Blockchains & Distributed Ledgers

Lecture 06 - Economics and Incentives

Aggelos Kiayias

Slide credits: AK, Dionysis Zindros, Christos Nasikas

Mining parameters

Mining incentives

A miner is incentivized to mine with 2 ways:

(1) Fees and (2) *rewards*

Mining fees

Fees are the remaining money from the Law of Conservation for confirming transactions:

$$\text{fees} = \sum_{tx \in block} \left[\sum_{i \in in(tx)} w(i) - \sum_{o \in out(tx)} w(o) \right]$$

These fees are claimed by the miner who included the respective transaction

w(.) = value

Mining rewards

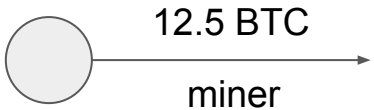
A miner is given a **reward**, today equal to 12.5 BTC

The *reward* and the *fee* are called the **coinbase** and together can be claimed by the miner.

Example: fees = 1 BTC, rewards = 12.5 BTC, total coinbase = 13.5 BTC

The coinbase transaction

- The **coinbase transaction** is the transaction by which a miner is paid the fees and the rewards
- There can be only **one coinbase transaction** per block
- It is the **first** transaction that appears in the block
- It has *no inputs*
- This is the only way fresh bitcoins are generated



Coinbase transaction validity

- The **induction basis** for transaction validity
  (we have seen the inductive **step** when we talked about UTXO evolution)
- Has no inputs, so does not conform to the Law of Conservation!
- Never forms part of the mempool -- only included in blocks for the first time
- When a block is confirmed, the coinbase is checked for validity:
  - It is the first in the block
  - There's only one of it
  - **output value ≤ block reward + block fees**
- Hence, a malicious miner cannot generate more money for themselves

The coinbase transaction

- As it does not have any inputs, scriptSig can be anything for coinbase
- scriptSig is used for certain block metadata:
  - The block height (this is verified for validity)
  - The name of the mining pool that mined the block
  - Extra entropy (nonce) if the block header nonce (“ctr”) has been exhausted
  - Signalling for protocol updates
    (whether a miner is in favour of an upgrade or not -- more on this in a later lecture)

Money supply in bitcoin

- The **money supply** in bitcoin is **algorithmically predetermined**
- Achieved with an **algorithm** known beforehand to all
- Concretely:
  - The coinbase of **genesis** has **reward 50 BTC**
  - Each next block has reward equal to its previous block
  - Every 210,000 blocks (expected every **4 years**) the reward is **halved**
  - The duration during which rewards stay the same is known as an **era**

total_btc_supply = $$\frac{\sum_{i=0}^{32} 210000 \lfloor \frac{50.10^8}{2^i} \rfloor}{10^8}$$

number of eras until reward is negligible

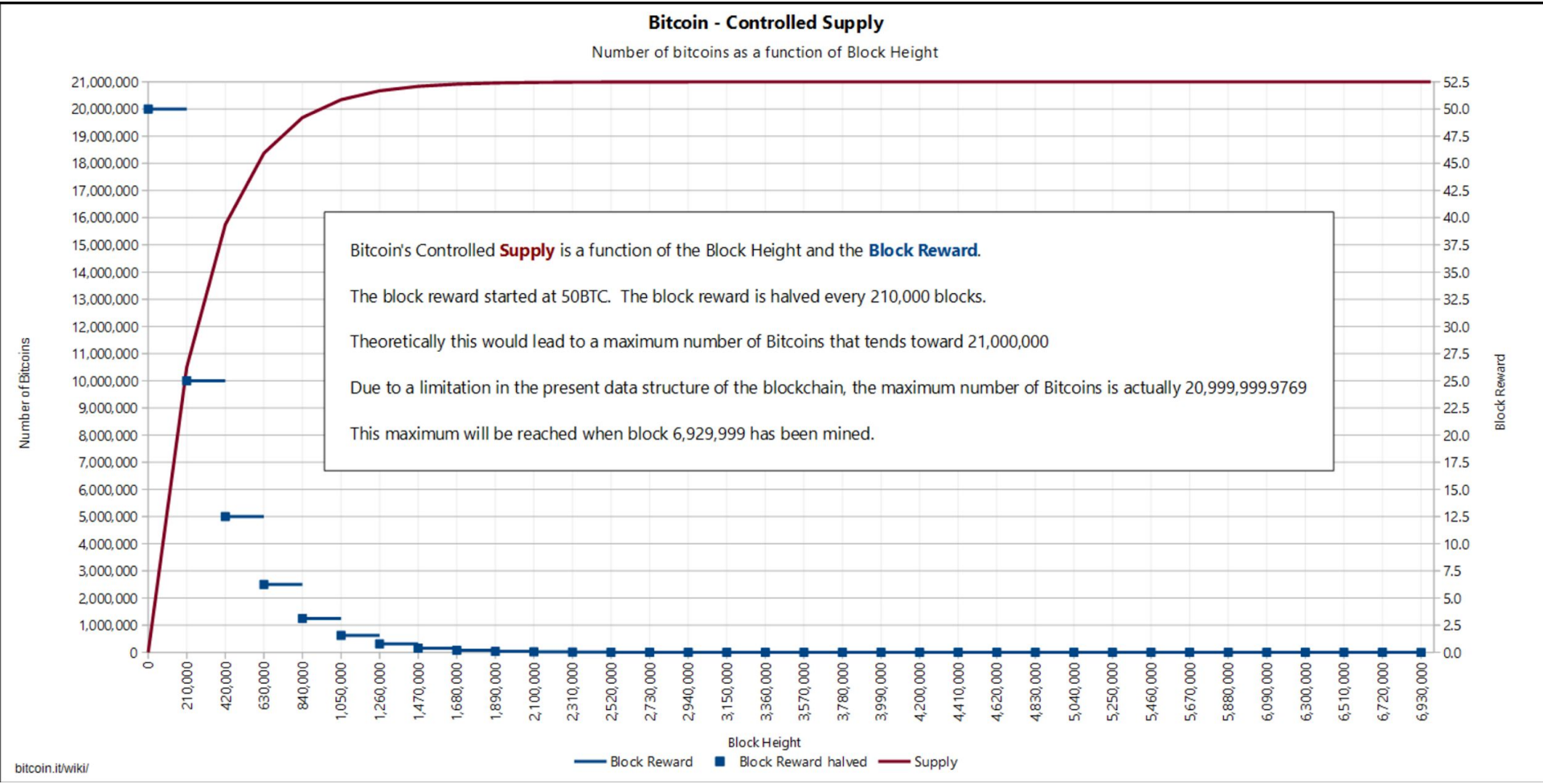
era duration in blocks

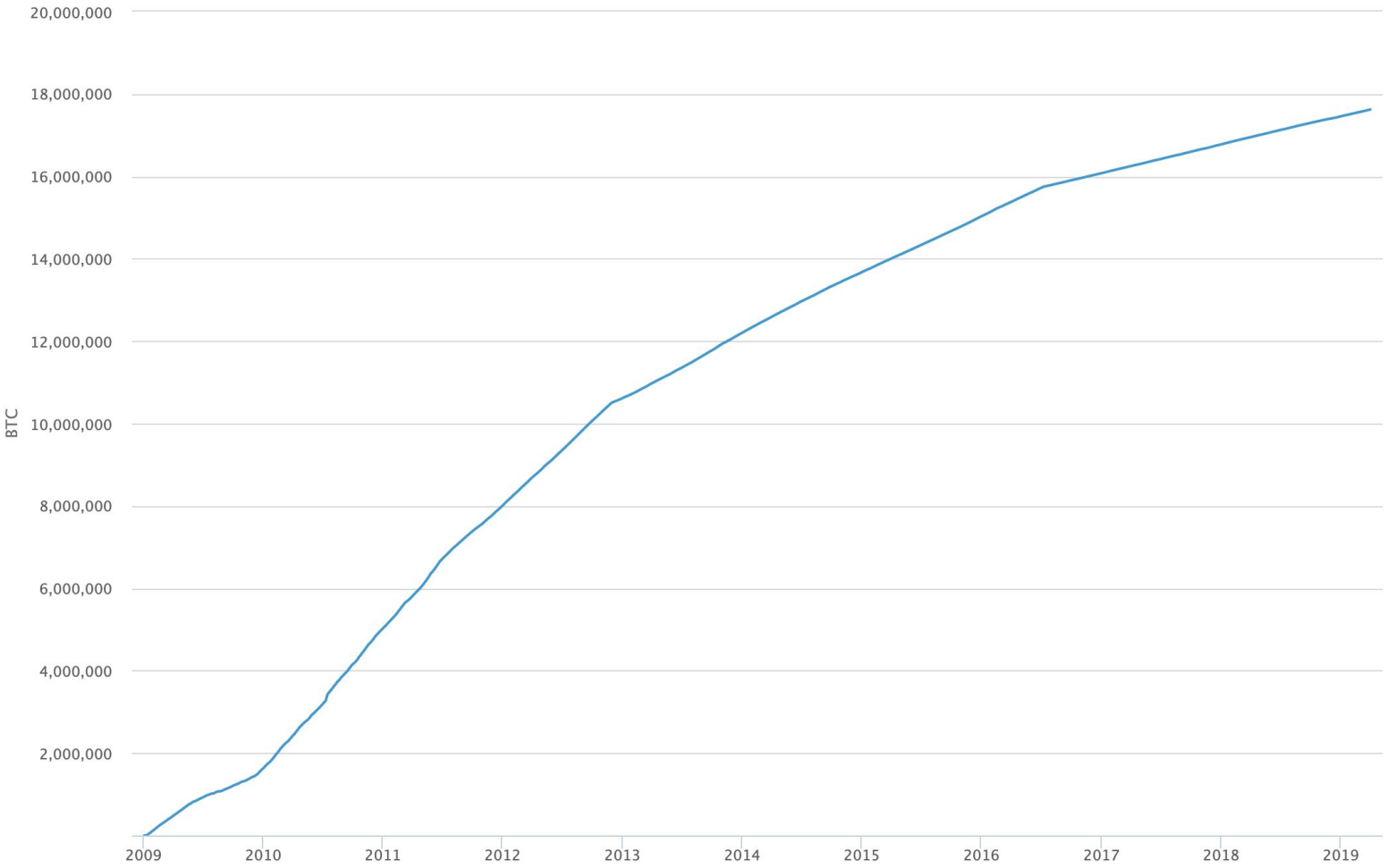
genesis block reward

$$\text{total\_btc\_supply} = \frac{\sum_{i=0}^{32} 210000 \left\lfloor \frac{50 \cdot 10^8}{2^i} \right\rfloor}{10^8}$$

satoshi / bitcoin

Date reached	Block	Reward Era	BTC/block	Year (estimate)	Start BTC	BTC Added	End BTC	BTC Increase	End BTC % of Limit
2009-01-03	0	1	50.00	2009	0	2625000	2625000	infinite	12.500%
2010-04-22	52500	1	50.00	2010	2625000	2625000	5250000	100.00%	25.000%
2011-01-28	105000	1	50.00	2011*	5250000	2625000	7875000	50.00%	37.500%
2011-12-14	157500	1	50.00	2012	7875000	2625000	10500000	33.33%	50.000%
2012-11-28	210000	2	25.00	2013	10500000	1312500	11812500	12.50%	56.250%
2013-10-09	262500	2	25.00	2014	11812500	1312500	13125000	11.11%	62.500%
2014-08-11	315000	2	25.00	2015	13125000	1312500	14437500	10.00%	68.750%
2015-07-29	367500	2	25.00	2016	14437500	1312500	15750000	9.09%	75.000%
2016-07-09	420000	3	12.50	2016	15750000	656250	16406250	4.17%	78.125%
2017-06-23	472500	3	12.50	2018	16406250	656250	17062500	4.00%	81.250%
2018-05-29	525000	3	12.50	2019	17062500	656250	17718750	3.85%	84.375%
	577500	3	12.50	2020	17718750	656250	18375000	3.70%	87.500%
	630000	4	6.25	2021	18375000	328125	18703125	1.79%	89.063%
	682500	4	6.25	2022	18703125	328125	19031250	1.75%	90.625%
	735000	4	6.25	2023	19031250	328125	19359375	1.72%	92.188%
	787500	4	6.25	2024	19359375	328125	19687500	1.69%	93.750%


