|  |
| --- |
| Bitcoin combines the idea of using computational puzzles to regulate the creation of new currency units with the idea of secure timestamping to record a ledger of transactions and prevent double spending. |
| In Bitcoin, puzzle solutions themselves don’t constitute money. They are used to secure the block chain, and only indirectly lead to minting money for a limited time |
| Bitcoin, as we’ve seen, doesn’t require trusted timestamping, and merely tries to preserve the relative order of blocks and transactions |
| In Bitcoin, by contrast, for an attacker to change history, they must solve computational puzzles at a faster rate than the rest of the participants combined |
|  |
| Chapter 1: Introduction to Cryptography & Cryptocurrencies |
| All currencies need some way to control supply and enforce various security properties to prevent cheating. |
| cryptographically secure hash function:  (1) collision‐resistance (nobody can find a collision) (message digest)  (2) hiding  (3) puzzle‐friendliness |
| we can find a collision by only examining roughly the square root of the number of possible outputs results |
| Functions for which people have tried really, really hard to find collisions and haven’t yet succeeded (MD5 is deprecated) |
|  |
| Every time you commit to a value, it is important that you choose a new random value nonce |
| And it turns out that the binding property is implied by the collision‐resistant property of the underlying hash function. (But binding property do not imply collision-resistant) |
|  |
| If a search puzzle is puzzle‐friendly, this implies that there’s no solving strategy for this puzzle which is much better than just trying random values of x . |
| Merkle‐Damgard transform: is divided into blocks of length m‐n . The construction works as follows: pass each block together with the output of the previous block into the compression function |
| SHA‐256 uses a compression function that takes 768‐bit input and produces 256‐bit outputs |
|  |
| 1.2 Hash Pointers and Data Structures |
| Whereas a regular pointer gives you a way to retrieve the information, a hash pointer also gives you a way to verify that the information hasn’t changed |
| A use case for a block chain is a tamper‐evident log. |
| binary tree with hash pointers is known as a Merkle tree    we remember just the hash pointer at the head of the tree |
| Proof of membership: they need to show us this data block, and the blocks on the path from the data block to the root. |
| sorted Merkle tree |
|  |
| 1.3 Digital Signatures |
|  |
| sign the hash of the message  sign a hash pointer |
| ECDSA: |
| There is no encryption in Bitcoin, because nothing needs to be encrypted |
|  |
| GoofyCoin  The first rule is that a designated entity, Goofy, can create new coins  The string, together with Goofy’s signature, is a coin  The second rule of GoofyCoin is that whoever owns a coin can transfer it on to someone else    double‐spending attack |
| ScroogeCoin  The first key idea is that a designated entity called Scrooge publishes an append‐only ledger containing the history of all the transactions that have happened  defend against double‐spending by requiring all transactions to be written the ledger before they are accepted    The second kind of transaction is PayCoins. It consumes some coins, that is, destroys them, and creates new coins of the same total value  The problem here is centralization. |
|  |
| Chapter 2: How Bitcoin Achieves Decentralization |
| 1. Who maintains the ledger of transactions?  2. Who has authority over which transactions are valid?  3. Who creates new bitcoins? |
| The key technical problem to solve in building a distributed e‐cash system is achieving distributed consensus |
|  |
| the nodes must agree on exactly which transactions were broadcast and the order in which these transactions happened |
| Paxos |
| Bitcoin breaking traditional assumptions:   1. Incentives (currency) 2. Randomness (as time goes on, the probability that your view of any block will match the eventual consensus view increases) |
|  |
| 2.3 Consensus without identity using a block chain |
| Sybil attack |
| nobody is forced to reveal their real‐life identity, like their name or IP address, in order to participate |
| a transaction is a data structure that contains Alice’s signature, an instruction to pay to Bob’s public key, and a hash. |
| Stealing Bitcoins (X)  Denial of service attack (X)  Double‐spend attack (✔)  honest nodes’ behavior is always to extend the longest valid branch that they see  a cautious merchant would not release the software to Alice even after the transaction was included in one block, and would continue to wait |
| The most common heuristic that’s used in the Bitcoin ecosystem is to wait for six confirmations. |
|  |
| 2.4 Incentives and proof of work |
