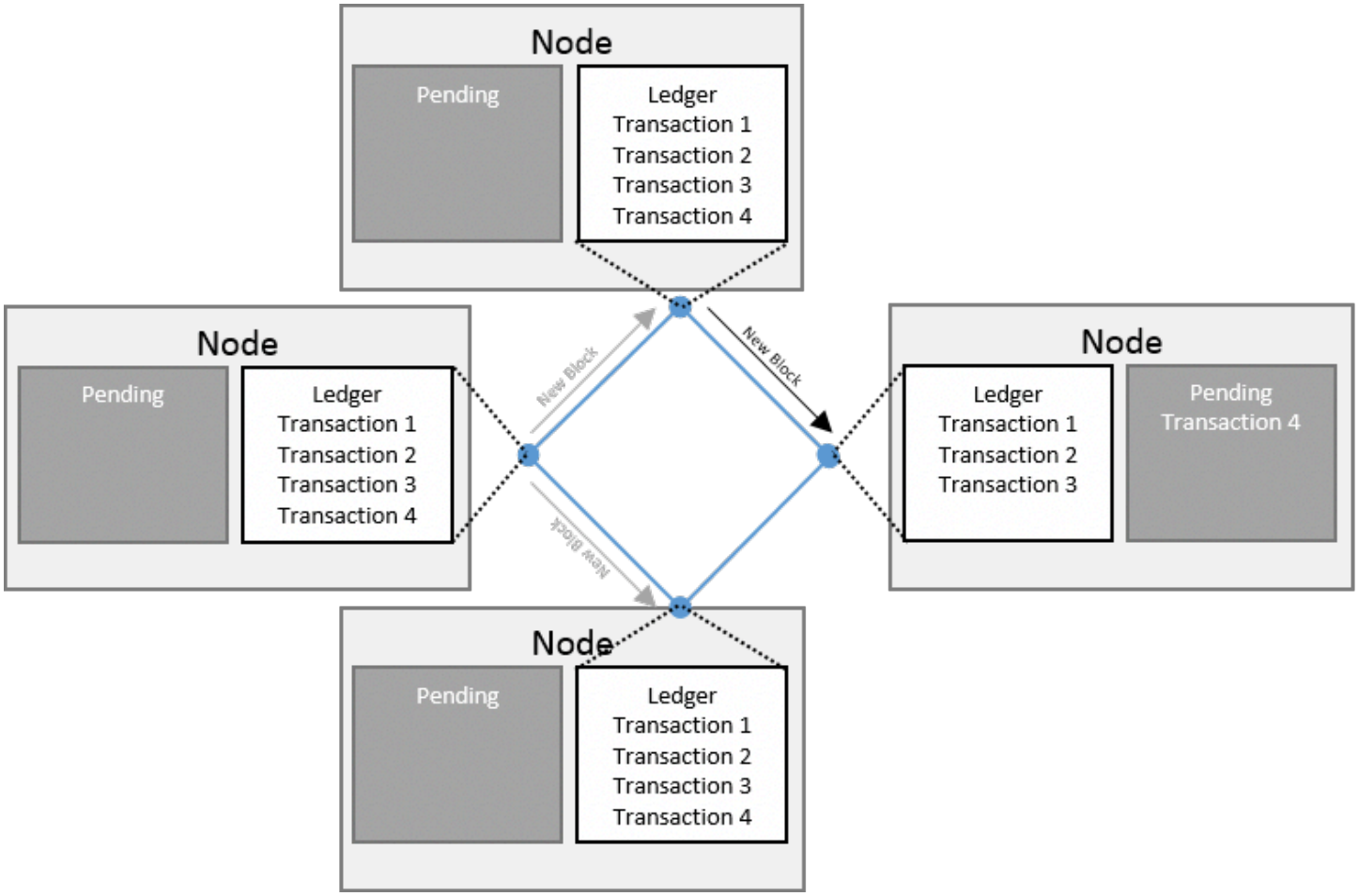
A simple network maintaining a copy of a ledger across nodes: new transactions are submitted to a node, which will then alert the rest of the network that a new transaction has arrived.