Bitcoin - Controlled Supply
Number of bitcoins as a function of Block Height
Bitcoin's Controlled Supply is a function of the Block Height and the Block Reward.
The block reward started at 50BTC. The block reward is halved every 210,000 blocks.
Theoretically this would lead to a maximum number of Bitcoins that tends toward 21,000,000
Due to a limitation in the present data structure of the blockchain, the maximum number of Bitcoins is actually 20,999,999.9769
This maximum will be reached when block 6,929,999 has been mined.
Number of Bitcoins
Block Reward
Block Height
Block Reward
Block Reward halved
Supply
bitcoin.it/wiki/


20,000,000
18,000,000
16,000,000
14,000,000
12,000,000
10,000,000
8,000,000
6,000,000
4,000,000
2,000,000
BTC
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019

Bitcoin’s price

- Extremely volatile
- April 2019: 1 BTC = 4,429 EUR
- mid 2018: 1 BTC = 5,502 EUR
- late 2017: 1 BTC = 17,000 EUR
- beginning 2015: 1 BTC = 208 EUR
- max 2013: 1 BTC = 900 EUR
- min 2013: 1 BTC = 73 EUR
- 2012: 1 BTC = 4 EUR
- 2010: 1 BTC = 0.06 EUR
- 22 May 2010: First real purchase with bitcoin

22 May 2010: One pizza for 10,000 BTC

Bitcoin Price (BTC)

$5,001.55 +$4,969.94 (37.2K%)

1H 24H 1W 1M 1Y ALL

Market capitalization

Total bitcoin supply:

04 April 2019: 17,626,900 BTC = $87,655,483,440.00 USD

Bitcoin market capitalization

coinmarketcap.com

Total Market Capitalization

Mining target

Recall the PoW equation: **H(x || ctr || previd) < T**

Pr[a query yields a valid block] = T / 2^{κ}

- T is **the target**
- In bitcoin, κ = 256
- T is the same for all miners
- Block validity **requires proof-of-work validation**
- Adversarial blocks with insufficient proof-of-work are rejected as invalid

Difficulty adjustment

- $T_0 = 2^{256 - 32}$
- The target T is not constant but **dynamically adjusted**
- **This is the way the network ensures a rate of 1 block per 10 minutes**
- T is adjusted collectively by the network
- Adjustment happens every 2016 blocks (expected 2 weeks)
- The period of 2016 blocks is known as an **epoch**
- Difficulty is adjusted so that if the previous 2016 blocks had been generated with the *new* **T**, they would have taken *exactly* 2 weeks

The target adjustment equation

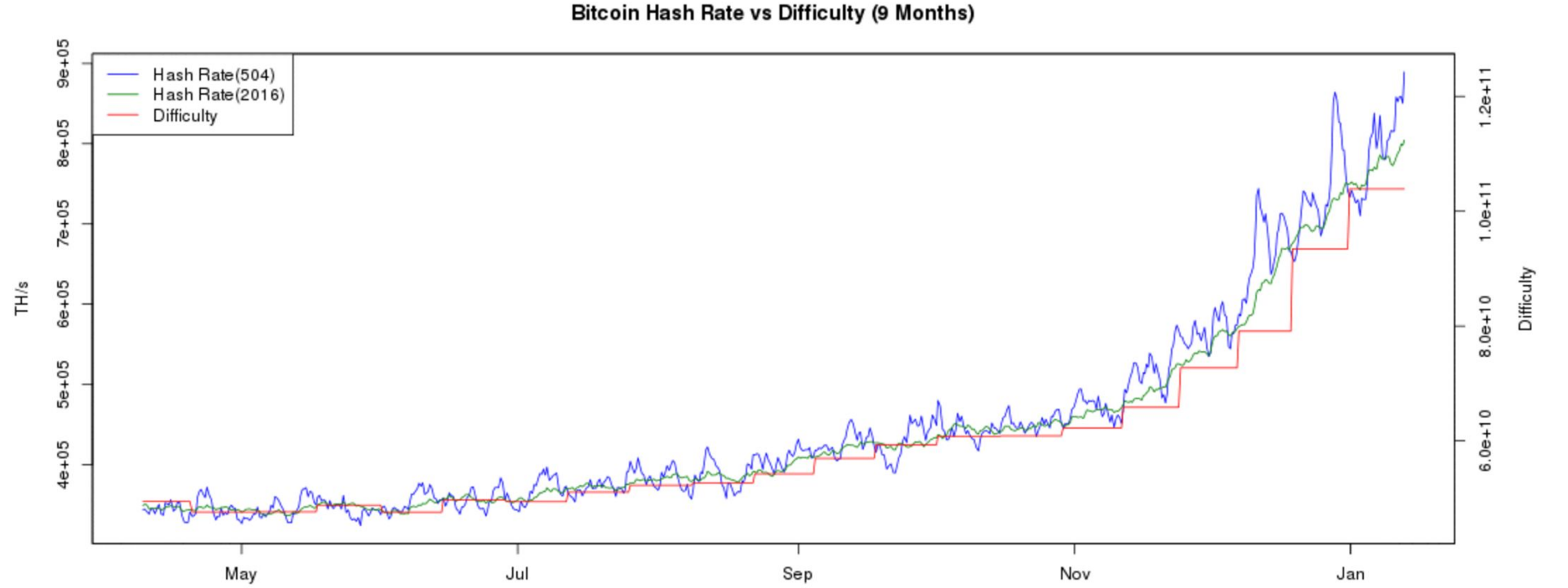
$$T' = T \frac{\mathrm{TS}(\mathsf{block}_n) - \mathrm{TS}(\mathsf{block}_{n-2016})}{2 \text{ weeks}}$$

Difficulty

- The inverse of the target is known as the **difficulty**

$$\textbf{difficulty} = T_0 / T$$

$$\textbf{difficulty}_0 = 1$$


Bitcoin Hash Rate vs Difficulty (9 Months)
Hash Rate(504)
Hash Rate(2016)
Difficulty
TH/s
4e+05
5e+05
6e+05
7e+05
8e+05
9e+05
Difficulty
6.0e+10
8.0e+10
1.0e+11
1.2e+11
May
Jul
Sep
Nov
Jan

Bitcoin denominations

- **1 bitcoin** is divisible up to **10^{-8}**
- 10^{-8} bitcoin = 1 **satoshi**
- 1 satoshi = 0.00000001 BTC
- 1 BTC = 100,000,000
- The bitcoin implementation stores **integers** in the output edges, representing the number of satoshis
- Hence, there are no floating-point errors

Mining pools

- Mining gives a high reward per block (£80,000)
- Has a small probability of success
- The variance is high
  -- I prefer to get £1600 per month than £80,000 per 4 years
- Miners typically collaborate in **mining pools**
- If a miner in a pool finds a block, the proceeds are split among the pool members
- Splitting is *pro rata* according to the computational power contributed by each
- Miners that mine outside of pools (very rare) are called **solo**
- Pools are maintained by a trusted, centralized **pool owner**

Mining inside a pool

- The pool maintains a different **internal** target for proof-of-work $T_{pool} > T_{bitcoin}$
- If a block satisfies $T_{bitcoin} < H(B) < T_{pool}$, it is called a **share**
- The miners of the pool mine as follows:
  - They include a coinbase tx with output the pool address
  - If $H(B) < T_{bitcoin}$, they **broadcast the block to the bitcoin network**
  - If $H(B) < T_{pool}$, they **broadcast the share inside the pool**

Pool share verification

Pool owner verifies shares:

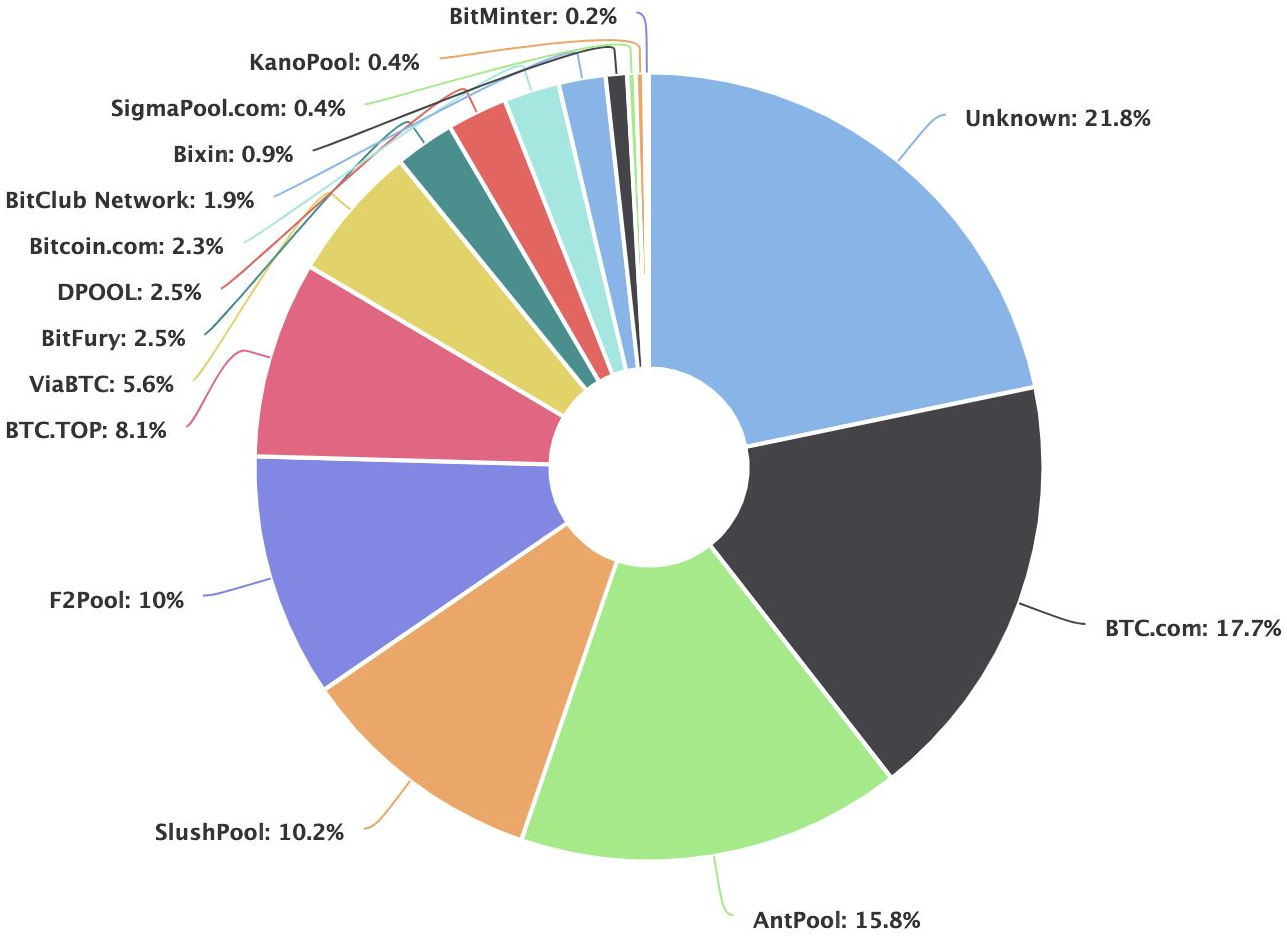
- Check that PoW is achieved with $\mathbf{T_{pool}}$
- Check that coinbase tx pays to the **pool address** and not some other address

Pool rewarding

- When a bitcoin block is created, each node in the pool is rewarded *proportionally* to the pool blocks they have recently generated (compared to the total number of pool blocks generated)
- Node participants pay a participation fee to the **owner** of the pool
- Pools are a trusted scheme: Miners trust the pool owner, but the pool owner does not trust the miners. Miners don’t trust the other miners in the pool.
- The pool owner can steal money, but they will be detected

- Why can’t a pool miner mine shares with the pool address, but blocks with his own address?

