Chap-5

Date: M T W T F S S

Bitcoin Mining

⇒ Bitcoin depends on miners

1.

Mining Bitcoin in 6 easy steps:

(1) Join network, listen for transactions
↳ validate all proposed transactions

(2) Listen for new blocks, maintain bc.
↳ when a new block is proposed validate it.

(3) Assemble a new valid block.

(4) Find the nonce to make your block valid.

(5) Hope everybody accepts

(6) Profit.

Current mining difficulty
$2^{66.2}_{32}$

Setting the mining difficulty
next-difficulty = previous-difficulty * (2 weeks) / (time to mine last 2016 blocks)

2. Mining Hardware

SHA256
↳ General purpose hash function

⇒ CPU mining

```
while (1)
{
    HDR[KNoncePos]++;
    If (Sha256(Sha256(HDR) <
        (65535 << 208)/Difficulty)
        return;
}
```

⇒ GPU mining
↳ parallel ALUs
↳ bit specific instructions
↳ can drive many from 1 CPU
↳ can overclock

Goodput: throughput x success rate

FPGA mining
↳ Field Programmable Gate Area
high power draw than GPUs more

Date: M T W T F S S

Bitcoin ASICs
↳ special purpose

3. Energy consumption & Ecology

4. Mining Pools
↳ Goal: pool participants all attempt to mine a block with some coinbase recipient.
Distribute revenues to members based on how much they have performed.
minus cut for pool manager

Mining shares
↳ prove work with 'near-valid-blocks' (shares)

Mining pool variations:
↳ Pay-per-share: flat reward per share (minus a significant fees)
↳ Proportional: typically since last block.
↳ low risk for pool manager.
↳ Lukey-jr approach.

Are mining pools a good thing?
↳ More mining more predictable.
↳ Allow small miners to participate
↳ More miners using updated validation software.

Cons
↳ lead to centralization.
↳ Discourage miners from running full nodes.
Can we prevent pools? no.

Mining Incentives & Strategies:
Game-theoretic analysis of mining
Forking attack
↳ if $q > 0.5$
Attack is detectable & reversable

Forking attacks via bribery:
renting $q > 0.5$.
Payment techniques:
=> out of band bribery
=> Run a mining pool at loss
=> Insert large tips in blockchain

P. TO =>

Date: M T W T F S S

Block-withholding attacks
↳ don't announce blocks right away. Try to get ahead.
↳ Selfish mining

Punitive forking
↳ blacklist transaction from address X.
↳ Extreme: stop mining on chain with address X.

Feather-forking strategy
↳ first deny then accept after confirming n blocks
↳ Chances of pruning: α^2

Pros:
↳ Freezing individual bitcoin owners

A second look at transaction fees: $\Rightarrow priority = \frac{sum(input\_value \times input\_age)}{size\_in\_bytes}$
Acc: without fees if priority > 0.576

Bribery attacks
↳ start new mining pool
↳ pay miners directly
↳ kickbacks

Date: M T W T F S S

Chap- 6
Bitcoin and Anonymity

1. Anonymity Basics

⇒ Bitcoin addresses are public key hashes rather than real identities.

⇒ Computer Scientist call this Pseudonymity.

Anonymity = Pseudonymity + unlinkability

Defining unlinkability in Bitcoin:

(1) Hard to link different addreses of same user.

(2) Hard to link different transactions of some user.

(3) Hard to link sender of a payment to it's recipient

Quantifying Anonymity

↳ Anonymity set

↳ adversary model, what adversary knows, doesn't know, cannot know

Anonymous e-cash via blind signatures.

Anonymity & decentralization in conflict

2. How to de-anonymize Bitcoin?

⇒ Best practise: always receive at a fresh address

Shared spending is evidence of joint control

Clustering of addresses

Idioms of use

↳ each address used only once as change.