Submitting a transaction to a node, waiting in the Pending Transaction List: at this point, it is a pending transaction, not included in a block within the ledger.



Transaction 4 information transmitted from node to node: eventually, a node will include this new transaction within a block and complete the system's required consensus method. This new block will be distributed across the system and all ledgers will be updated to include the new transaction.



Transaction 4 has been included into a block, nodes are transmitting the information: the final node has not yet received the latest information.

Blocks

Users submit candidate transactions to the ledger by sending these transactions to one or more nodes participating in the blockchain. Submitted transactions are propagated to the other nodes in the network (but this by itself does not include the transaction in the blockchain). The distributed transactions then wait in a queue, or **transaction pool**, until they are added to the blockchain by a mining node.

Mining nodes are the subset of nodes that maintain the blockchain by publishing new blocks. Transaction are added to the blockchain when a mining node publishes a block. A block contains a set of validated transactions.

'Validity' is ensured by checking that the providers of funds in each transaction (listed in the transaction's 'input' values) have each cryptographically signed the transaction. This verifies that the providers of funds for a transaction had access to the key which could sign over the available funds. The other mining nodes will check the validity of the transactions.

Blocks

Users submit candidate transactions to the ledger by sending these transactions to one or more nodes participating in the blockchain. Submitted transactions are propagated to the other nodes in the network (but this by itself does not include the transaction in the blockchain). The distributed transactions then wait in a queue, or **transaction pool**, until they are added to the blockchain by a mining node.

Mining nodes are the subset of nodes that maintain the blockchain by publishing new blocks. Transaction are added to the blockchain when a mining node publishes a block. A block contains a set of validated transactions.

'Validity' is ensured by checking that the providers of funds in each transaction (listed in the transaction's 'input' values) have each cryptographically signed the transaction. This verifies that the providers of funds for a transaction had access to the key which could sign over the available funds. The other mining nodes will check the validity of the transactions.

After creation, each block is hashed thereby creating a digest that represents the block. The block's hash digest is used to help protect the block from change since all nodes will have a copy of the block's hash and can then check to make sure that the block has not been changed.

The actual **construction of a block** is slightly more complicated. The data fields comprising a block typically consist of the following:

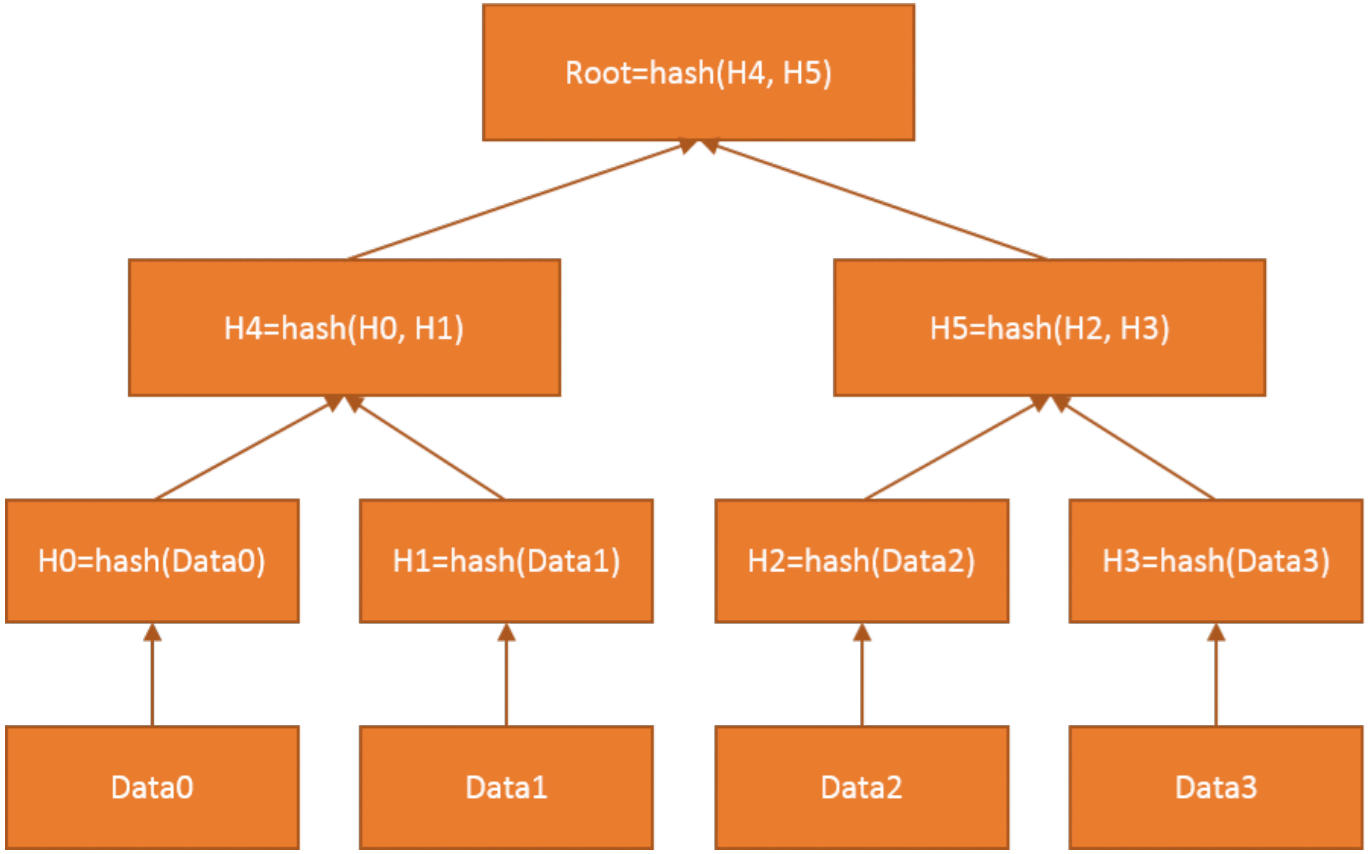
- The block number, also known as block height
- The current block hash value
- The previous block hash value
- The *Merkle tree root hash* (defined below)
- A timestamp
- The size of the block
- The *nonce value*, which is a number manipulated by the mining node to solve the hash puzzle that gives them the right to publish the block
- A list of transactions included within the block

Rather than storing the hash of every transaction within the header of a block, a data structure known as a Merkle tree is utilized. A Merkle tree combines the hash values of data together until there is a singular root (a Merkle tree root hash).

The Merkle tree root hash is an efficient mechanism used to summarize the transactions in a block and verify the presence of a transaction within a block. This structure ensures that the data sent in a distributed network is valid, since any alteration to the underlying data would be detected and can be discarded.

Example of a Merkle tree:

- The bottom row represents the **data to be summarized**, in the case of blockchains this is the transaction data.
- The second to bottom row shows that **data being hashed**.
- The hashed data from the second row is then combined and then hashed on the third to bottom row.
- Finally, the top row shows the **Root hash**, which combines and hashes H4 and H5 (shown below). The root hash is a hash of all previous combinations and hashes made.