- Why can’t a pool miner mine shares with the pool address, but blocks with his own address?
- They don’t know if it will be a share or a block during mining! After mining is completed, changing the address will invalidate the PoW.


BitMinter: 0.2%
KanoPool: 0.4%
SigmaPool.com: 0.4%
Bixin: 0.9%
BitClub Network: 1.9%
Bitcoin.com: 2.3%
DPOOL: 2.5%
BitFury: 2.5%
ViaBTC: 5.6%
BTC.TOP: 8.1%
F2Pool: 10%
SlushPool: 10.2%
AntPool: 15.8%
BTC.com: 17.7%
Unknown: 21.8%

	Pool	Hashrate Share	Hashrate	Blocks Mined	Empty Blocks Count	Empty Blocks Percentage	Avg. Block Size (Bytes)	Avg. Tx Fees Per Block (BTC)	Tx Fees % of Block Reward
0	NETWORK	100.00 %	45.33 EH/s	141	2	1.42 %	1,223,954	1.08225240	8.66 %
1	AntPool	19.15 %	8.68 EH/s	27	0	0.00 %	1,223,365	1.08066490	8.65 %
2	BTC.com	16.31 %	7.39 EH/s	23	1	4.35 %	1,233,210	1.03386865	8.27 %
3	Poolin	10.64 %	4.82 EH/s	15	0	0.00 %	1,225,077	1.11124874	8.89 %
4	SlushPool	9.22 %	4.18 EH/s	13	0	0.00 %	1,248,950	1.13871978	9.11 %
5	F2Pool	8.51 %	3.86 EH/s	12	0	0.00 %	1,191,480	1.10212394	8.82 %
6	unknown	7.80 %	3.54 EH/s	11	0	0.00 %	1,232,072	1.16722453	9.34 %
7	BTC.TOP	7.09 %	3.21 EH/s	10	0	0.00 %	1,189,027	1.05074060	8.41 %
8	ViaBTC	4.96 %	2.25 EH/s	7	0	0.00 %	1,180,631	1.05787656	8.46 %
9	DPOOL	4.26 %	1.93 EH/s	6	0	0.00 %	1,301,065	1.17691249	9.42 %
10	BitClub	3.55 %	1.61 EH/s	5	0	0.00 %	1,254,027	1.15430945	9.23 %

Ways to mine

CPU: Mining takes place in the CPU

GPU: Mining takes place in the graphics card (high parallelization)

ASIC: Specialized hardware for mining, separate hardware device


+12V
GND
ANTMINER

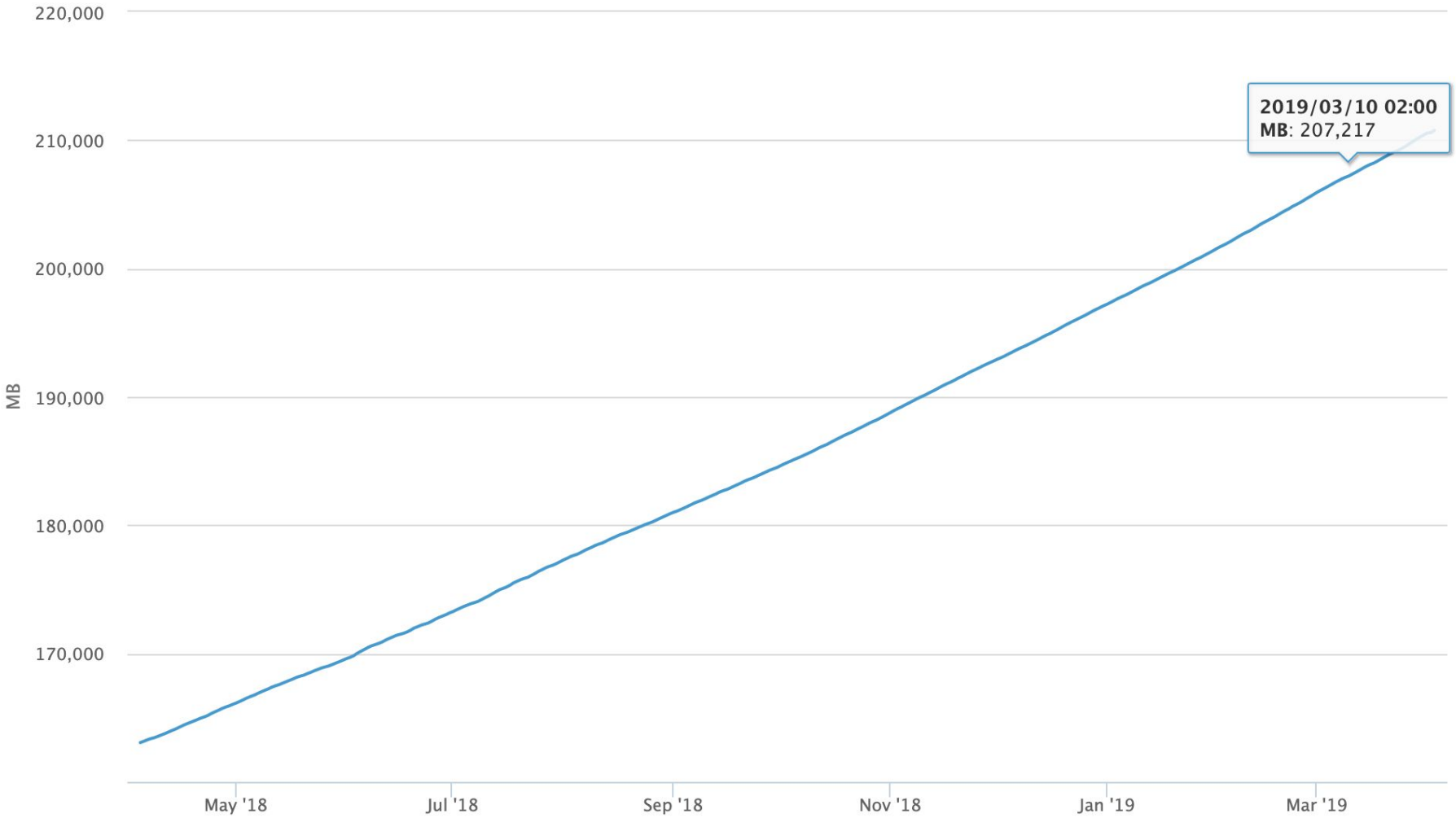
Is it worth it to mine? Probably not...

- WIth **CPU** Core i7: income **$0.04 / year**
- Mε **GPU** NVIDIA GTX590: income **$0.13 / year**
- Mε **GPU** ATI 6990: **income $0.58 / year**
- With specialized hardware AntMiner S5+:
  - Income: $5136 / year
- Electricity cost: $2409 / year (with $0.08 / kWh)
- Pool participation cost: $102.74
- Profit: $2624 / year
- Initial hardware cost: $2307
- **Profits the first year: $317**

Wallets

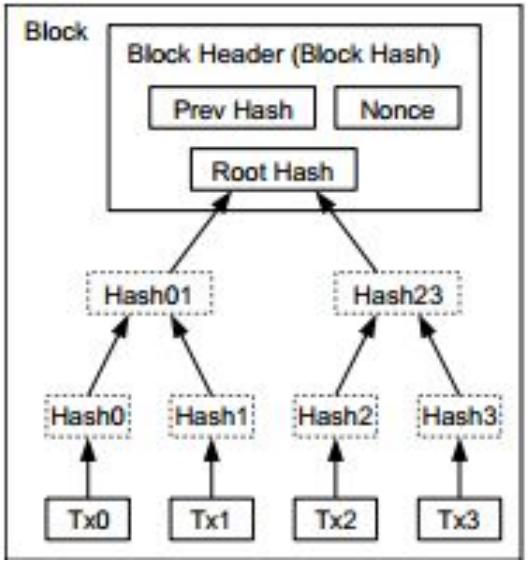
Full nodes

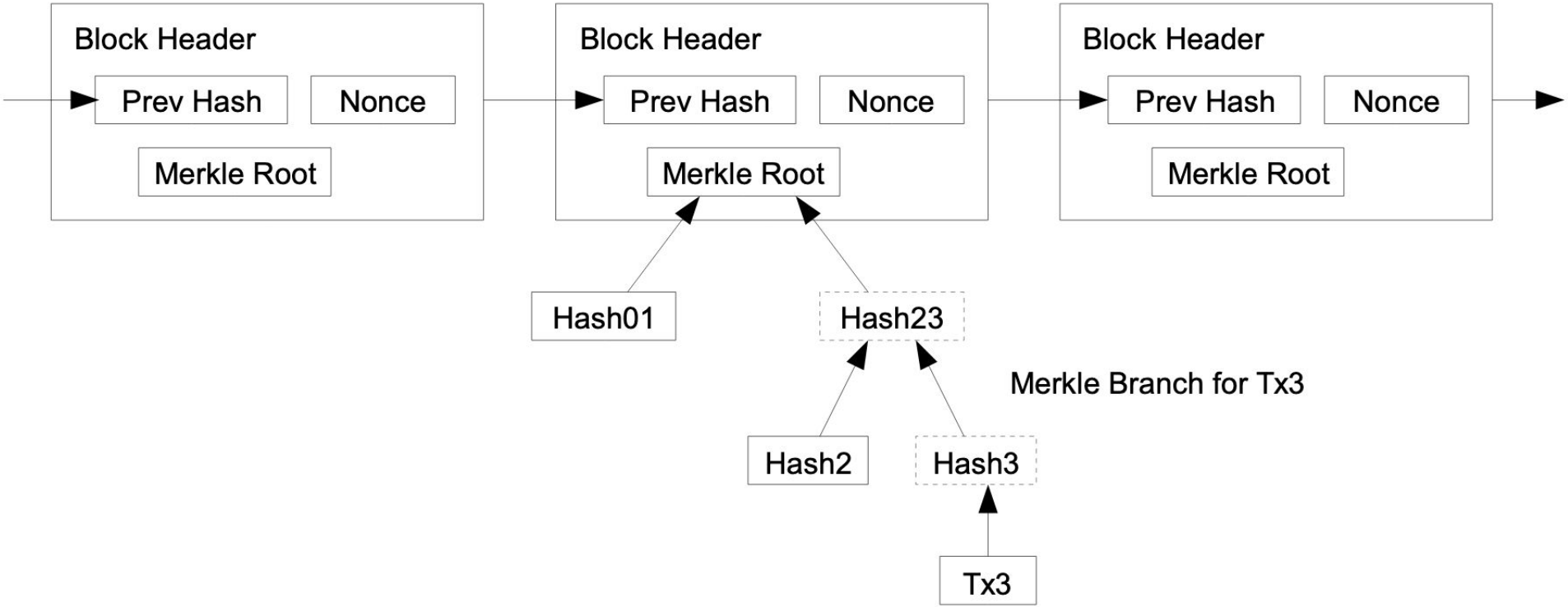
- Some wallets maintain the whole blockchain
- Full nodes:
  - Keep the whole blockchain history (~187 GB)
  - Keep the whole UTXO
  - Verify each tx
  - Verify each block
  - Relay every tx and block


220,000
210,000
200,000
190,000
180,000
170,000
MB
2019/03/10 02:00
MB: 207,217
May '18
Jul '18
Sep '18
Nov '18
Jan '19
Mar '19

Merkle trees of transactions

- Transactions not yet confirmed, but received by a full node are collected into a data structure called the **mempool**
- To build a block, the mempool transactions **are collected** into a Merkle Tree in an (arbitrary, but valid) order defined by the miner
- The application data in the **block header** for which the proof-of-work equation is solved only contain **the root** of this Merkle Tree:
  x is the Merkle Tree Root!




Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Block Header
Prev Hash
Nonce
Merkle Root
Hash01
Hash23
Merkle Branch for Tx3
Hash2
Hash3
Tx3

Advantages of using a Merkle tree

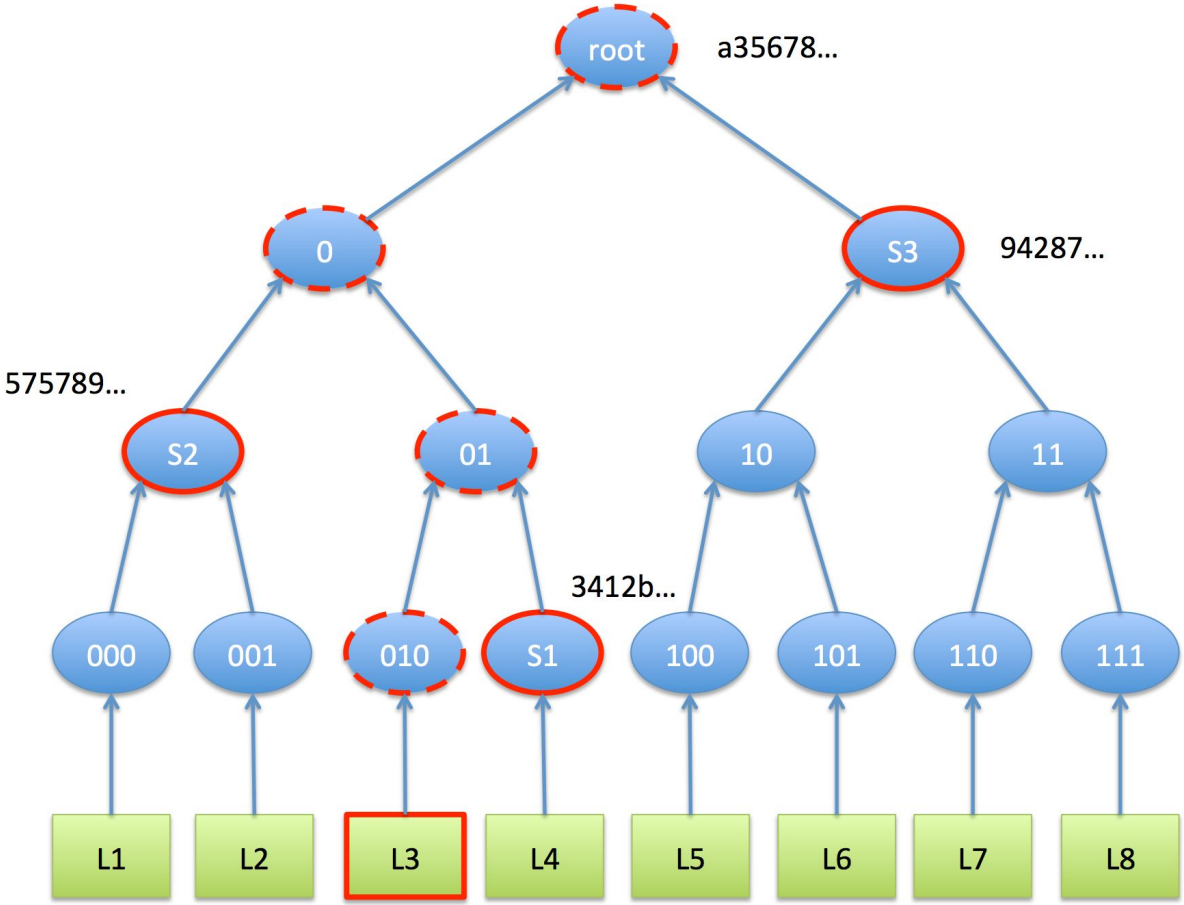
- Proof-of-work difficulty **does not depend** on the number of confirmed transactions
  - Each miner is incentivized to include all transactions they can and have a non-zero fee
- The PoW difficulty **only depends on the target T**
  - This allows better control of the mining rate
- **It enables SPV wallets!**

SPV

- Simple Payment Verification
- A different type of wallet
- Useful for mobile, laptops etc.
- Doesn’t need to download the whole blockchain
  - Does not download all transactions
- Much faster
- Keeps **only the block headers from genesis till today (C)**
- Connects to multiple **untrusted** servers
- Server is a full node which **proves** to the SPV wallet each claim

SPV

- The SPV wallet sends to the SPV server the bitcoin addresses they have
  - **But not the private keys!**
  - The SPV server knows which transactions to send to the SPV client
  - The bitcoin addresses are shared in the form of a Bloom filter
- The SPV wallet verifies each block's **PoW** and authenticated ancestry
  - Keeps a longest chain as usual -- but without transactions
- The SPV wallet verifies **each tx** it receives
  - Sigs
  - Law of conservation
- The SPV wallet verifies that the tx belongs to the Merkle Tree root of a block


root
a35678...
0
S3
94287...
575789...
S2
01
10
11
3412b...
000
001
010
S1
100
101
110
111
L1
L2
L3
L4
L5
L6
L7
L8

SPV Security

- SPV wallets
  - **don’t keep** a UTXO
  - **don’t verify or receive** txs they are not interested in
  - **don’t verify** coinbase validity
- Have the **same** level of security as a regular full node (under the honest majority assumption)
- What can a malicious SPV server achieve?
  - Temporary fork to invalid block (invalid coinbase, txs, non-existing UTXO, double spending...)

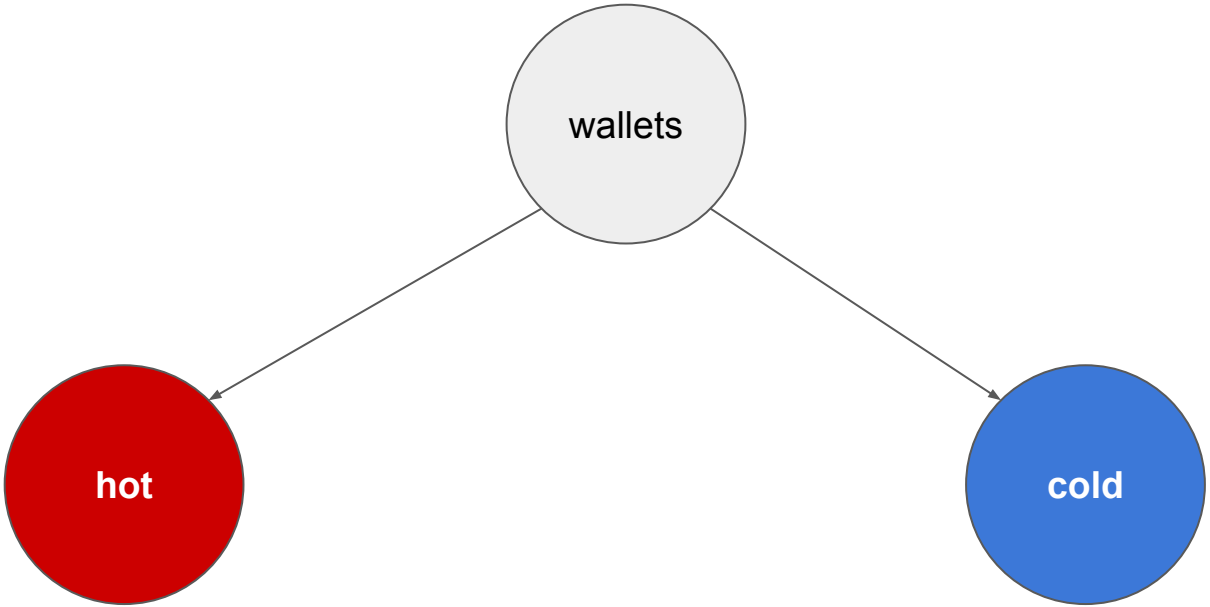
Wallet seeds and HD wallets

- An infinite sequence of wallet private keys can be generated from a single “master private key” (BIP 0032) -- an **HD wallet**
- A private key can be encoded in the form of a human-readable *seed*
- This seed is sufficient to recover *all* the private keys of a wallet
- Typically backed up on paper and optionally encrypted with password

Example:

```
deal smooth awful edit virtual monitor term sign start home shrimp
wrestle
```

Wallet classification



Hot and cold wallets

- I can have my keys on an Internet-connected computer
  - “**Hot** wallet”
  - Easy to use
  - I can always spend my money immediately
- I can keep my private keys offline
  - “**Cold** wallet”
  - Kept on a computer not connected to the Internet or a hard drive
  - My keys cannot easily be stolen
  - I can move my keys to a hot wallet when I need to spend it
  - I can see how much money I have using my public keys which can be kept safely online!