From Services to users

↳ (i) high centralization in service providers

(ii) Address - identity link in forums

Network layer de-anonymization

⇒ transaction graph analysis

Date: M T W T F S S

Solution: use Tor
↳ Caveat: low latency (Browsing)
Blockchain => high latency

- Mix nets provide better anonymity

3. Mixing
=> To protect anonymity, use an intermediary.
=> Online wallets do this

Dedicated mixing services
↳ Promise not to keep records
↳ Don't ask for your identity.

Principles for mixing services
1. Use a series of mixes
↳ (Source) → A → B → C → Dest
Mixcoin paper
2. Uniform transactions
in particular: all mix transactions must have the same value.
3. Client side must be automated. (Desktop wallet software)
4. Fees must be all-or-nothing
↳ Probabilistic fees

Current mixes follow none of these principles. :clown:

Remaining problem: trusting mixes
1. Stay in business
2. Users can test for themselves
3. Cryptographic warrentles

=> Currently no reputable dedicated mix.

4. Decentralized Mixing
Why?
=> No bootstrapping problem
=> Theft impossible
=> Possible better anonymity
=> More philosophically aligned with Bitcoin.

Coinjoin algorithm:
(i) Find peers who went to mix
(ii) Exchange input/output addresses
(iii) Construct transaction
(iv) Send it around, collect signatures
(Before signing, each peer checks if her output is present)
(v) Broadcast the transaction.

Date: M T W T F S S

Coinjoin: remaining problems
⇒ How to find peers
⇒ Peers know your input–output mapping. (Worse problem than for centralized mixing)
⇒ Denial of service

Peer anonymity:
⇒ Tor or special purpose anonymous routing mechanism

Denial of service solution:
⇒ Proof of work / Burn
⇒ Cryptographic "blame" protocol

High-level flows could be identifying

Heuristic: merge avoidance
⇒ Instead of a single payment receiver provides multiple output addresses sender avoids combining different inputs.

5. Zerocoin and Zerocash

⇒ Zerocoin ⇒ protocol-level mixing → Pro
⇒ cryptographic guarantee of mixing.
⇒ Cons: not currently compatible with bitcoin

Basecoin and Zerocoin

Basecoin → Zerocoin
(123) ⇓
Basecoin (456)

Zerocoin: a cryptographic proof that you owned a Basecoin and made it unspendable.
⇒ Miners can verify these proofs.
⇒ Gives you the right to redeem a new basecoin

Two Challenges:
⇒ How to construct these proofs?
⇒ How to make sure each proof can only be spent once?

Date: M T W T F S S

Zero Knowledge Proofs

Minting Zerocoins
↳ Value after transaction on blockchain via basecoin

Generate Serial number: S
eventually made public

random secret r
(never public, ensures unlinkability)

Compute H(S, r)

To Spend a Zerocoin S:
⇒ Reveal S
(miners will verify S hasn't been spent before)
⇒ Create zero-knowledge proof that:
"I know a number r such that H(s, r) is one of the zerocoins in the block chain.

Zerocoin as anonymous because of r.

Zerocoin is "efficient"
⇒ proof is a disjunction over all zerocoins
Yet the proof is relatively small

Zerocash: Zerocoin without basecoin

Two differences:
Different crypto for proofs (More efficient.
⇒ Proposal to run system without basecoin.

Zerocash: untraceable e-cash
⇒ All transactions are zerocoins
⇒ Splitting and merging supported
⇒ Put transaction value inside the envelope.
⇒ ledger merely records existence of transactions.

The Catch: Random, secret inputs are required to generate public parameters

Date: M T W T F S S

6. Tor and the Silk road

=> safe(ish) if at least one router honest

=> Key challenge: hiding routing information

=> solution: layered encryption

Hidden Services:

(1) Connect to "rendezvous point" through Tor

(2) Publish name → rendezvous point mapping

(3) Client connects to rendezvous point

Chap-7

Community, Politics & Regulation

7.1: Consensus in Bitcoin

(i) Consensus about rules

(ii) Consensus about history

(iii) Consensus about value

Tinkerbell effect: exist because you believe in it.

7.3: Stakeholders: who's in charge

Who has power?

(i) Core developers

(ii) Miners

(iii) Investors

(iv) Merchants & their customers

(v) Payment Service

=> Bitcoin advocacy group

=> Coin center

7.4 Roots of Bitcoin

=> Cypherpunk and digital cash

=> Satoshi Nakamoto.

=> Growth

Date: M T W T F S S

7.6: Anti Money-Laundering
KYC: Know Your Customer
(i) Identify and authenticate clients
(ii) Evaluate risk of client
(iii) Watch for anomalous behavior.

Mandatory Reporting:
greater than $10,000

· Collusion and antitrust law:
↳ Price fixing (Bakery example)

7.7 Regulation
Alternate allocation of goods to the market participants that would result in everybody being better off, or at least not worse off. Such an alternate allocation is called a Pareto improvement.

Lemons market
Regulatory Fixes:
↳(i) Disclosure
(ii) Quality Standards
(iii) Warranties

Date: M T W T F S S

Chap-8

Alternative Mining Puzzles

⇒ Incentive system steers participants

⇒ Variety of possible goals

↳ ASIC resistance, pool resistance, intrinsic benefits

8.1: Essential Puzzle Req.

⇒ Cheap to verify

⇒ Adjustable difficulty

⇒ Chance of winning is proportional to hashpower.

↳ Large players get only proportional advantage

↳ Small players get proportional compensation

Bad puzzle: a puzzle that takes N steps to solve a "Sequential" proof of work

↳ Fastest miner always wins

⇒ Good puzzle ⇒ Weighted sample

↳ property is called progress-free

3 crucial properties:

⇒ Adjustable difficulty

⇒ fast verification

⇒ progress-freeness

Sha256-based partial pre-image finding

8.2: ASIC-resistant puzzles

Approach: reduce the gap b/w custom hardware and general purpose equipment

Solution: Memory hard puzzles

Scrypt:

↳ Memory hard hash function

↳ also used in password hashing

1. Fill memory with random values
2. Read from memory in random order.

Scrypt Step 1 of 2

Input: x

$V_1 = H(x)$

$V_2 = H(V_1)$ or $H(H(x) \| 2)$

$V_n = H^N(x)$

Date: M T W T F S S

Scrypt 2/2 (read)

Input : X

$A = H^{N+1}(X)$

For N iterations

$i = A \bmod N$

$A = H(A \text{ xor } V_i)$

Output: A

Disadvantages:

=> Verification takes the some time as computation

=> requires N steps

=> Scrypt ASICS are readily available

Cuckoo hash cycles

=> Memory hard puzzle that's cheap to verify.

Input: X

for i=1 to E:

$a = H_0(X+i)$

$b = N + H_1(X+i)$

edge(a mod N, b mod N)

Is there a cycle of size K?

If so output: X, K edges

More approaches:

=> X11: 11 different hash functions combined

=> Moving target: Change the puzzle periodically

Counter argument: Sha2 is fine

=> only marginal efficency in ASICs machines.

8.3 Proof-of-Useful-Work

=> equiprobable solution space

=> inexhaustible puzzle space

Primecoin:

↳ Puzzle based on finding large prime numbers

Cunningham chain

$p_1, p_2, \ldots p_n$

where $p_i = 2^i q + 1$

Permacoin: Mining with Storage

↳ Side effect: Massively distributed, replicated storage system

P.T.O →

Date: M T W T F S S

Storage-based puzzle

1. Build a Merkle tree, where each leaf is a segment of the file
2. Generate a public signing key PK, which determines a random subset of file segment
3. Each mining attempt:

(a) Select a random nonce

(b) $h_1 = H(prev \| mrkl\_root \| PK \| nonce)$

(c) h_1 selects k segments from subset

(d) $h_2 = H(prev \| mrkl\_root \| PK \| nonce \| F)$

(e) Winner if $h_2 <$ Target

Summary:

=> Natural goal

=> pure public good

=> Realized benefit has limited.