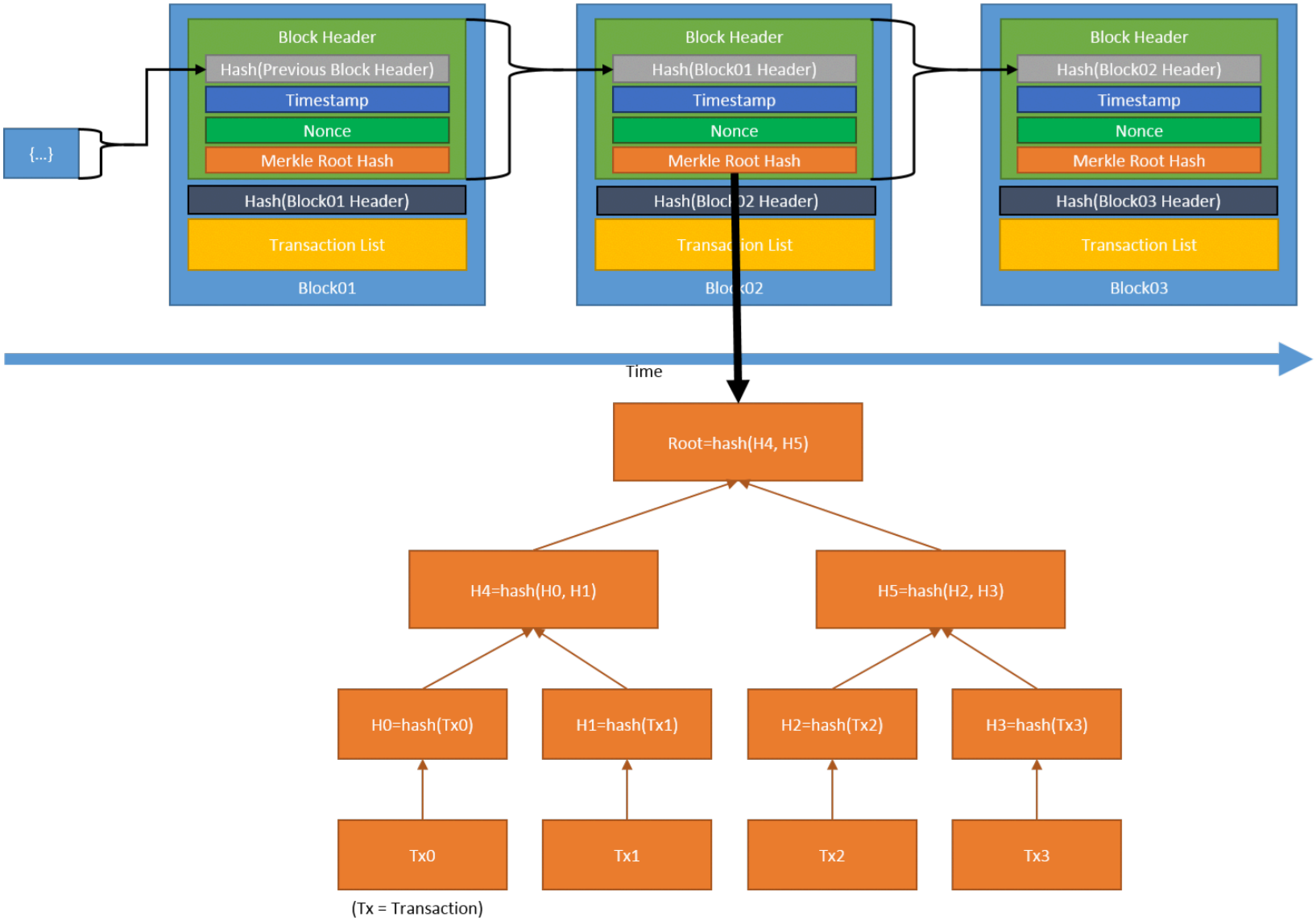


Example of a Merkle Tree

The following figure (next slide) illustrates the relationship between a Merkle tree and a block.

The bottom row of the tree contains blockchain transactions Tx0 through Tx3. The Merkle root is stored within the block header.

The entire block header is hashed; the block header hash value is stored within the block itself, as well as within in the next block, and this helps provide the immutability of transactions since the Merkle root hash will not match if any change is made to the transactions.



Blockchain with Merkle Tree

Chaining Blocks

Each block contains the hash of the previous block's header, thus forming the blockchain. Any change of a previously published block would result in a different hash for this block, which in turn would cause all subsequent blocks to also have different hashes since they include the hash of the respective previous block.

This makes it possible to **easily detect and reject** any changes to previously published blocks.