Other ways to store cold wallets

Paper wallet

- Private key is printed on a piece of paper
- Can be kept in a physical safe or a real bank vault
- Can optionally be encrypted with a secret password (which is remembered)

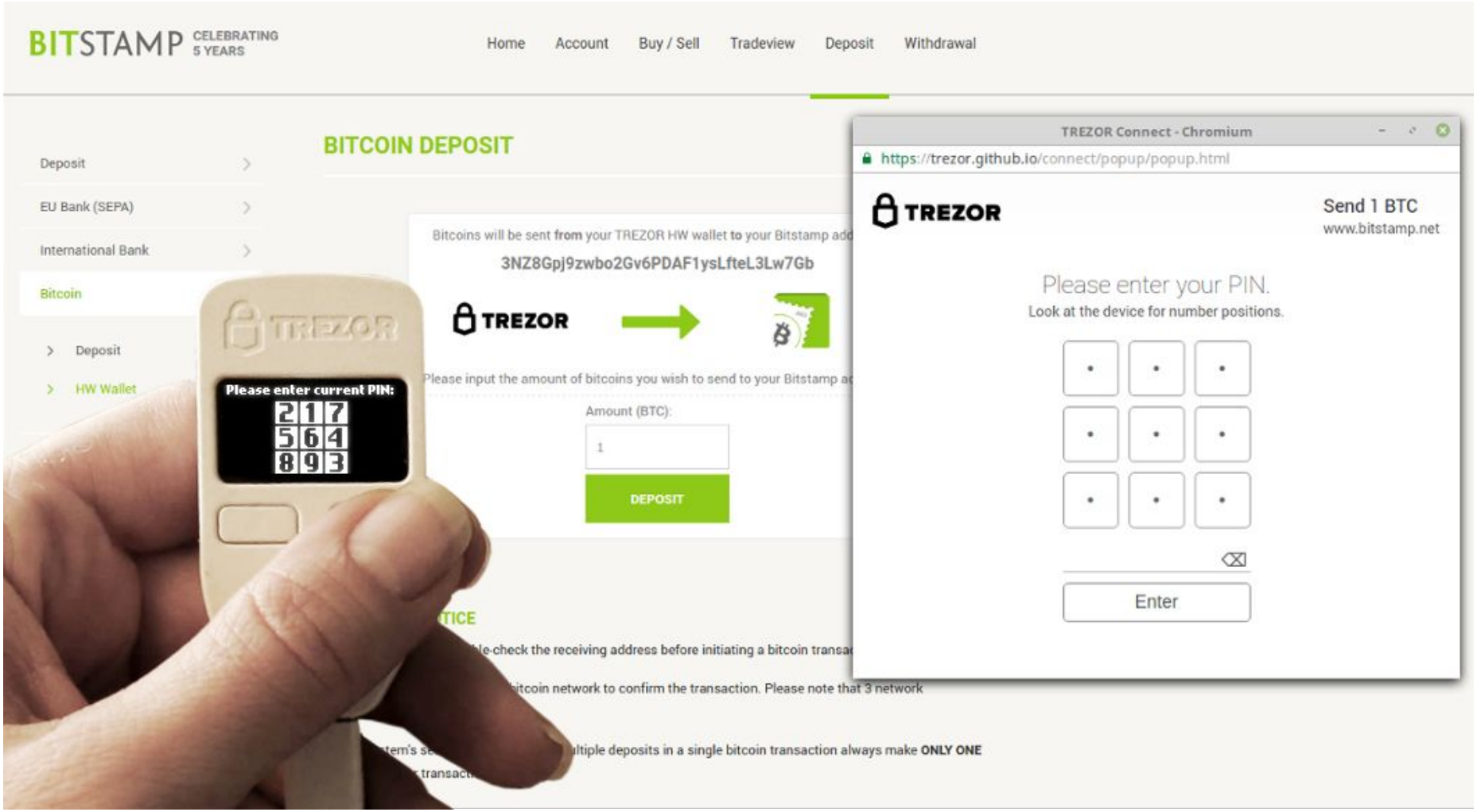
Brain wallet

- Private key is literally SHA256(“my dog’s name is Barbie”)
  or some other passphrase
- Full private key can be recovered by memory
- Extremely unsafe! (More than $100,000 stolen due to low entropy passwords)

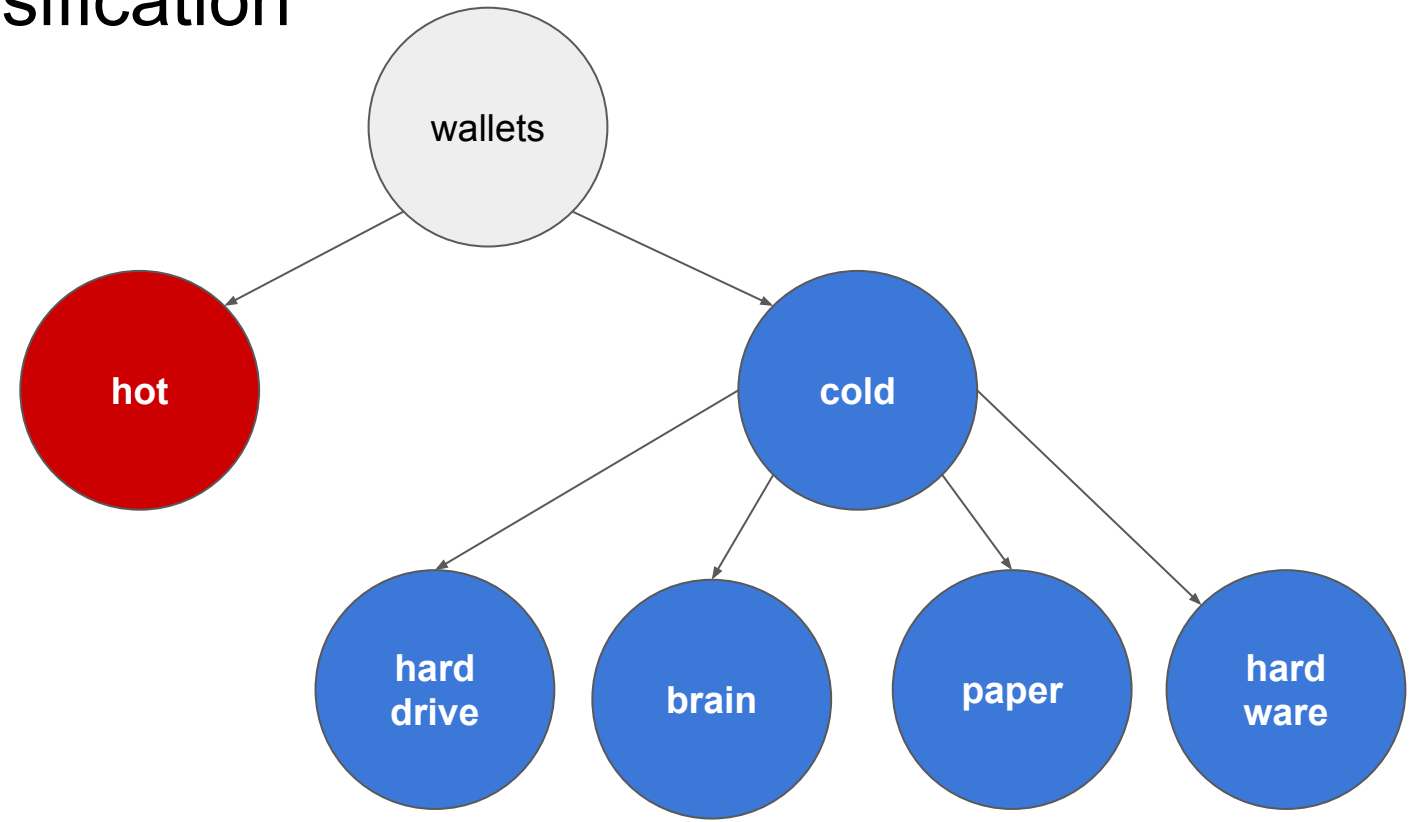
Hardware wallets

- Special hardware device used to store private keys
- Cold wallet
- Most popular ones: **Trezor** and **Ledger**
- Connects to a computer via USB
- Keys never leave the device
- Device produces signed transaction, sends transaction to computer
- Addresses you sent to are verified by looking at the screen
- As hardware/software is specialized, much harder to “hack” or have bugs
- Works safely even if host computer is compromised
- Protected by a pin in case of theft
- Can be backed up into paper and/or other hardware wallets


TREZOR
Confirm sending
0.0469 BTC
to
1Nuu2753n7h3ZnCQJ
CT2HVKTFfQjhpXhcw
Confirm
Cancel


BITSTAMP CELEBRATING 5 YEARS
Home
Account
Buy / Sell
Tradeview
Deposit
Withdrawal
BITCOIN DEPOSIT
Deposit
EU Bank (SEPA)
International Bank
Bitcoin
Deposit
HW Wallet
Bitcoins will be sent from your TREZOR HW wallet to your Bitstamp add
3NZ8Gpj9zwbo2Gv6PDAF1ysLfteL3Lw7Gb
TREZOR
Please input the amount of bitcoins you wish to send to your Bitstamp a
Amount (BTC):
1
DEPOSIT
TICE
le-check the receiving address before initiating a bitcoin transa
itcoin network to confirm the transaction. Please note that 3 network
ltiple deposits in a single bitcoin transaction always make ONLY ONE
TREZOR
Please enter current PIN:
2 1 7
5 6 4
8 9 3
TREZOR Connect - Chromium
https://trezor.github.io/connect/popup/popup.html
TREZOR
Send 1 BTC
www.bitstamp.net
Please enter your PIN.
Look at the device for number positions.
Enter

Wallet classification



Blockchain Incentives

Mining Games

- As we have seen: miners are incentivised via rewards to follow the protocol.
- Does this ensure that the protocol will be executed?
  - Is the reward mechanism “incentive compatible”?
  -
  - Protocol is **dominant strategy**: a party will fare best by following the protocol.
  - Protocol is **Nash equilibrium**: if all parties follow the protocol, you cannot do better by deviating.
  -

Dominant Strategies: example

Participants want to maximise payoff

Split or Steal Game:

	Split	Steal
Split	50/50	0/100
Steal	100/0	1/1

Here stealing is the dominant strategy!

Not necessarily the best outcome...

See prisoner's dilemma
And the golden balls TV game

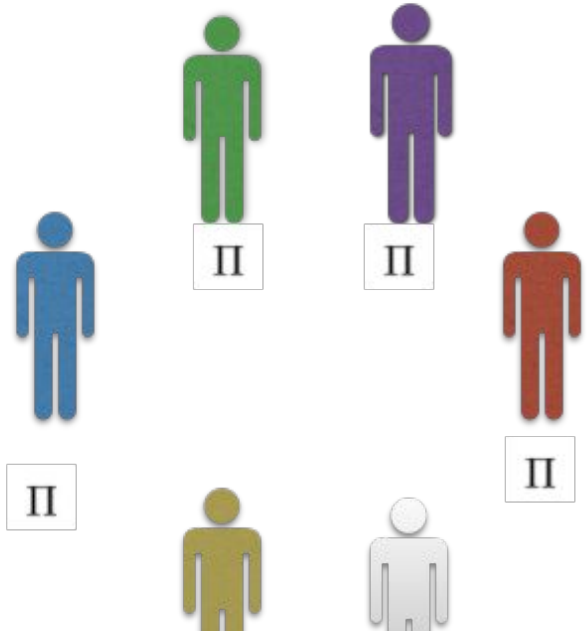
https://en.wikipedia.org/wiki/Prisoner%27s_dilemma

Protocol is Nash Equilibrium, I

- All the participants are rational and want to maximize the utility they obtain at the end of the execution.
- Utility of a participant is a function that takes as input the strategies of all the participants and has as output a real number that represents the gains of this participant at the end of the execution.

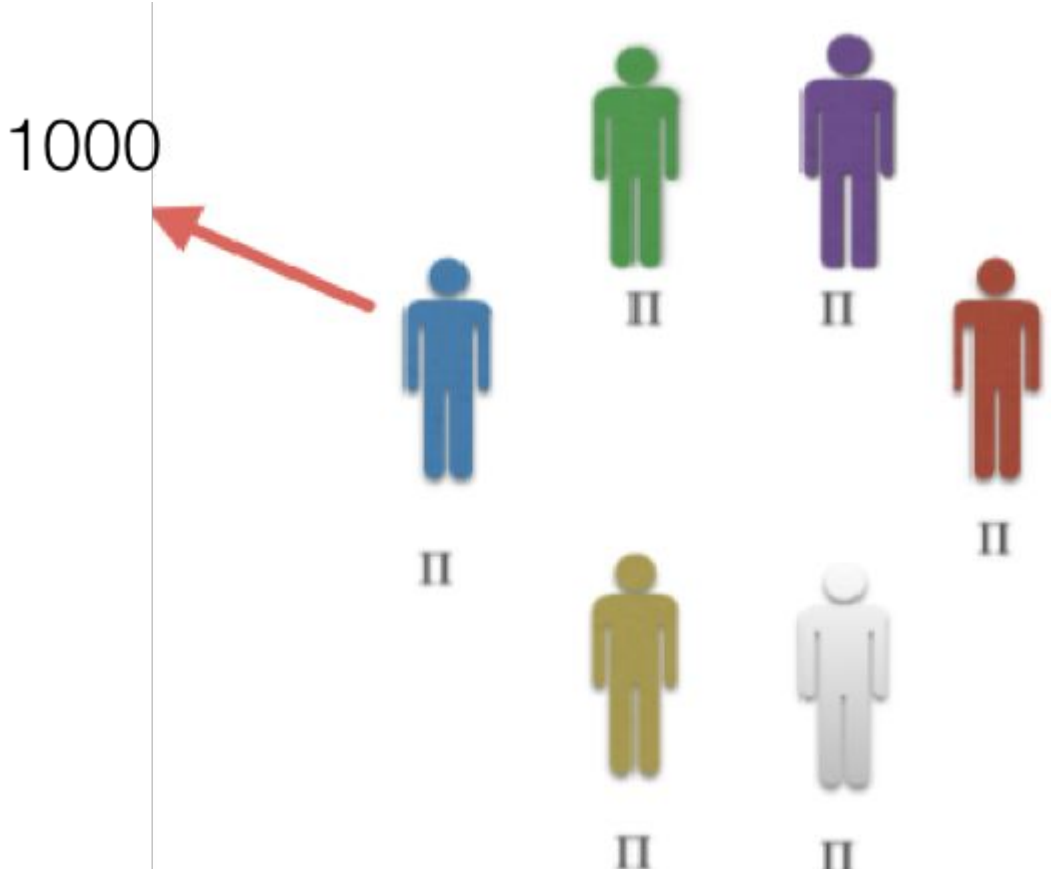
Protocol is Nash Equilibrium, II

All the participants are rational: they want to maximize their utility.

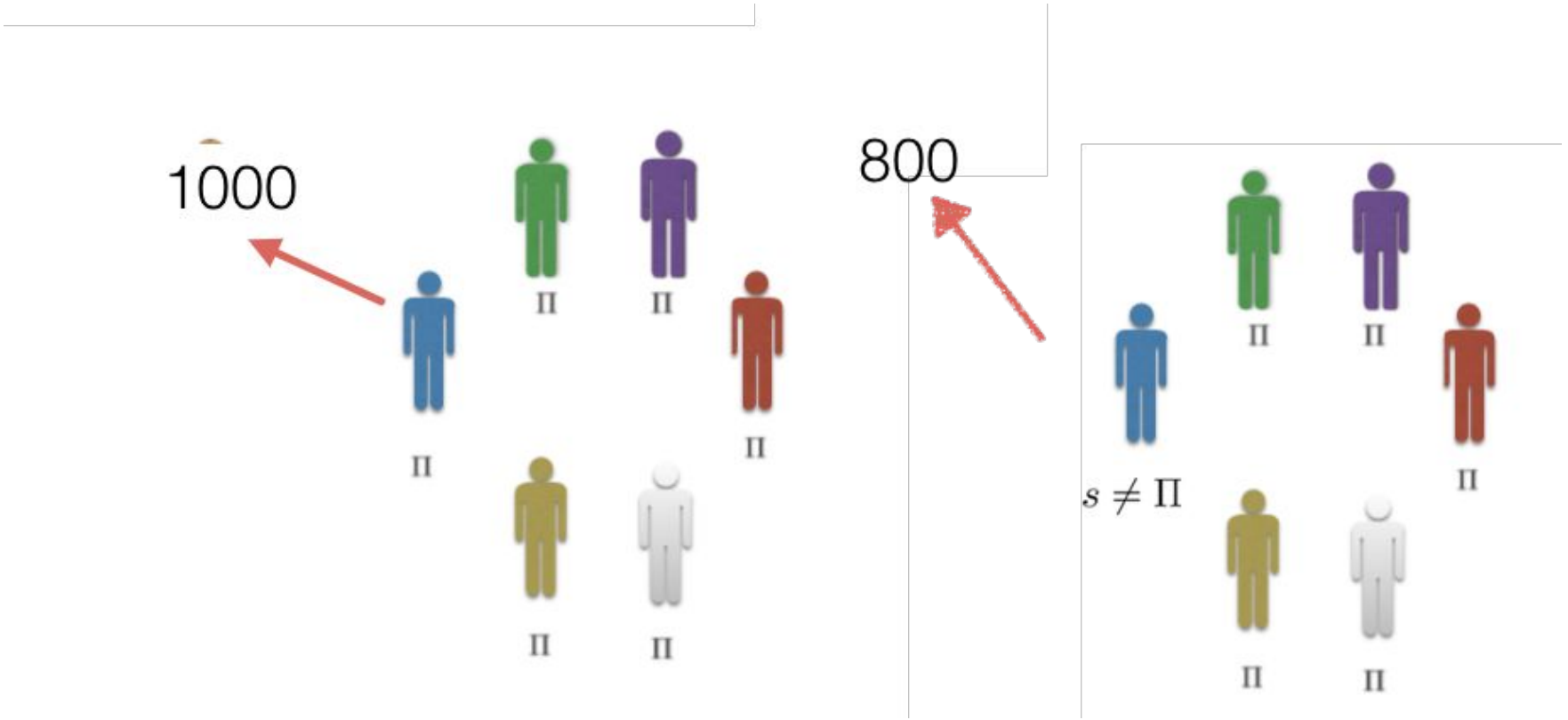


Protocol Π is Nash equilibrium

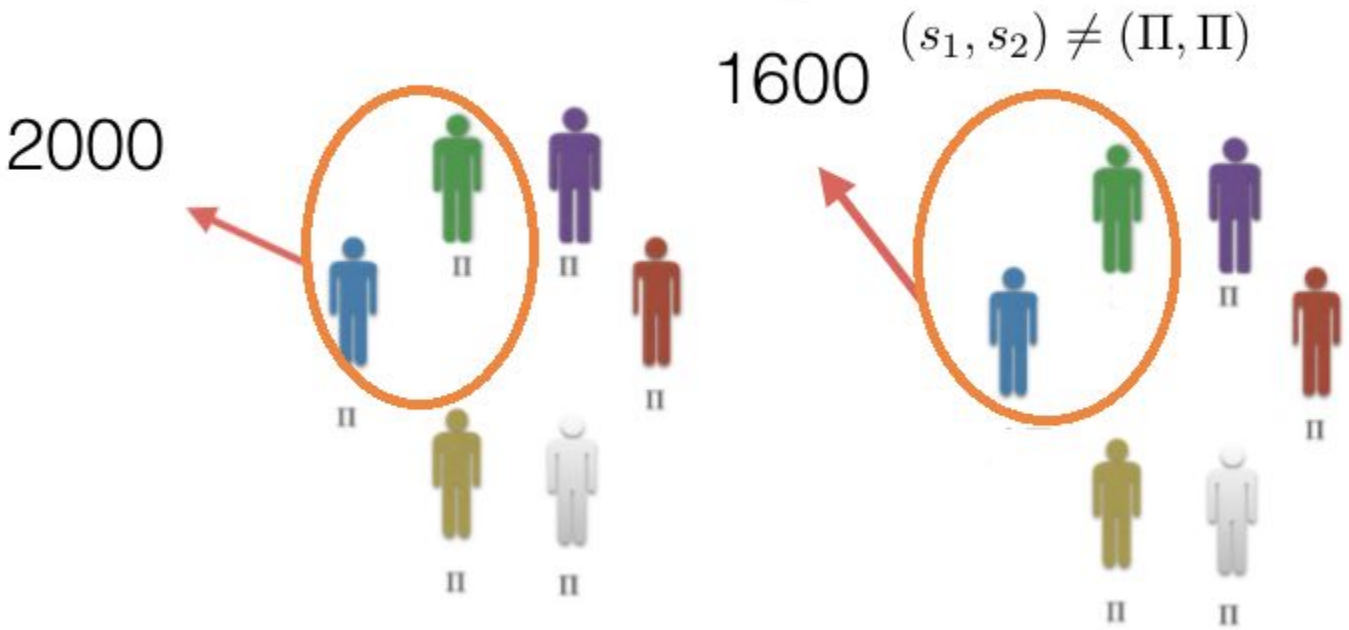
Protocol is Nash Equilibrium, III



Protocol is Nash Equilibrium, IV



Generalisation to Coalitions

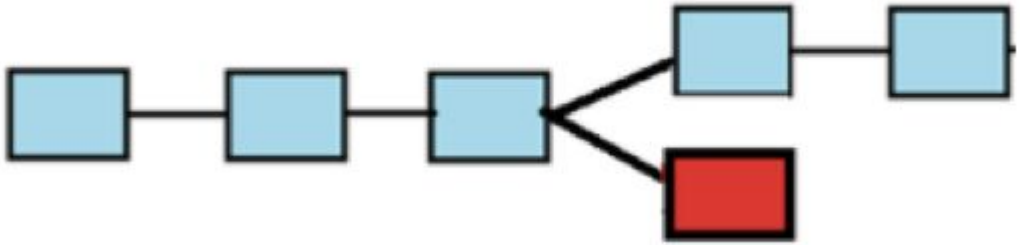


Is Bitcoin a Nash Equilibrium ?

- What could be the utility in Bitcoin?
- How could utility be defined in a probabilistic protocol?

Absolute Rewards, I

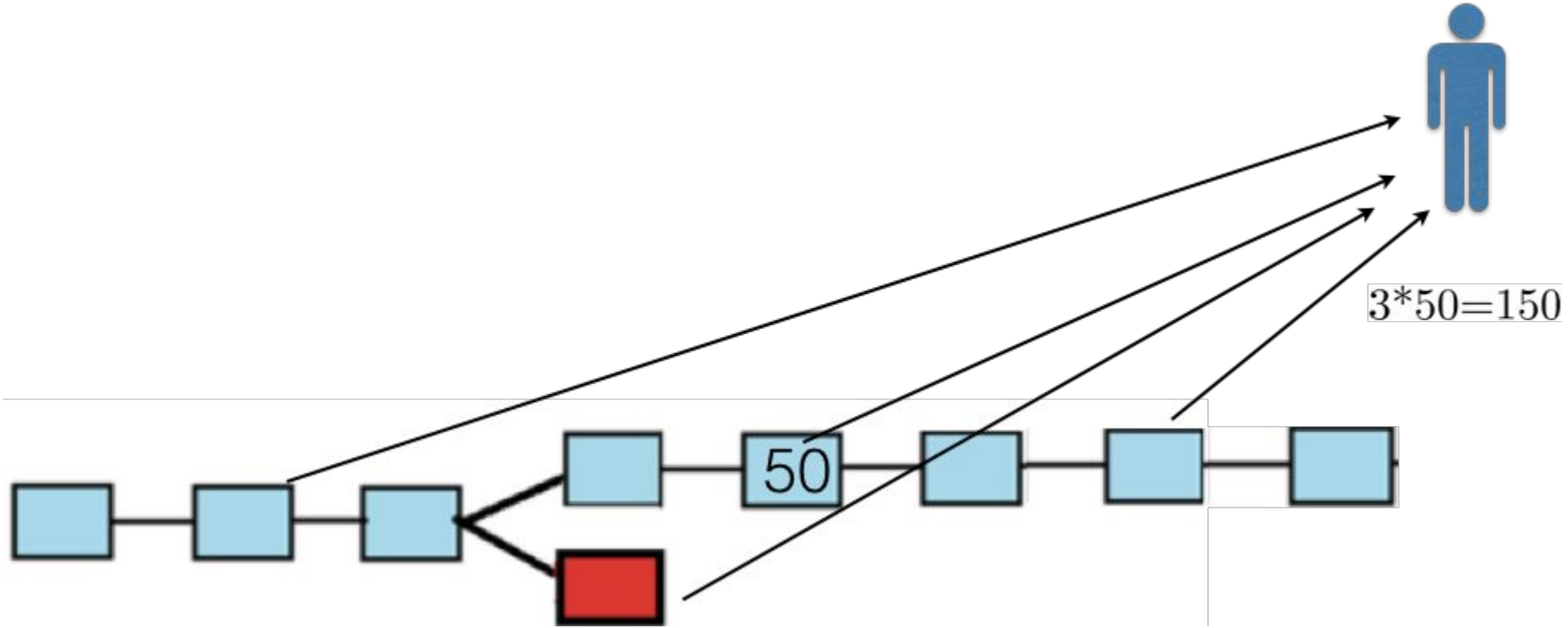
- If we specify the algorithms of all the participants and the outcome of all the randomness used by participants in a finite execution of the Bitcoin protocol then we have a unique outcome.



Absolute Rewards, II

- Each block of the adopted chain gives a reward to its producer.
- **Absolute rewards utility**. The utility of a coalition is equal to the number of BTC that it has obtained at the end of the execution.

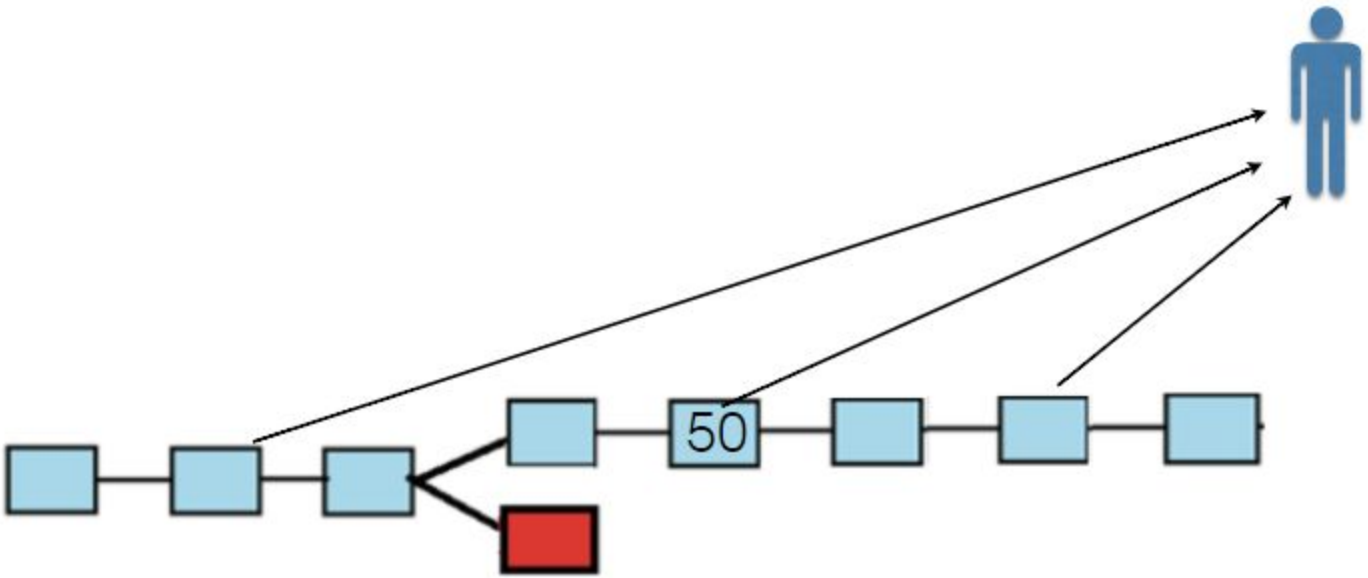
Absolute Rewards, III



Relative Rewards, I

- **Relative Rewards**. The utility of a coalition in Bitcoin is equal to relative rewards when it wants to maximize the amount of BTC that it earns divided by the total amount of BTC that all the participants receive at the end of the execution.

Relative Rewards, II



$$\frac{3 * 50}{8 * 50}$$

Utility in probabilistic protocols

- Given the strategies of all the participants, the outcome of the Bitcoin execution is a random variable. So the utility of a coalition is also a random variable. How to resolve this?
  - Via Expectation: will determine the expected value of utility.
  - Via events that happen with high probability.

Bitcoin and Equilibria, I

- Kroll et al. in ``The economics of Bitcoin mining, or Bitcoin in the presence of adversaries" (2013) show that a certain modeling of the Bitcoin protocol is a Nash equilibrium w.r.t. absolute rewards.

If utility is equal to the expected value of absolute rewards and block difficulty is fixed, then the expected number of blocks is proportional to mining power and this is what is delivered by a Bitcoin execution

Bitcoin and Equilibria, II

- Eyal and Sirer in `` Majority is not enough: Bitcoin mining is vulnerable” (2014) show that Bitcoin is susceptible to a type of attack called selfish mining and the protocol is not a Nash equilibrium w.r.t. relative rewards.

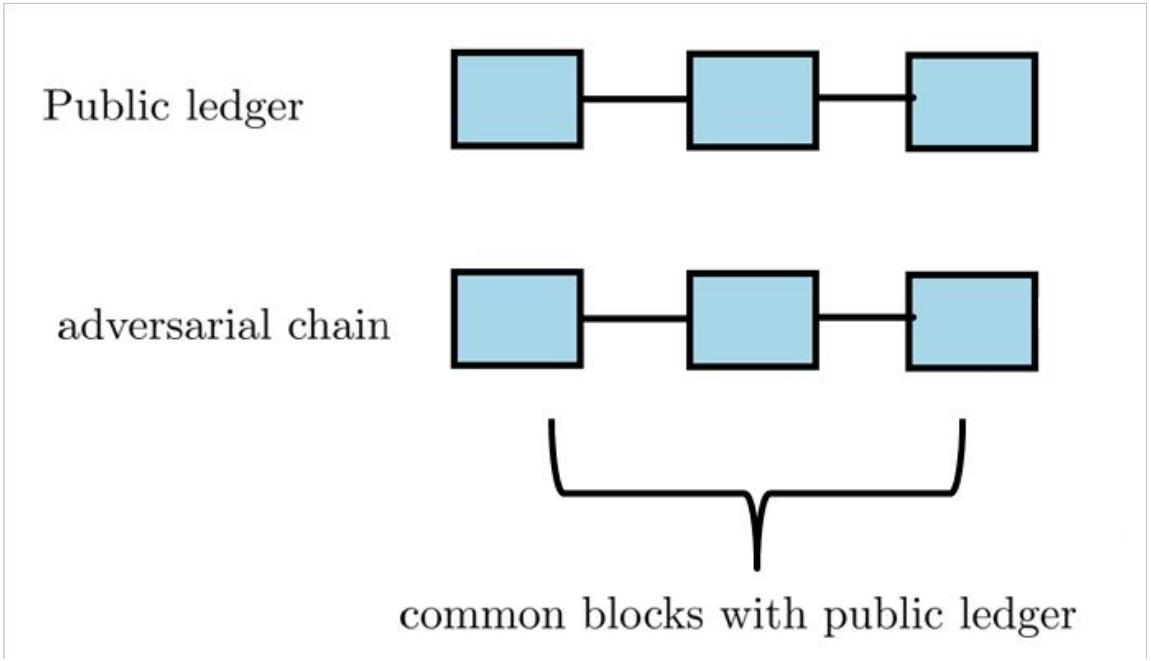
Selfish Mining, I

- A strategy that enables a coalition to collect more expected relative rewards by deviating from the protocol.
  - Attacker maintains a private chain, strategically releasing its blocks to deny honest parties' blocks from being adopted to the "main chain."

Eyal, Ittay, and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable."(2014)

Selfish Mining, II

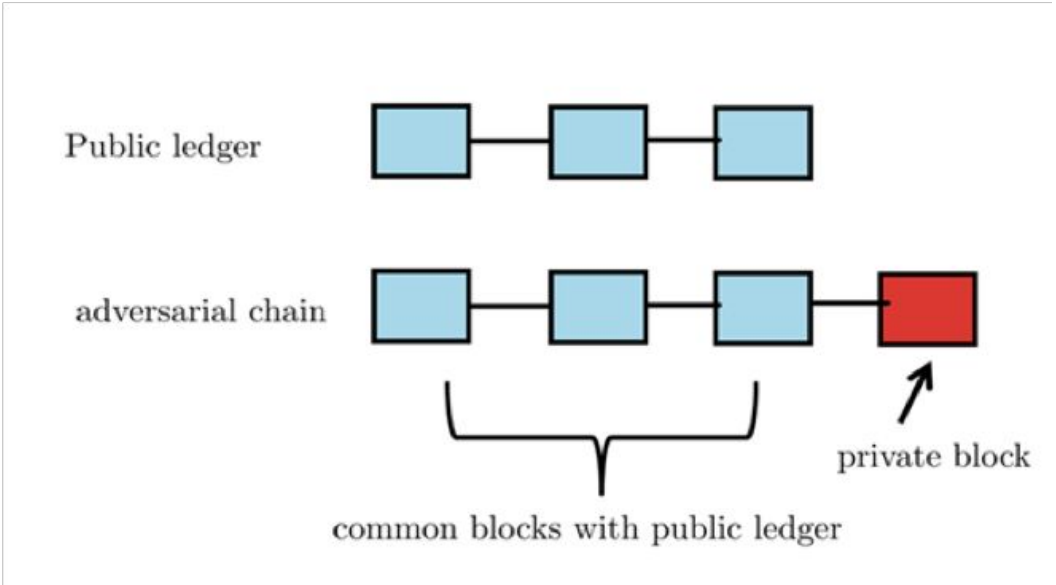
1)The attacker adopts the longest chain and tries to extend it.



Selfish Mining, III

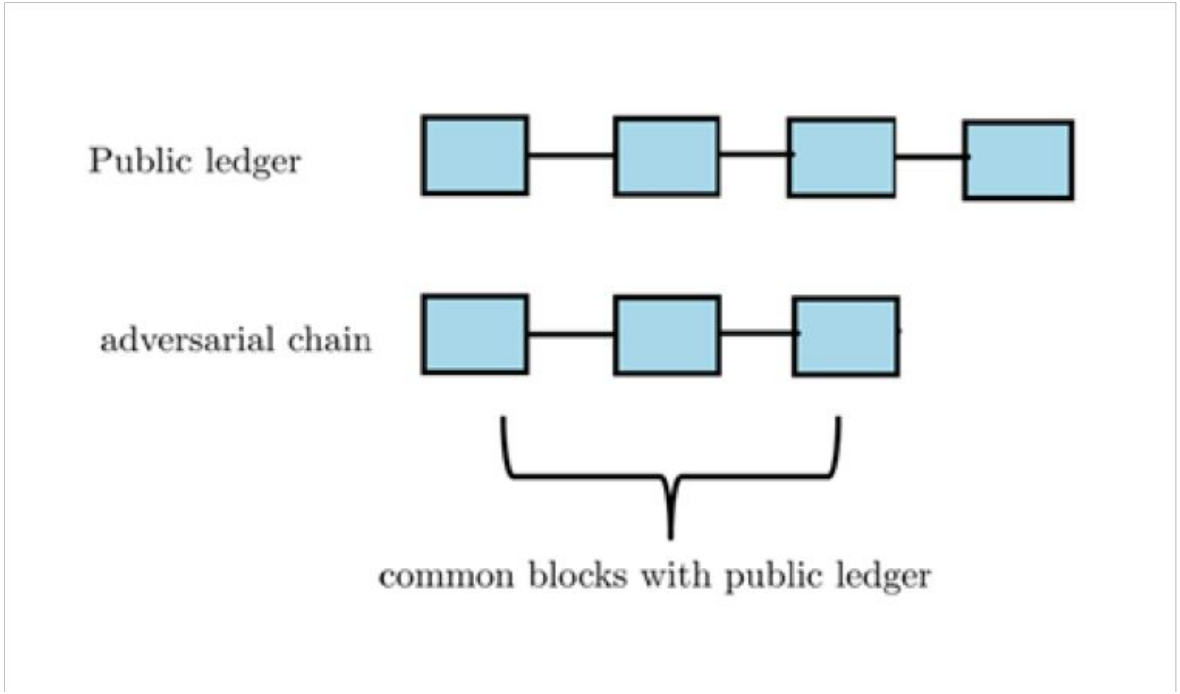
We have the following two cases : 2a or 2b

2a)The attacker is first to produce a block.



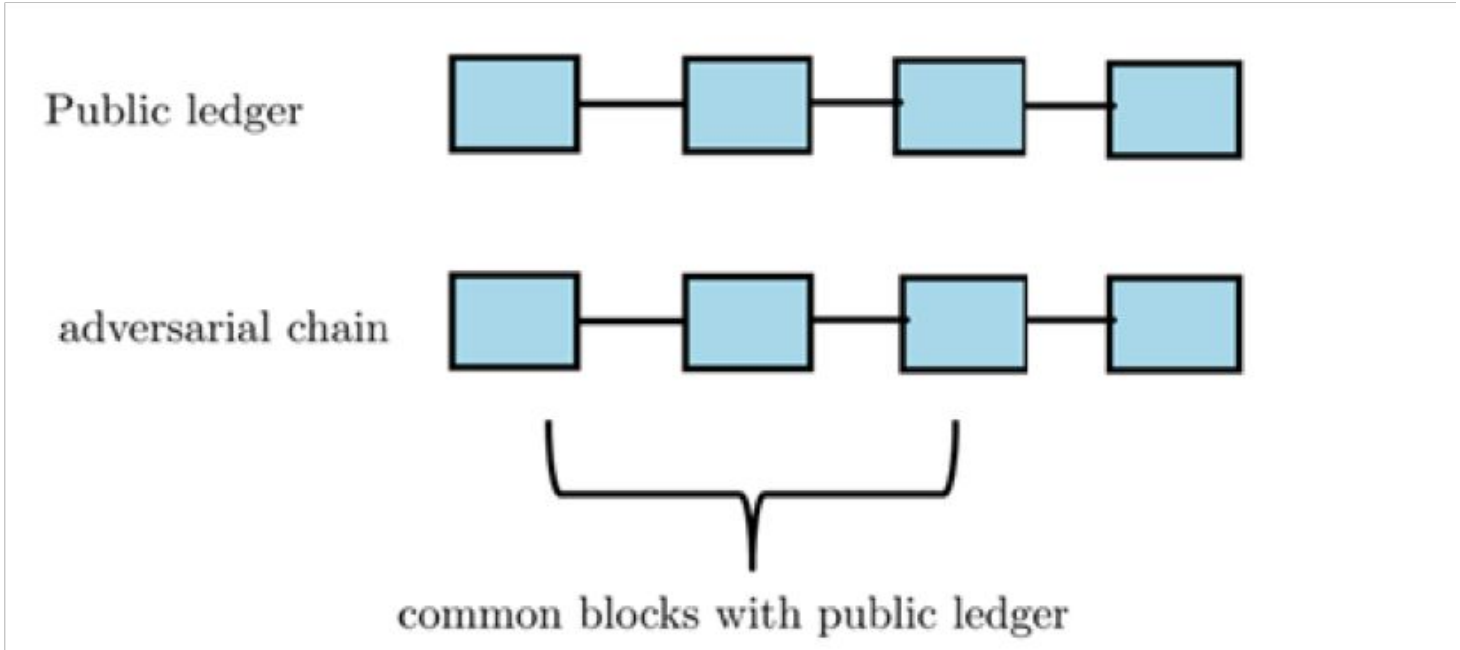
Selfish Mining, IV

2b)The attacker does not manage to produce first a block.



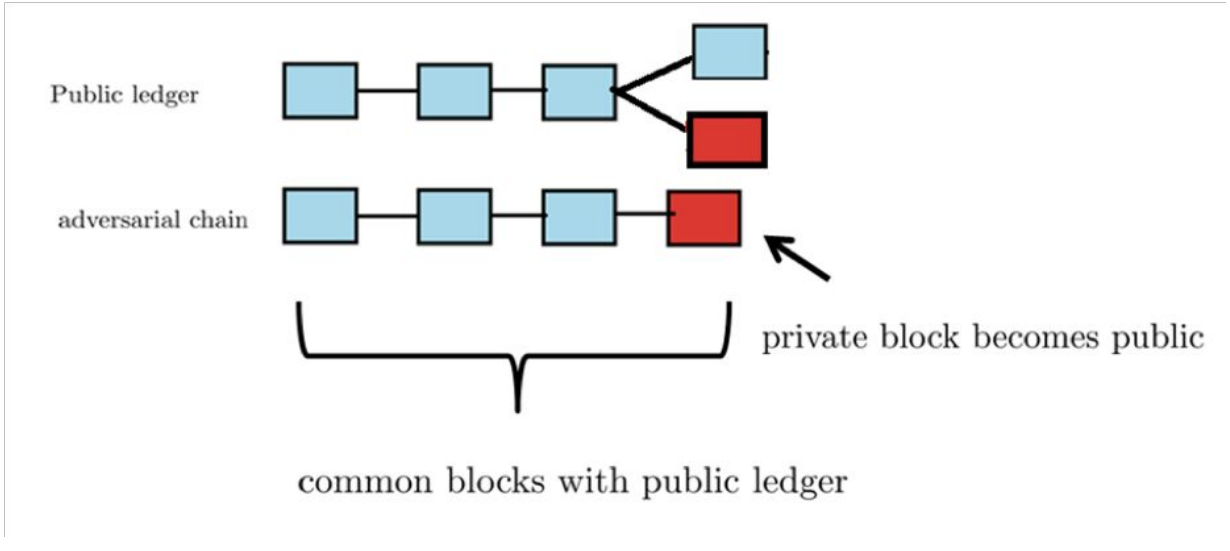
Selfish Mining - case 2a)

The attacker in this case adopts the public ledger.



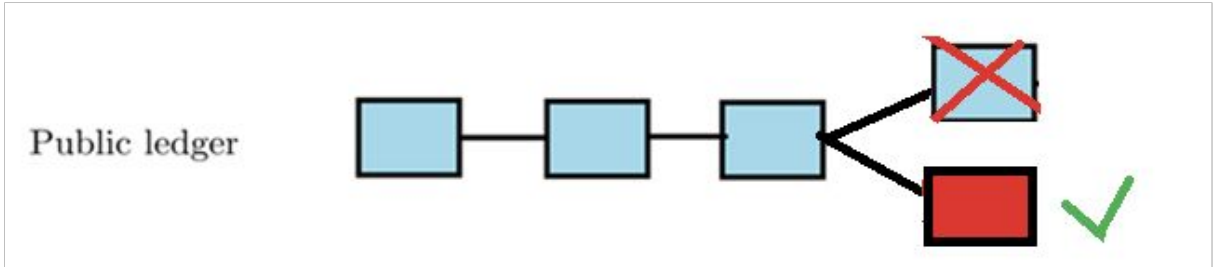
Selfish Mining - case 2a)

The attacker withholds its block(s) private until the public chain comes to be (i) one block behind the private one, (ii) equal in length to the attackers. (choice can depend on “network dominance” of the adversary)



Selfish Mining, Case 2a), II

If the other parties choose to extend the adversarial chain then the adversary has managed to censor legitimate blocks from the public ledger.



Selfish Mining, Case 2a), III

- More generally: the adversary will be capable of censoring blocks, if
  a. the adversary's chain gets two blocks ahead of the public chain.
  b. The adversary manages to deliver first its block to the other parties. When an honest party receives two chains of the same length then it chooses the first that it received.

Selfish Mining, Analysis, I

- The computational power of the attacker contributes only towards censoring blocks and not to extend the public ledger.
- So when it implements the attack, the total number of blocks (expected value) in the public ledger is smaller compared to the total number of blocks in the case where it follows the protocol.
- If the attacker does not manage to deliver its block first, it loses the rewards from this block.

Selfish Mining, Analysis, II

- Consider a process operating in “block rounds." Attacker has probability α to produce the next block. Assume attacker always wins the network race vs. public blocks.
  - Honest play: *n* rounds result to *n* blocks. Attacker owns αn blocks in expectation.
    Utility (Relative Rewards) = α, (Absolute Rewards) = α n.
  - Selfish play: *n* rounds result to (1-α)*n* blocks. Attacker owns αn of those blocks.
    Utility (Relative Rewards) = α/(1-α), (Absolute Rewards) = α n.
- In a static difficulty setting absolute rewards remain the same but relative rewards increase.

Bitcoin and Equilibria, III

- In the changing difficulty setting.
  - Selfish mining will impact chain growth and thus the difficulty recalculation mechanism will lower the difficulty.
  - This means that block production per actual unit of time will increase.
  - As a result, attacker will also receive higher number of (absolute) rewards compared to honest play.

Block Reward Zero Attack

- When the block reward becomes zero the following deviation may arise:
  - When a miner receives two blocks of the same height instead of choosing the first one, it has incentives to choose the block that leaves the most transaction fees unclaimed.
  - A selfish miner can take advantage of this behaviour and create a fork with a block including fewer transaction fees compared to the transaction fees in the head of the public ledger.

Carlsten, Miles, et al. "On the instability of bitcoin without the block reward." (2016)

Bribery Attack

- The attacker creates a fork and includes in the first block a transaction T0 that gives *bribe* money to miners who will adopt the fork and will extend this block.
- The input of T0 is also transmitted in the public ledger and double spends the bribe money.
- If the chain of the attacker does not manage to become longer than the public ledger then the attacker does not lose the bribe money.
    a. In this case, miners who adopted this fork will have spent computational power without gaining anything.

Bonneau, Joseph, et al. "Why buy when you can rent? bribery attacks on Bitcoin consensus." (2016).

Mining Pool Games

- To create a pool or join an existing one?
- Assuming cost of verification and pool maintenance is non-negligible:
  - Optimal solution is a single dictatorial pool.
    - (reason: offset costs with the player that has the lowest service cost).


BY