8.4: Nonoutsourceable Puzzles

Large mining pools are threat

=> Bitcoin's core value is decentralization

=> Position: large pools should be discouraged

The vigilante attack

=> submit shares like normal, but if he finds a real solution, discards it.

Nonoutsourceable puzzle:

Solution:

(prev, mrkl_root, nonce, PK, s_1, s_2)

↳ Requires private key for signing.

=> discourage all pools

↳ including harmless P2pools.

Date:
M T W T F S S

8.5: Proof-of-stake and Virtual Mining

Virtual Mining: winners chosen at random by lottery.

Potential benefits:

⇒ lower overall costs
- ↳ No harm to enviroment
- ↳ Saving distolbuted to all coin holders
- ↳ 51% attack is even harder.

Variations of Virtual Mining

⇒ Proof of stake: "Stake" of a coin grows over time as longs as the coin is unused.

⇒ Proof of burn: mining with a coin destroys it.

⇒ Proof of deposit: can reclaim a coin after some time.

⇒ Proof of activity: can coin might be win (if online)

⇒ waste is the cost of security

Date: M T W T F S S

Chap-9
Bitcoin as a Platform

9.1: Bitcoin as an Append-Only Log

Secure timestamping:
Goal: prove knowledge of x at time t.
⇒ If desired, without revealing x at time t.
Evidence should be permanent.

Hash commitments
Publishing H(x) is a commitment to x.

Secure timestamping applications
⇒ Proof of Knowledge
⇒ Proof of receipt
⇒ Hash-based signature schemes.
Offline solution: newspeper timestamp

Timestamping in Bitcoin
⇒ Specify the hash of your data instead of a valid public key.
⇒ Send 1 Satoshi to the address.

Pros: compatible, easy
Cons: creates unspendable UTXO forever.

Commitcoin: Brute force a public key & signature starting with the first n bits of your data hash.
Pros: Compatible, "invisible", no UTXO bloat.
Cons: more expensive, low data rate.

9.2: Bitcoin as "Smart Property"

⇒ Every bitcoin carries a history.
- Bad for anonymity
- Enables blacklisting
- Observations: bitcoins aren't fungible! Everyone is unique.

~~Coin~~ There are no bitcoins, just unspent tx outputs.

Authenticated metadata for currency:
⇒ Sign desired metadata + banknote serial #

Date: M T W T F S S

=> Currency can now represent anything.
=> Anti-counterfeiting properties are inherited.
=> Underlying value also maintained
=> New meaning relies on trust in the issuer.
=> Some users may not understand new metadata

Colored Coins:

Pros:

=> Compatible with Bitcoin
=> flexible to represent any asset
=> ignored by community

=> Cons:

=> Small cost of unspendable ~~transaction~~ markers.
=> must check every previous transaction.

Applications:

=> stock certificates
=> tickets
=> deeds to real-world property houses? cars?

=> Ownership of domain names
=> Namecoin.

9.3: Secure Multi-party Lotteries in Bitcoin

Hash commitments:

Publishing $H(x)$ is a commitment to x.

=> can't find an $x' \neq x$ later s.t $H(x') = H(x)$
=> $H(x)$ reveal no information about x.

→ time →

Alice	chose x	Publish $H(x)$	Publish x
Bob	y	$H(y)$	y
carol	z	$H(z)$	z

Timed hash commitments

=> Force x to be revealed by time t.

Step-1: Input: ....;
Pay B(Bond) to either of:
Alice & Bob, or (Multisig)
Alice (anybody who knows x s.t $H(x) = c$ => New script Signed Alice

P.T.O →

Date: M T W T F S S

Step-2 Input: 1, Pay Bond to Bob:
n_lock_time : t
↳ Bob can claim the bond at time t.
Signed(Alice) Signed (Bob).