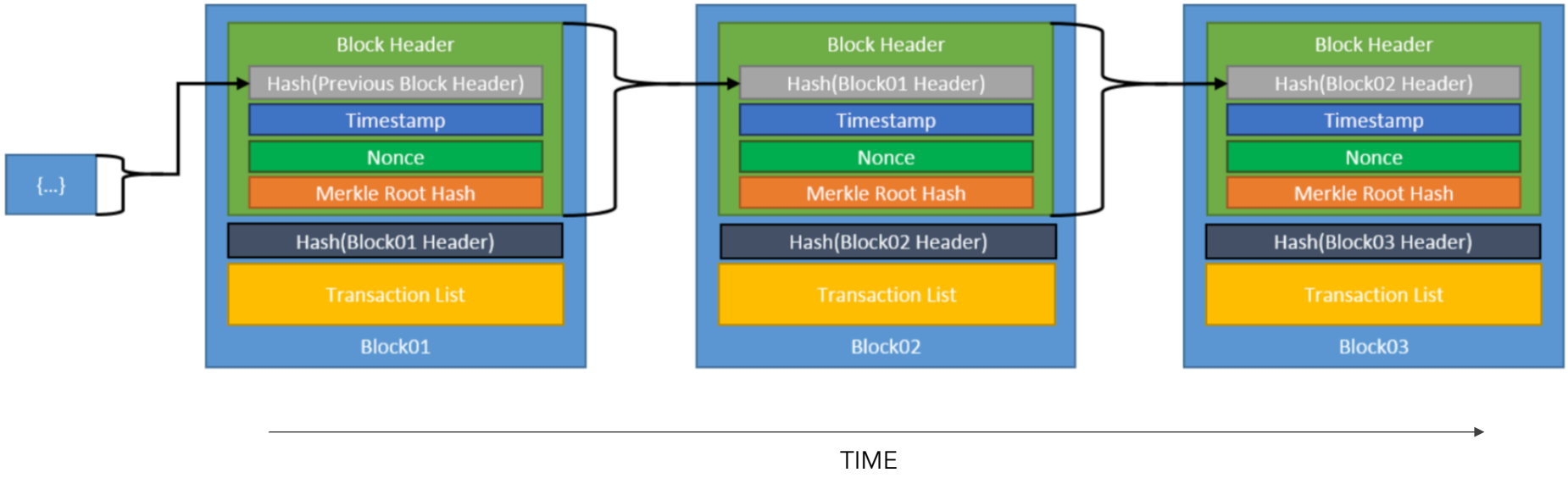


Figure of a generic chain of blocks

Blockchain Operation

Building on the static view of the components of a generic blockchain, we discuss how a blockchain is expanded through the addition of new blocks representing sets of transactions.

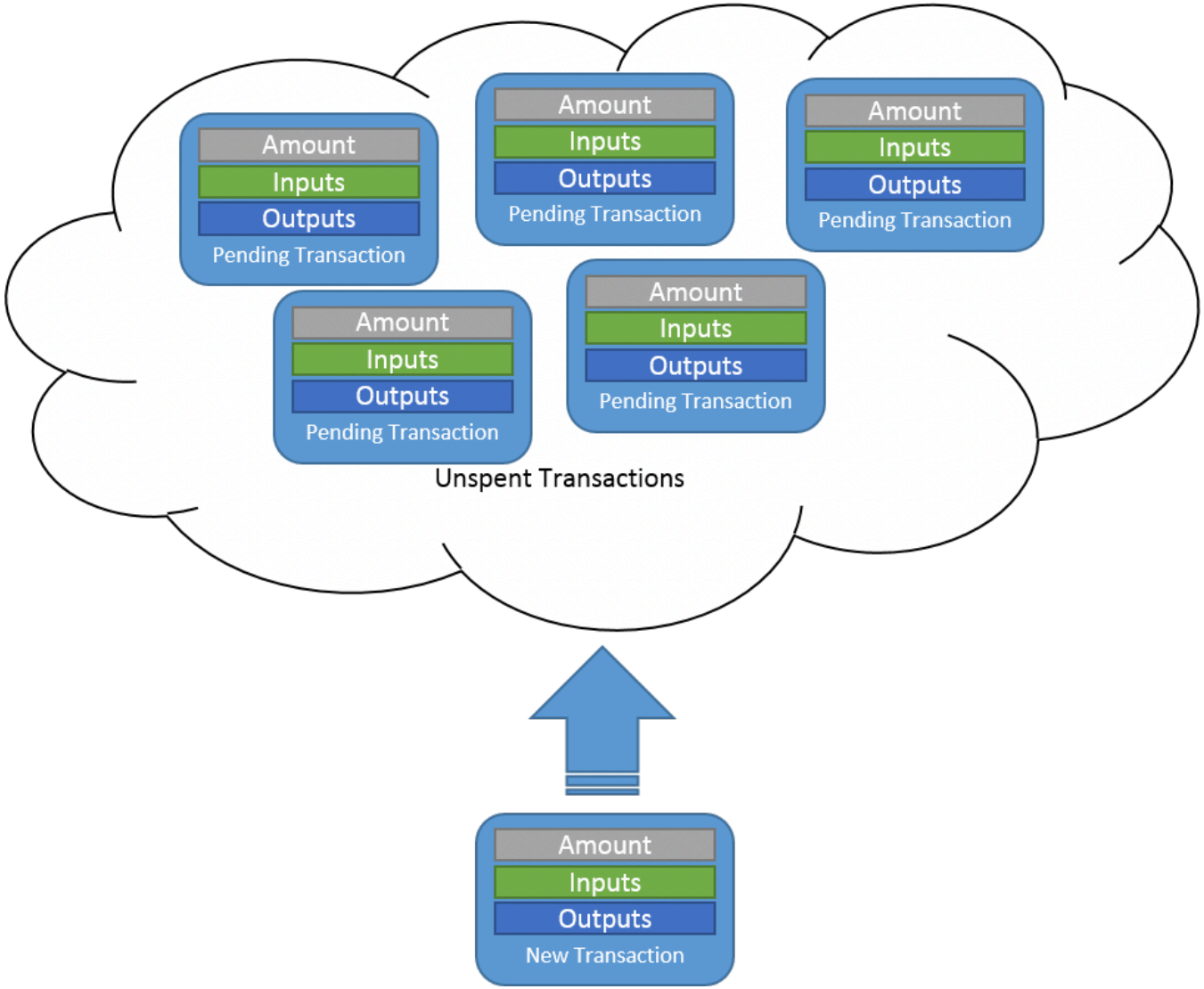
We focus here on a permissionless blockchain that utilizes the **proof of work consensus method**. This is the most popular method to date and the one used by Bitcoin and its derivatives.

- Blockchains are maintained through the consensus of a set of computer systems running blockchain software, known as mining nodes.
- With no central authority determining which node publishes the next block on the blockchain, each node maintains a copy of the blockchain and may propose a new block to the other mining nodes.
- Invalid blocks will be detected and rejected because it is difficult to compute a valid block, but computationally easy to verify one.
- Mining is an intentionally resource-intensive task, taking large amounts of processing power, memory, or both, depending on the specific blockchain application.

Any computer running blockchain software is considered a node of that blockchain.

- There are generally two types of nodes: full nodes and lightweight nodes. The job of a full node is to store the blockchain data, pass along the data to other nodes, and ensure newly added blocks are valid.
- Validation entails ensuring that the format of the block is correct, all hashes in the new block were computed correctly, the new block contains the hash of the previous block, and each transaction in the block is valid and signed by the appropriate parties.
- Full nodes may also act as mining nodes (i.e., generating new blocks).
- Lightweight nodes do not need to store full copies of the blockchain and often pass their data on to full nodes to be processed. Lightweight nodes are generally found on smartphones and Internet of Things (IoT) devices—devices with limited computational and/or storage capability.
- Any node may **propose new transactions**, and these proposed transactions are propagated between nodes until they are eventually added to a block.

Proposed transactions within a blockchain system are stored on mining nodes within an unspent transaction pool—waiting to be included within a block as shown in the next figure.



Transaction Being Added to Unspent Transaction Pool