| block reward：the node that creates a block gets to include a special transaction in that block （this is the only way in which new bitcoins are allowed to be created）  Transaction fees：Whoever creates the block that first puts that transaction into the block chain gets to collect the difference, which acts a transaction fee |
| The key idea behind proof‐of‐work is that we approximate the selection of a random node by instead selecting nodes in proportion to a resource that we hope that nobody can monopolize. |
| Bitcoin achieves proof‐of‐work using hash puzzles  find a nonce satisfies |
| only some nodes even bother to compete in this block creation process |
| the average time between successive blocks produced in the Bitcoin network is about 10 minutes |
| ownership of bitcoins is nothing more than other nodes agreeing that a given party owns those bitcoins. |
| 51‐percent attack |
|  |
| Chapter 3: Mechanics of Bitcoin |
|  |
| in Bitcoin, the entirety of a transaction output must be consumed by another transaction, or none of it |
| To ensure it hasn’t been spent, we need to scan the block chain between the referenced transaction and the latest block |
| Change addresses, Efficient verification, Consolidating funds, Joint payments, Transaction syntax |
| this script just specifies one public key and requires a signature for that public key in order to spend the coins |
| 3.2 Bitcoin Scripts |
| we combine the new transaction’s input script and the earlier transaction’s output script |
| The first two instructions in this script are data instructions — the signature and the public key used to verify that signature — specified in the scriptSig component of a transaction input in the redeeming transaction. |
| Proof of burn:  a script that can never be redeemed |
| Pay-to-script-hash:  the receiver can instead tell the sender send your coins to the hash of this script  it is Bob’s responsibility to specify the fancy script when he wants to redeem the coins. |
| 3.3 Applications of Bitcoin scripts |
| Escrow transactions: creates a MULTISIG transaction  Green addresses: transaction offline  Efficient micro-payments: MULTISIG transaction, sign last one  Lock time:  Smart contract: |
| 3.4 Bitcoin blocks |
|  |
| header is the only thing that’s hashed during mining. So to verify a chain of blocks, all we need to do is look at the headers |
| The only transaction data that’s included in the header is the root of the transaction tree — the “mrkl\_root” field |
| 3.5 The Bitcoin network |
| gossip protocol |
| checks:   1. The first and most important check is transaction validation 2. Second, they check that the outputs being redeemed here haven’t already been spent. 3. they won’t relay an already-seen transaction 4. nodes will only accept and relay “standard” scripts based on a small whitelist of scripts |
| there will be no more disagreement once this block propagates to the network |
| a node will forward a block only if it builds on the longest branch |
| Size of the block chain: 2014 is over 26 gigabytes |
| Lightweight nodes: don’t store the entire block chain |
| 7 transactions per second, which is all that the Bitcoin network can handle |
|  |
| Chapter 4: How to Store and Use Bitcoins |
| Storing bitcoins is really all about storing and managing Bitcoin secret keys |
| Different approaches to key management offer different trade-offs between availability, security, and convenience |
| addresses are encoded so that they can be communicated from receiver to spender: as a text string or as a QR code  Encoding keys: base 58 and QR codes |
| Speeding up vanity address generation |
| But the good news is that cold storage doesn’t have to be online to receive coins |
| hierarchical wallet |
| 4.3 Splitting and Sharing Keys |
| if we're given any K of those pieces then we'll be able to reconstruct the original secret, but if we're given fewer than K pieces then we won't be able to learn anything about the original secret. |
| 2-out-of-N secret sharing： draw a line |
| threshold signature: two-factor security |
| Multi-signatures |
| 4.4 Online Wallets and Exchanges |
| Bitcoin exchanges: like a bank  no transaction actually happened on the Bitcoin block chain  risks:  bank run  bank running a Ponzi scheme  penetrate the security of the exchange |
| 4.5 Payment Services |
| at the end of this process the customer pays bitcoins and the merchant gets dollars, minus a small percentage, and everyone is happy  The payment service handles everything else — receiving bitcoins from customers and making deposits at the end of the day  a payment service has to be an active participant in the exchange markets |
| 4.7 Currency Exchange Markets |
|  |
| Chapter 5: Bitcoin Mining |
| 5.1 The task of Bitcoin miners  1. Listen for transactions  2. Maintain block chain and listen for new blocks.  3. Assemble a candidate block.  4. Find a nonce that makes your block valid.  5. Hope your block is accepted  6. Profit. |