Step-3: Input: 1, Pay Bond to Alice Signed(Alice), x
x revelead if Alice reclaims her bond

Lottery with timed commitments
Pros:
=> can be implemented on bitcoin today.
Cons:
=> complexity is $O(N^2)$
=> bonds must be higher than amount bet.
griefers still might shut down large pools.

9.4: Bitcoin as Public Randomness Source

Cryptographic beacons
↳ service to regularly publish random data.
=> Uniform randomness
=> No party can predict in advance
=> All parties see the same values
Applications: lotteries, auditing, Zero-knowledge proofs, cut-and-choose

=> Public display of randomness
=> NIST beacon (Quantum)
=> Natural phenomena
=> Stock Market beacon

Why not use the blockchain?
=> Miners find random nonce for each block.

If you could predict the next nonce with a greater than 1/d probability, you'd have

Date: M T W T F S S

a mining shortcut
Currently $d > 2^{66}$

Cost of Manipulation
⇒ Attacker might mine a block but discard it.
Or bribe other miners to do so

Bernoulli trials
Discarding a block costs 12.5 btc

Pros:
⇒ First proposal for fully decentralized beacon.
⇒ Output every 10 minutes.
⇒ Can precisely analyze manipulation costs
⇒ Can extend security with multiple blocks.
Not very efficient.

Cons:
⇒ Timing is imprecise
↳ Not sync w/ real time.
⇒ Need to delay to insure against forks.

⇒ Manipulation may be too cheap for some applications.

Built-in beacon support in script:
⇒ add an opcode for a beacon call.
⇒ can build multi-party lotteries
↳ only one round
↳ no bonds
↳ no time delay for refunds

Prediction markets:
↳ trade shares in a potential future event.
Shares worth X if the event happens, 0 if not
Current price / X = estimated probability.

Date: M T W T F S S

Chap-10

Altcoins and the Cryptocurrency Ecosystem

10.1: Altcoins: History & Motivation

Bitcoin is not alone.
Between 500-1000 altcoins launched to date.
Bitcoin & litecoin are 99% of total.

Features of altcoins:
=> Better (or different) security
  ↳ Mining puzzle
=> contract / platform features.
=> Different parameters and monetary policy
  ↳ inflation, inter block time
=> Community or common interest support.

10.2: A few altcoins in detail

Namecoin => first altcoin (launched in april 2011)
Feature: Domain Name Registration
  example.bit
Can be merge-mined with bitcoin.

Litecoin: launced, Sep 2011
=> Memory hard mining puzzle
=> 4th most popular, 1st most widely forked
=> Block rate is 4x faster.

PeerCoin (PPcoin)
=> launched August 2012
Hybrid mining:
=> First Proof-of-stake algorithm
  ↳ mine by spending "stake"
=> Proof of work can earn mining rewards
  ↳ but aren't counted for choosing the main chain.
=> Also uses regularly published "checkpoints"
  ↳ acts as safeguard, planned to remove in future.

Dogecoin: Culture
↳ launched in December, 2013
Culture, tipping, charity, sponsorship

Dogecoin: Random block rewards
-Each block is "random"
=> block bonus is pseudorandom function of previous block chain.

Date: M T W T F S S

=> miners know next reward in advance.
Switch to other altcoin when reward is low.
=> Feature removed in 2014.

Metrics for comparing altcoins
=> Market cap (price * total no. of coins)
↳ Overestimates value (but how much?)
↳ Doesn't account for lost/out-of-circulation coins
=> Exchange volume
↳ depends on nature of third party exchanges.
↳ can be moved deliberately
=> Total hashpower
=> Merchant support /usage?

10.3: Relationship between Bitcoin and Altcoins

Mining Attacks
=> Even a small miner (or ming pool) on a large network can demolish a small action.

10.4: Merge Mining

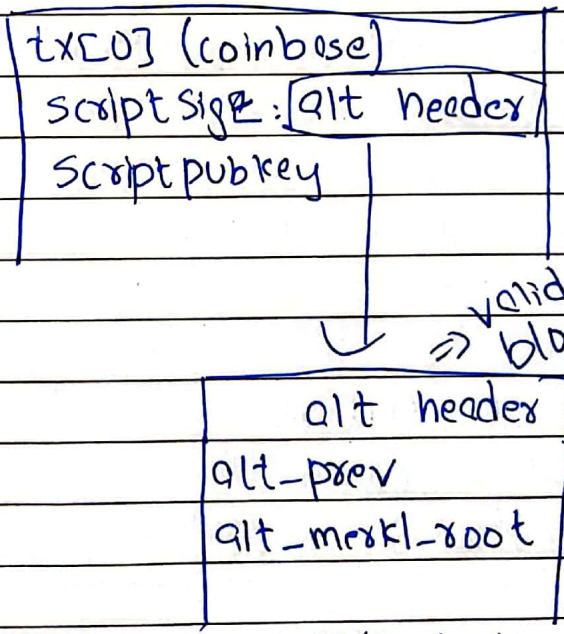
=> Ordinarily, mining is exclusive
=> Each attempt either has a chance to be a bitcoin block, or has a chance to be an Altcoin block.

Obstacle to bootstrapping

H(prev || merk-root || nonce) < target
(btc/altcoin → btc → altcoin → btc → altcoin)

How it works:

H(prev || merkl-root || nonce) < target



Coinbase scriptSig is ignored by btc.

Date: M T W T F S S

Merge mining is a mixed blessing.
=> Easier to recruit participants
=> Cheaper for attackers (coiled coin)
=> Miner might not validate transactions

10.5: Atomic Cross-chain Swaps

Problem: Alice has 1 BTC, Bob has 1 LTC
They want to swap but who goes first?
Goal: Either both transactions, complete or neither do.

Step-1: Alice generates secret x, Alice & Bob sign RefundA

Alice(BTC) x, $h = H(x)$

DepositA BTC
Either sigA and sigB
Or sigB and
reveal x where $H(x) = h$

RefundA
Timelocked to T+2
Signed by Bob
Signed by Alice

=> Alice generates DepositA, but doesn't publish it yet.
=> Alice generates RefundA, and gets Bob's signature on it.
=> Once RefundA is signed she publishes DepositA

If Bob learns x before time T+2, he can take the 1 BTC.
If Alice doesnot reveal x, she can claim her refund at T+2.

Step-2: Bob deposits 1 LTC, Alice & Bob sign RefundB
=> Bob generates DepositB, but doesn't publish it yet
=> Bob generates RefundB, and gets Alice signature on it.
=> Once RefundB is signed, he publishes DepositB.

If Alice reveals x before time T+1, she can take the 1 LTC.

Date: M T W T F S S

If Alice does not reveal x, Bob can claim his refund.

DepositB	LTC
Either sigA and sigB	
Or sigA and	
reveal x where $H(x) = h$	

⇓

RefundB
Timelocked to T+1
Signed by Bob
Signed by Alice

Step 3: Alice reveals x, both players claim their coins.

Alice LTC

DepositA	→	RefundA
Either sigA and sigB		Timelocked to T+2
Or sigB and		Signed by Bob
reveal x where $H(x) = h$		Signed by Alice

Bob BTC

DepositB	→	RefundB
Either sigA and sigB		Timelocked to T+1
Or sigA and		Signed by Bob
reveal x where $H(x) = h$		Signed by Alice

Date:
M T W T F S S

=> If Alice does not reveal x, Bob can claim his refund at T+1.
=> If Alice takes the 1 LTC she reveals x before time T+1.
=> If Bob learns x before time T+2, he can take the 1 BTC.
=> If Alice does not reveal x, she can claim her refund at T+2.

Atomic cross chain swaps
=> Decentralized exchange blw Altcoins
=> Not been seen in wild
↳ disadvantages: multiple transactions, DOS risk
=> Third party exchanges are used instead

Hash commits: interdependent transactions.
=> Possible with existing script languages