| After you've exhausted all possible nonces for the block header, you'll change the extra nonce in the coinbase transaction |
| even if two different miners were working on a block with identical transactions, the blocks would still differ |
| next\_difficulty = (previous\_difficulty \* 2016 \* 10 minutes) / (time to mine last 2016 blocks)  maintain the property that blocks should be found by the network on average about once every ten minutes |
| 5.2 Mining Hardware |
| Bitcoin actually requires SHA‐256 to be applied twice to a block |
| GPU: overclocked  say you can run your graphics card 50 percent faster but doing so will cause errors in the SHA‐256 computation to 30 percent of the time |
| GPU mining is basically dead for Bitcoin today |
| Mining today is dominated by Bitcoin ASICs |
| outputting shares , or near‐valid blocks |
| 5.5 Mining incentives and strategies |
|  |
| Forking attack  Temporary block‐withholding attacks  Blacklisting and punitive forking  Feather‐forking |
|  |
| Chapter 6: Bitcoin and Anonymity |
| unlinkability:  1. It should be hard to link together different addresses of the same user.  2. It should be hard to link together different transactions made by the same user.  3. It should be hard to link the sender of a payment to its recipient. (ultimate recipient) |
| 6.2 How to De-anonymize Bitcoin |
| shared spending is evidence of joint control |
| Idioms of use. |
| transaction graph analysis & Network level deanonymization |
| 6.3 Mixing |
| 6.4 Decentralized Mixing |
| 6.5 Zerocoin and Zerocash |
|  |
| Chapter 7: Community, Politics, and Regulation |
|  |
| Chapter 8: Alternative Mining Puzzles |
| 8.1 Essential Puzzle Requirements |
| Adjustable difficulty, fast verification, and progress‐freeness |
| 8.2 ASIC‐resistant puzzles |
| Memory‐hard puzzles （Scrypt，Cuckoo Cycle）  memory‐bound puzzles |
| 8.3 Proof‐Of‐Useful‐Work |
| Challenges in adapting useful‐proof‐of‐work  equiprobable solution space  inexhaustible puzzle space  algorithmically generated |
| Primecoin. |
| Permacoin and proof‐of‐storage. |
| 8.4 Nonoutsourceable Puzzles |
| 8.5 Proof‐of‐Stake and Virtual Mining |
|  |
| Chapter 9: Bitcoin as a Platform |
| 9.1. Bitcoin as an Append‐Only Log |
| Secure timestamping  prove that some invention we filed a patent on was actually in our heads much earlier  prove that someone has else has received a message we sent them  Attacks on Proofs‐of‐“Clairvoyance”.（😭） |
| Secure timestamping in Bitcoin：   1. CommitCoin allows you to encode your data into the private key （This allows anyone looking at the block chain to compute the private key, which contains the commitment, using the two signatures） 2. In 2014, the preferred way to do Bitcoin timestamping is with an OP\_RETURN transaction which results in a provably unspendable output |
| Overlay Currencies |
| 9.2 Bitcoins as “Smart Property” |
| Fungibility |
| Colored coins (transact online, fast transaction settlement, and non‐reliance on a bank) |
| Uses of colored coins and smart property |
| 9.3: Secure Multi‐Party Lotteries in Bitcoin |
| a lottery that is just as “fair”, but also solves the problem of making sure the loser pays |
| 9.4: Bitcoin as Public Randomness Source (Cryptographic “Beacons”)  the beacon will continuously publish new random data at a regular rate that nobody can predict in advance |
| 9.5: Prediction Markets and Real World Data Feeds |
|  |
| Chapter 10: Altcoins and the Cryptocurrency Ecosystem |
| 10.7 Ethereum and Smart Contracts |
| In Ethereum, a contract is a program that lives on the blockchain |
| written in bytecode and executed by a special Ethereum‐specific virtual machine |
| Gas, incentives, and security  Every transaction can specify the “gas price”, that is, how much ether it will pay per unit of gas consumed  The gas requirement means that very expensive computations are not suitable for Ethereum  Not cloud computing  Ethereum is suitable for implementing security protocol logic |
| Applications:  prediction markets, smart property, escrowed payments, micropayment channels, and mixing services, auctions and order books |
| Ethereum, on the other hand, uses an account‐based model  Ethereum just stores a balance for each address |
| Patricia tree , also known as a prefix tree, trie, or radix tree |
| Each Ethereum block includes the root of a Merkle Patricia tree committing to the state of every address, including contract addresses. Each contract’s state, in turn, includes a tree committing to the entire state of its storage |
| every account in Ethereum has a transaction counter tracking how many transactions it has sent |
| The block time is targeted at 12 seconds instead of 10 minutes. |
| These will be hard forks by design |
|  |
| Chapter 11: Decentralized Institutions: The Future of Bitcoin? |
| 11.1 The Block Chain as a Vehicle for Decentralization |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |