BLOCKCHAIN AT BERKELEY

BITCOIN IN THE WILD

GAME THEORY & ATTACKS

ANDREW KIRILLOV
ERIKA BADALYAN

Table of Contents

Mining Pools

What are Mining Pools?

Mining pools allow individual miners to combine, or 'pool', their computational power together

- Reduces variance in mining rewards
- Run by **pool managers** or **pool operators**
- Pool manager usually takes a cut of the mining rewards

Mining Shares

Miners in a pool submit **shares** (“near-valid” blocks) to the manager

- Basically just solving PoW with a lower difficulty
- Pool operator pays for valid shares
- Valid blocks are shares as well
  - Individual who finds valid block is not awarded any extra coins

`0x00000000ffff...`

Hash of a valid block

`0x0000ffffffff...`

Hash of a valid share

Pay-per-Share Reward Scheme

Pool pays out **at every share submitted.** Everyone is paid equally, and rewards are proportional pool difficulty.

- More beneficial for **miners**
- Individual miners have no risk from reward variance
  - Pool takes on the risk completely
- Problem: No incentive for individuals to actually submit valid blocks
  - Individuals are paid regardless

Proportional Reward Scheme

Pool pays out **when blocks are found.** Miners are paid proportional to the number of shares they've submitted since the last block.

- More beneficial for the **pool**
- Individual miners still bear some risk in variance proportional to size of the pool
  - Not a problem if pool is sufficiently large
- Lower risk for pool operators - only pay out when reward is found

Expected Reward

- Today's (10/06/20) network hashrate: **~136,461,000 TH/s**
- Antminer S17: **56 TH/s**
- Proportion of network hashrate = (56 TH/s) / (136,461,000 TH/s) => **0.00000041**

Solo mining:

- Expected to mine ~1 in every 2,439,024 blocks => **46.4 yrs**
- 11 halvings will occur in that time => block reward will be **0.00305175781 BTC**
- Assume BTC goes to $100k - still only gets you ~$305 in 46.4 yrs => **$6.57/yr**

Pool mining:

- Assume pool has ⅙ network hashrate => mines 1 in every 6 blocks => **1 hr**
- Current block reward is **6.25 BTC**
- Assume BTC at $10k => 8,760 hrs/yr * $0.15375 (your share of pool rewards) = **$1346.85/yr**

AUTHOR: MAX FANG
UPDATED: BRIAN HO

Pros & Cons

+ Pros

01 Allows individual miners to participate

- Cons

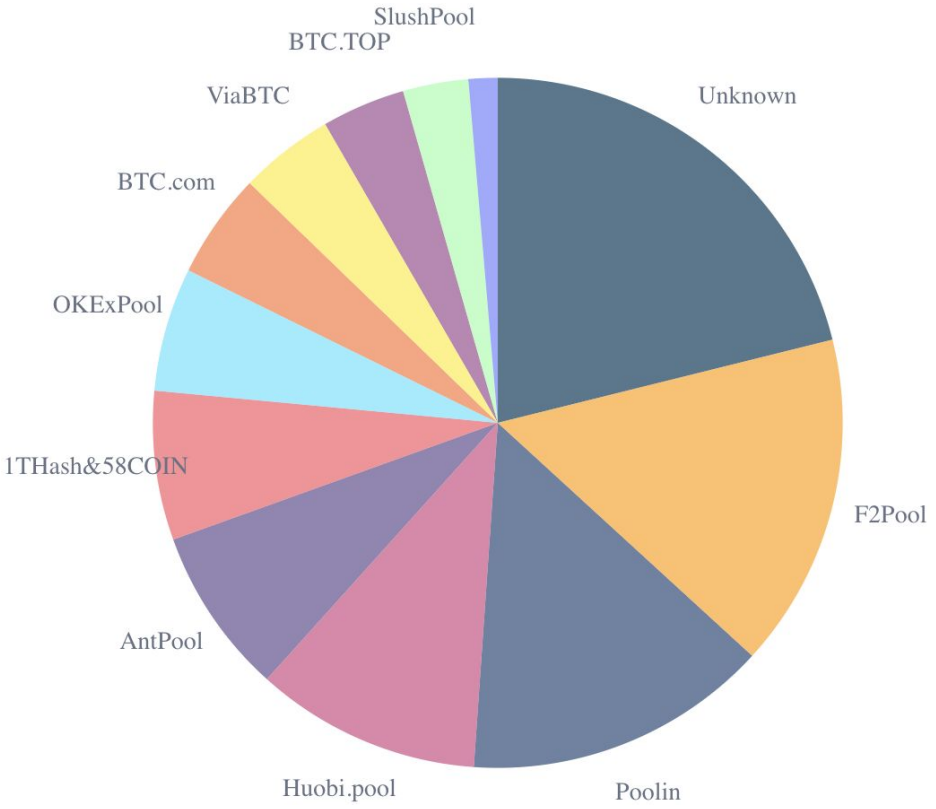
01 Pool manager must be trusted

02 Enables a multitude of attacks

AUTHOR: MAX FANG

Centralization

Source:
blockchain.info (10/7/19)



Community exhibits backlash against large mining pools

- Ex: GHash.io in 2014

Single entity might be be participating in multiple pools

- Called "**Laundering hashes**"
- Actual concentration of control over mining hardware **is unknown**

AUTHOR: MAX FANG

Incentive Alignment

Is it possible that mining pools are vulnerable to some incentive misalignment?
Can miners take advantage of the differences in these two types of payout schemes?

Incentive Alignment

Is it possible that mining pools are vulnerable to some incentive misalignment?
Can miners take advantage of the differences in these two types of payout schemes?

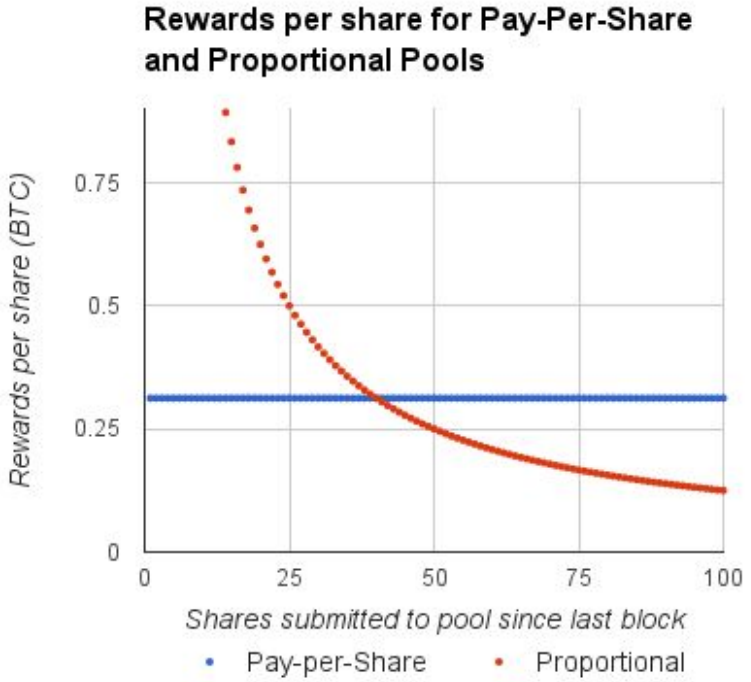
YES!

Pool Hopping

Rewards per share in a **proportional** pool are inversely related to the number of shares submitted while mining the current block.

Rewards per share in a **pay-per-share** pool are constant no matter how many shares submitted.

How does a miner maximize their profit?



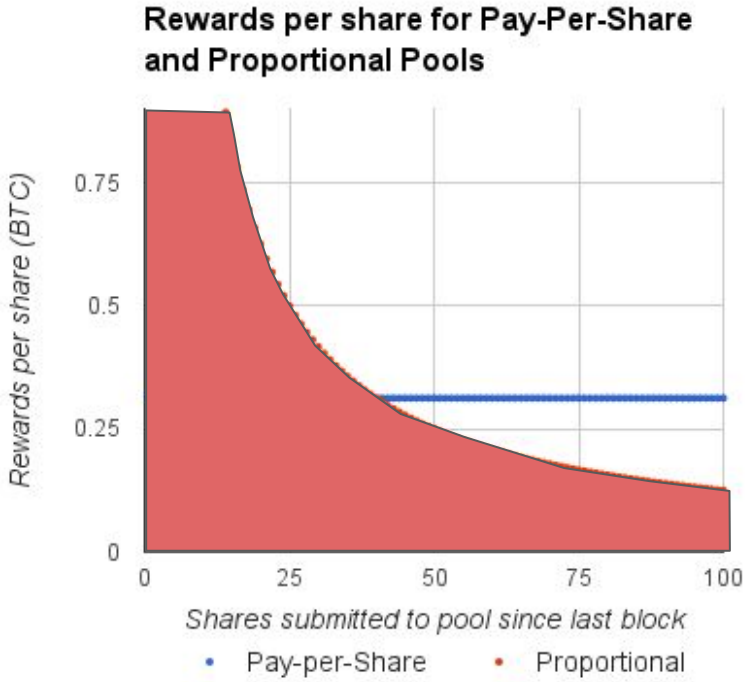
AUTHOR: RUSTIE LIN

Pool Hopping

Rewards per share in a **proportional** pool are inversely related to the number of shares submitted while mining the current block.

Rewards per share in a **pay-per-share** pool are constant no matter how many shares submitted.

How does a miner maximize their profit?



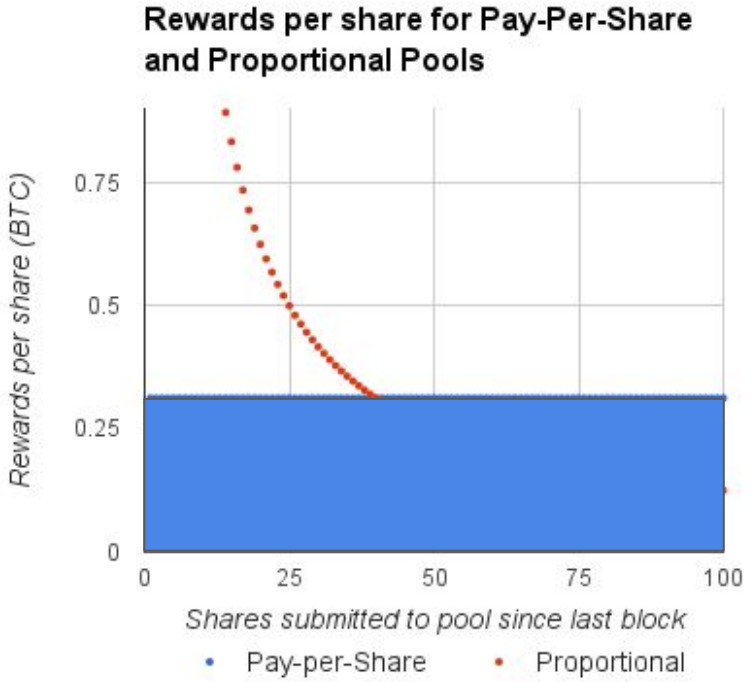
AUTHOR: RUSTIE LIN

Pool Hopping

Rewards per share in a **proportional** pool are inversely related to the number of shares submitted while mining the current block.

Rewards per share in a **pay-per-share** pool are constant no matter how many shares submitted.

How does a miner maximize their profit?



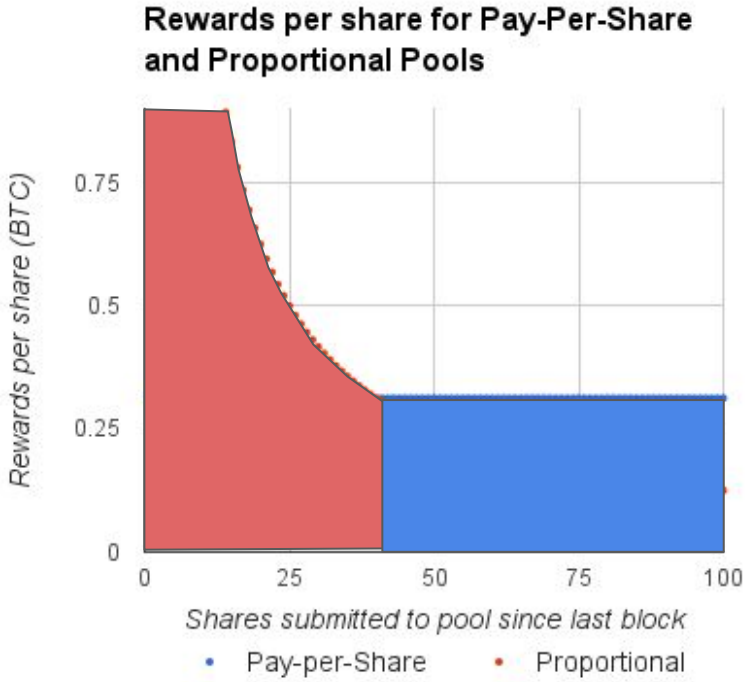
AUTHOR: RUSTIE LIN

Pool Hopping

Rewards per share in a **proportional** pool are inversely related to the number of shares submitted while mining the current block.

Rewards per share in a **pay-per-share** pool are constant no matter how many shares submitted.

How does a miner maximize their profit?

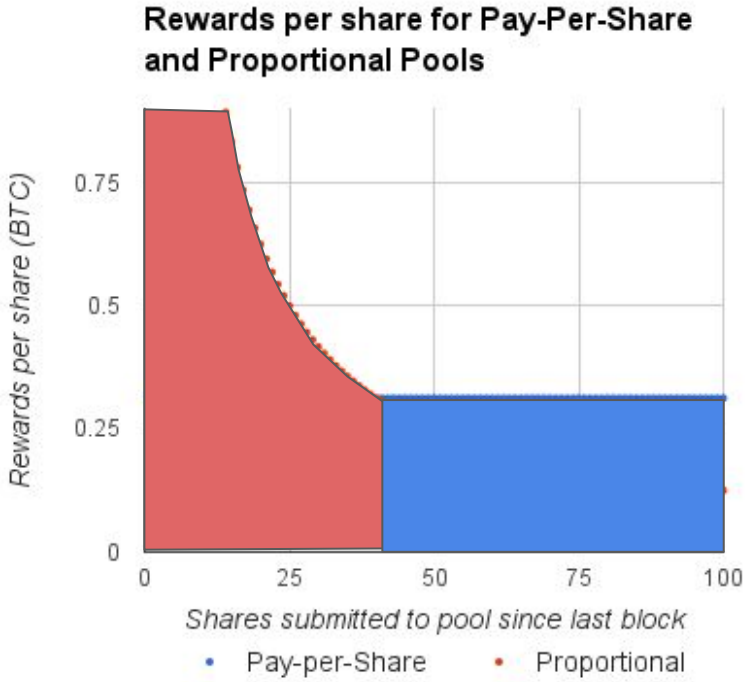


AUTHOR: RUSTIE LIN

Pool Hopping

Therefore, proportional pools are **not feasible in practice**

Designing a mining pool reward scheme with aligned incentives that is not vulnerable to pool hopping remains an **open problem**



AUTHOR: RUSTIE LIN

Pool Cannibalization

Cannibalizing Pools - Distribute some small % of mining power equally among all other pools, withhold valid blocks.

- Rewards will still be received
- Undetectable unless statistically significant

AUTHOR: MAX FANG
(Originally presented by Martin Koppelmann at SF Bitcoin Devs Seminar)

Pool Cannibalization Breakdown

Givens:

- We have 30 H/s. Total network is 100 H/s.
- Assume 1 BTC block reward. All of the following numbers are **expected value.**
- 30% HR (hashrate)
  - = 30% MR (Mining Reward)
  - = 0.3 BTC

Let's say we buy more mining equipment, worth 1 H/s

AUTHOR: MAX FANG
(Originally presented by Martin Koppelmann at SF Bitcoin Devs Seminar)

Pool Cannibalization Breakdown

Standard mining strategy:
Add 1 H/s to your own rig

Your hashrate breakdown

- 31/101 of network H/s = 30.69% HR
- Reward: 0.3069 * 1 BTC = 0.3069 BTC
- Revenue gain = **0.0069 BTC for 1% increase** in network hashrate

AUTHOR: MAX FANG
(Originally presented by Martin Koppelmann at SF Bitcoin Devs Seminar)

Pool Cannibalization Breakdown

Pool cannibalizing strategy:
Distribute 1 H/s among all other pools, **don't submit valid blocks**

Rest of network hashrate breakdown:

- 70/71 honest H/s, 1/71 dishonest H/s
  - 70% **effective** hashrate = 0.7 BTC
- You own 1/71 of other pools => reward = (1/71) * 0.7 BTC = 0.0098 BTC
- Revenue gain = **0.0098 BTC for 1% increase** in network hash rate

- **42% larger** (expected) revenue gain from pool cannibalization

AUTHOR: MAX FANG
(Originally presented by Martin Koppelmann at SF Bitcoin Devs Seminar)

Pool Wars

- Attack decisions resemble an iterative game
  - Two players: Pool 1 and Pool 2
  - At each turn, the player chooses whether or not to attack the other
- Each iteration of the game is a case of the Prisoner's Dilemma

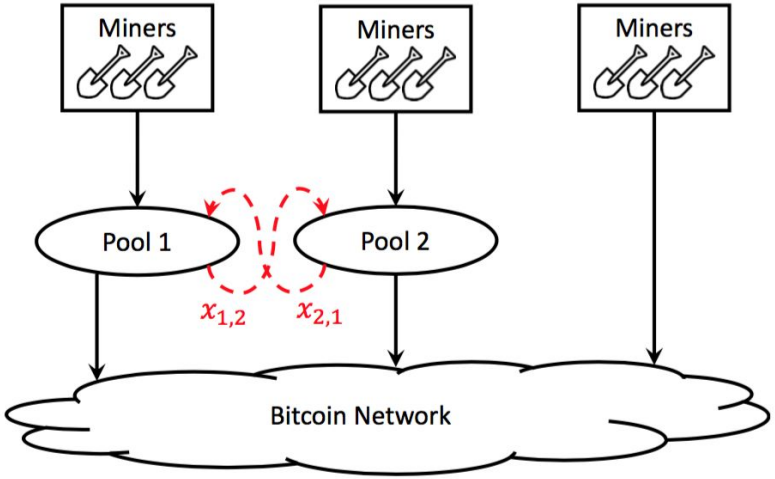


Fig. 7. Two pools attacking each other.

Nash Equilibrium

	POOL 2 IS IDLE	POOL 2 ATTACKS
POOL 1 IS IDLE	POOL 1: **+3** POOL 2: **+3**	POOL 1: **+1** POOL 2: **+4**
POOL 1 IS ATTACKS	POOL 1: **+4** POOL 2: **+1**	POOL 1: **+2** POOL 2: **+2**

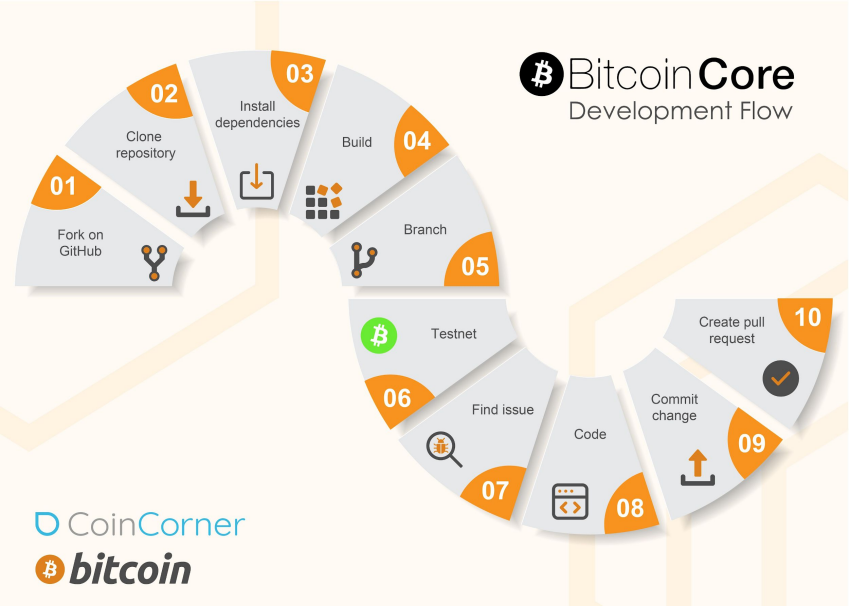
Nash equilibrium is a game state in which no player can earn a higher reward by **changing** strategies, assuming all other players' strategies remain **unchanged**.

AUTHOR: ANDREW KIRILLOV

Forking & Double Spends

Bitcoin Core

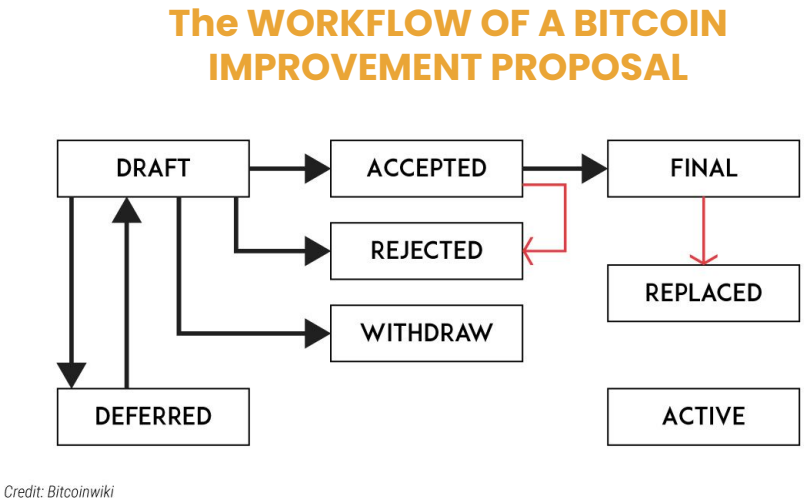
- **Bitcoin Core:**
  - The team of developers in charge of the Bitcoin GitHub repo
  - The software designed by these developers used by full Bitcoin nodes



AUTHOR: MAX FANG

Bitcoin Improvement Proposals

- **BIP:** Bitcoin Improvement Proposal
  - Three types:
    - Standards Track BIPs
    - Informational BIPs
    - Process BIPs
- First BIP proposed by Amir Taaki on 2011-08-19
- Signal support for a BIP by including reference in block when mining

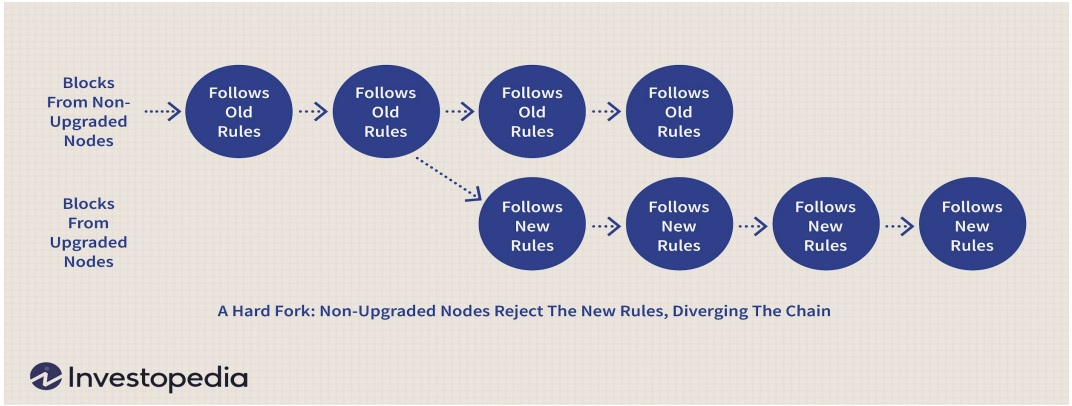


Credit: Bitcoinwiki

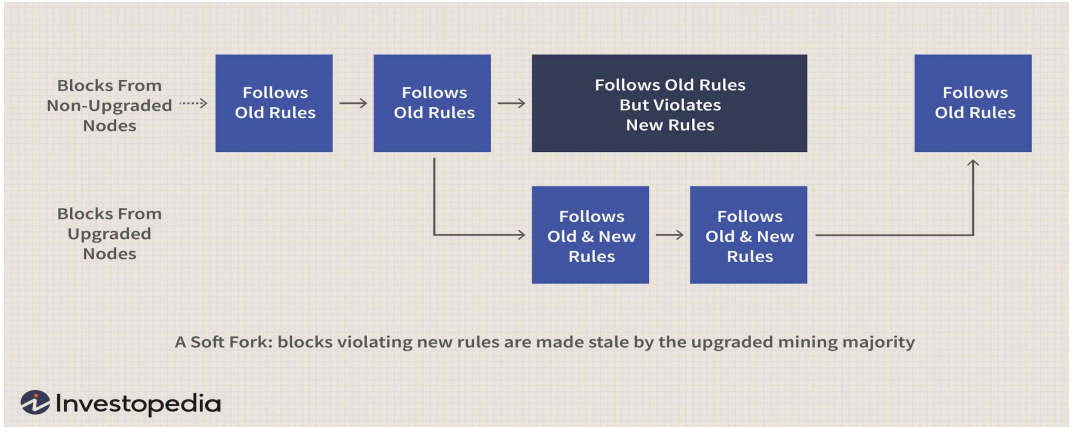
Source: https://en.bitcoin.it/wiki/Bitcoin_Improvement_Proposals

Forks

Hard Fork



Soft Fork



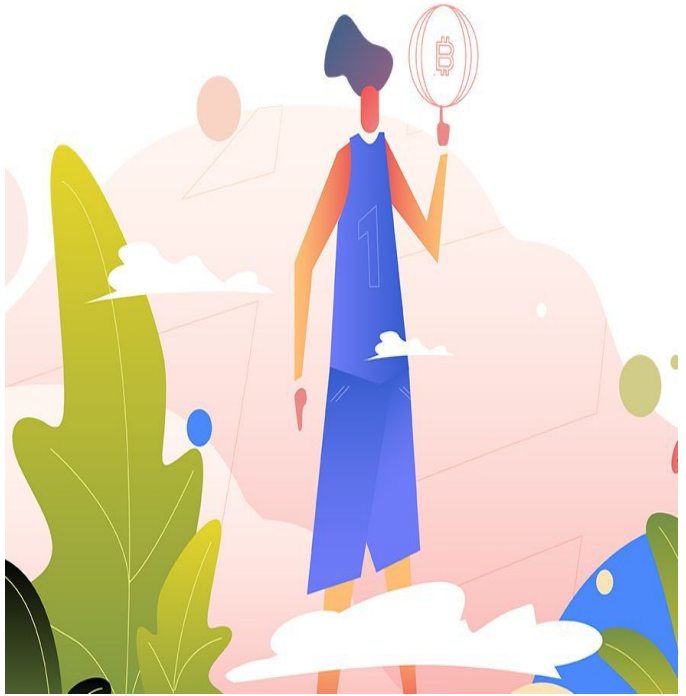
Source:
Investopedia

AUTHOR: MAX FANG

Double Spending

Double Spend: Successfully spending the same value more than once.

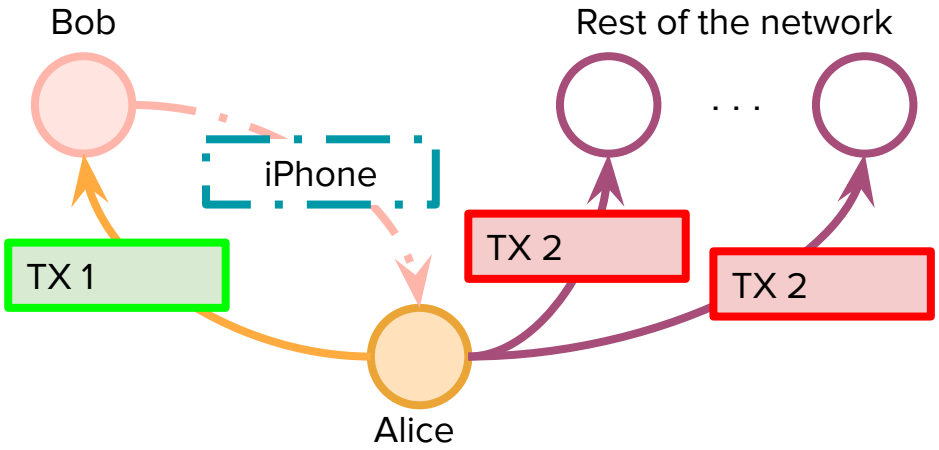
- In the year 2099, Alice wants to buy an iPhone 92XCS from Bob on the black market for 100 BTC but doesn't want to give up her bitcoins. #HODL
  - How can Alice double spend on Bob?

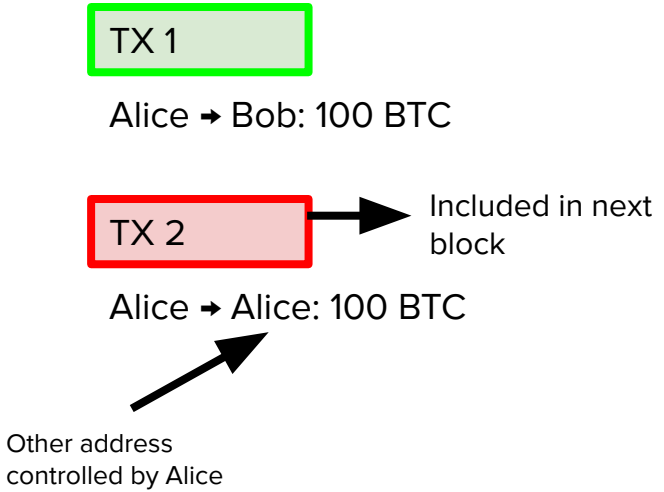


Race Attack

Suppose Bob simply checks that the transaction he sees is valid and **immediately** sends Alice the iPhone. Bob is vulnerable to a **Race Attack**!

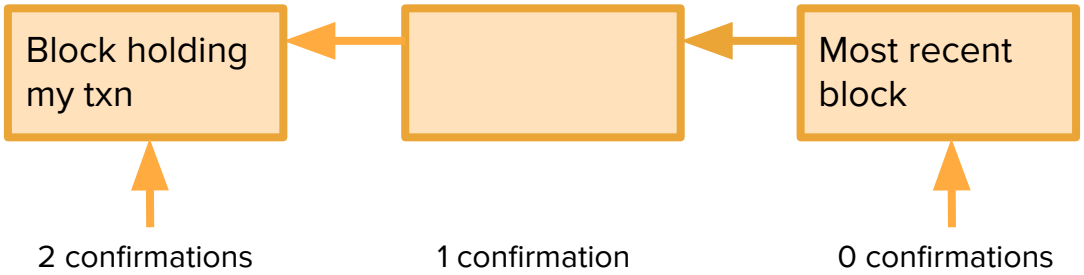
How can we stop this?





Confirmations

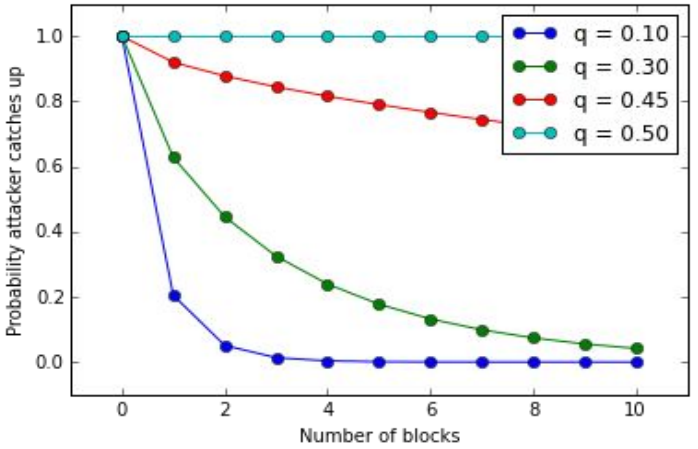
Confirmations: The number of blocks created on top of the block a txn is in.

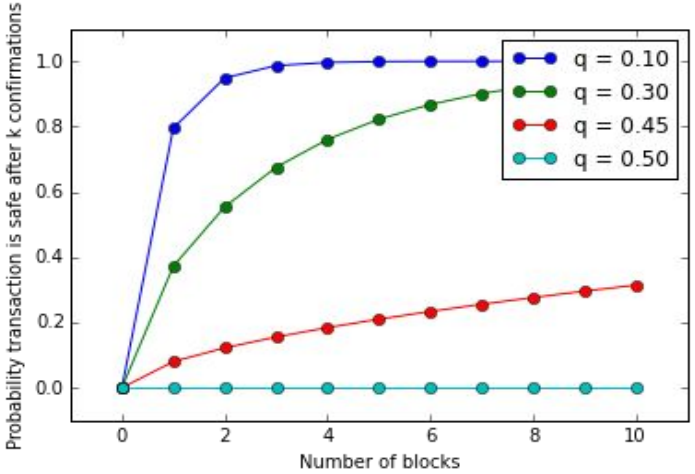


Probabilities

Probabilities of success for the attacker

★ (Inverse represents bounds of the probability of safety for the vendor given assumptions of the attacker's hashpower)





AUTHOR: PHILIP HAYES

51% Attack

What if Alice controls more than 50% of the total network hash power?

Whenever Alice's chain is behind the honest network's chain, she will *always* (in expectation) be able to catch up and out-produce the honest miners.

Therefore, the probability that Alice can successfully **double spend** with >50% hash power reaches 100%!

AUTHOR: PHILIP HAYES
UPDATED BY: RUSTIE LIN

Incentives, Pt. 1

Why would Alice not want to double spend?

If the rest of the network detects the double spend, it is assumed that confidence in the cryptocurrency and exchange rate would *plummet*.

Bribing Miners:

Alice might not physically control the mining hardware necessary to perform a double spend.

Instead, Alice can bribe miners or even entire pools to mine on her withheld chain.

AUTHOR: PHILIP HAYES

Incentives, Pt. 2

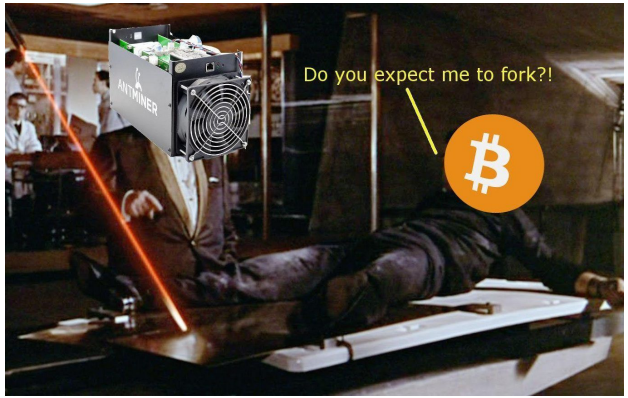
What if Alice is a **hostile government** / **adversarial altcoin** / **large finance institution** with *significant capital* available?

Alice can **acquire** enough mining ASICs or **bribe** enough miners / pools to achieve >50% effective hash power.

Alice can perform a so-called **"Goldfinger"** attack

- Objective: *destroy* target cryptocurrency, by destroying cryptocurrency confidence with a double spend or spamming the network with empty blocks.

Ex: Eligius pool kills CoiledCoin altcoin

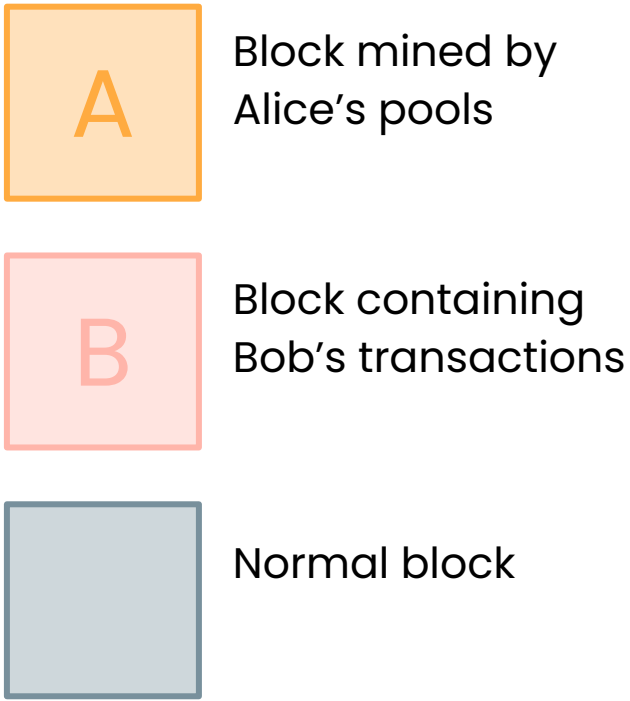


Censorship

Setting the Stage

Say Alice is a government, or has control over a government, that has jurisdiction over mining pools.
In addition, Alice's mining pools control **over 51% of the network's hashrate.**

Objective: Censor the Bitcoin address owned by Bob, and prevent them from spending any of their Bitcoin.



AUTHOR: MAX FANG

Naive Strategy

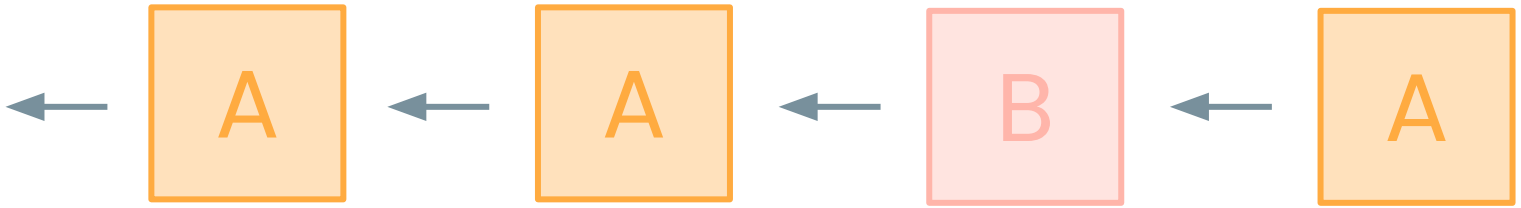
First strategy:
Alice tells her pools not to include Bob's transactions **(blacklisting)**.

Does this strategy actually work?

Naive Strategy

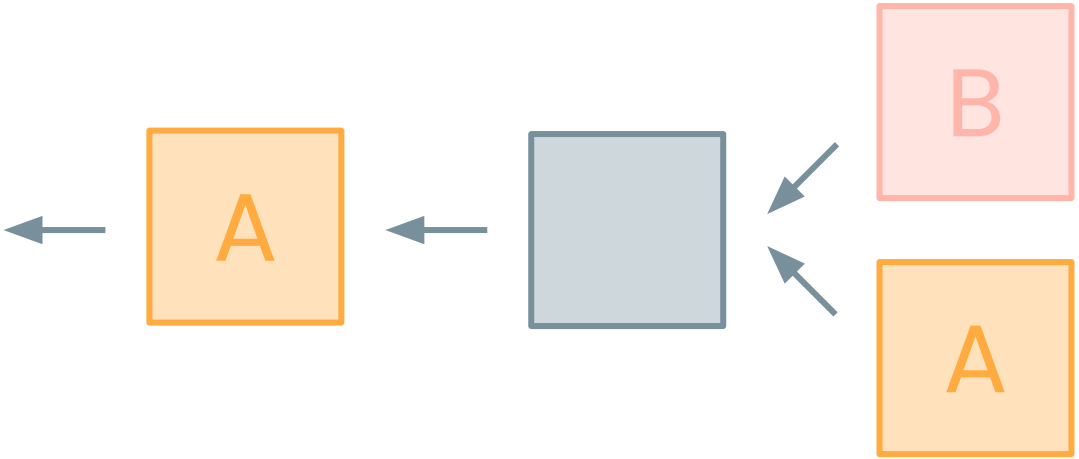
First strategy:
Alice tells her pools not to include Bob's transactions **(blacklisting)**.

- Doesn't work unless you are 100% of the network
- Other miners will eventually include Brian's transactions in a block
- Can only cause delays and inconveniences



Punitive Forking

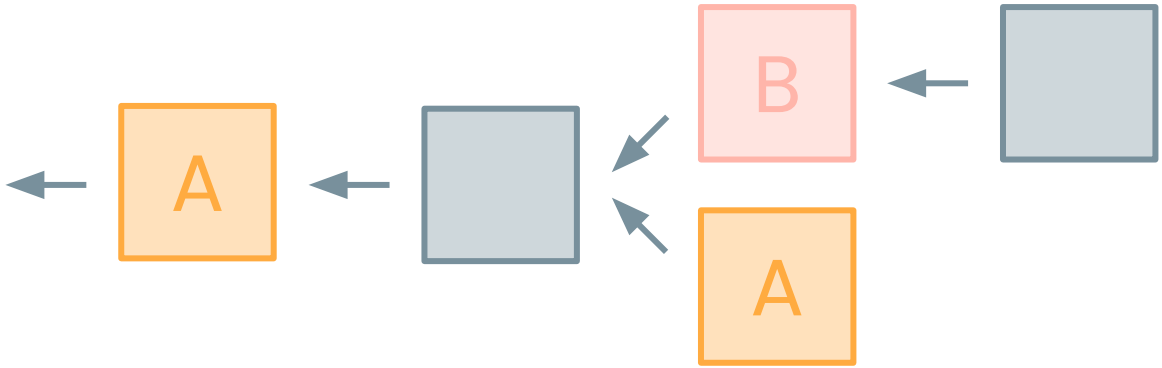
Second strategy:
Alice tells her pools not to mine on top of any chain containing Bob's transactions, and announces this to the world.



Does this strategy actually work?

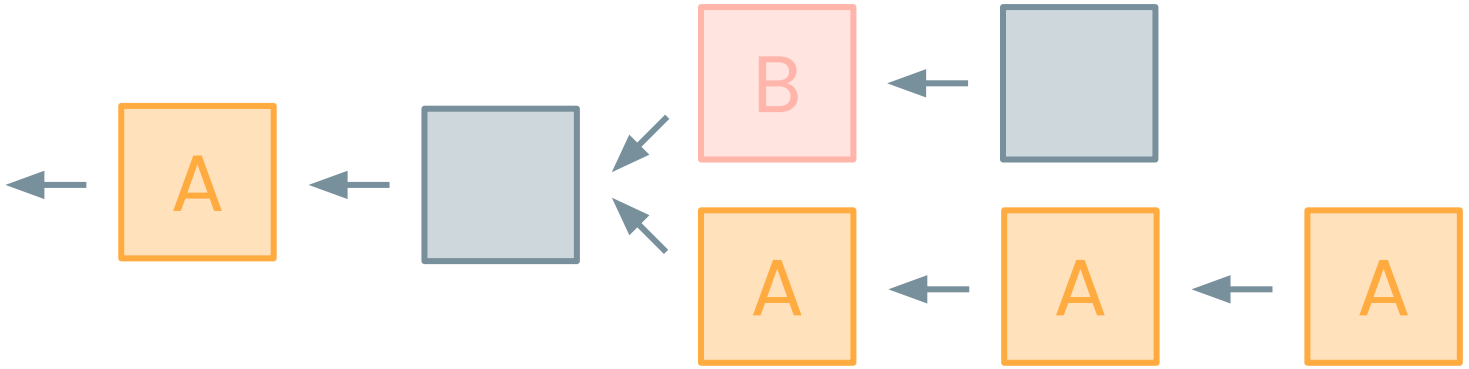
Punitive Forking

Remember, Alice controls >51% of hashrate...



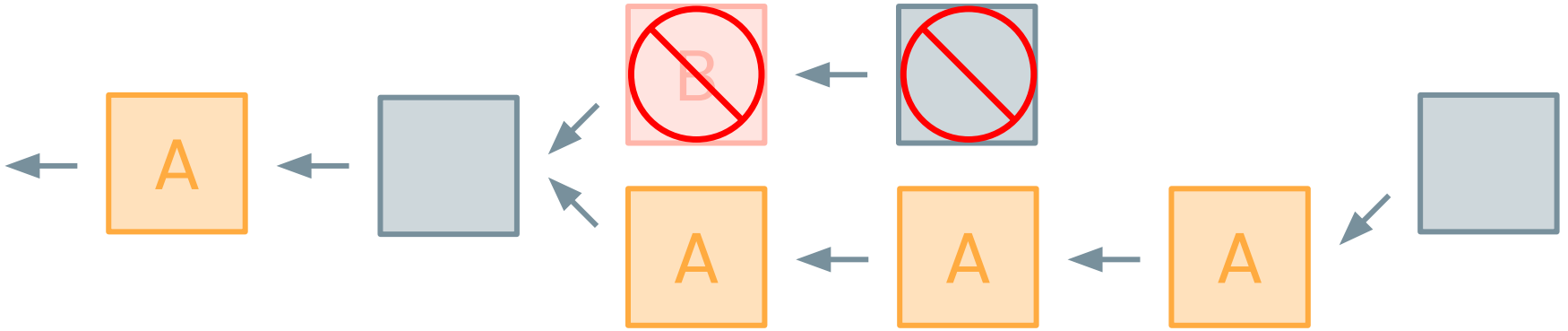
Punitive Forking

So, in time, her pools will mine the longest chain...



Punitive Forking

Invalidating Bob's transactions, and those in subsequent blocks.



Punitive Forking

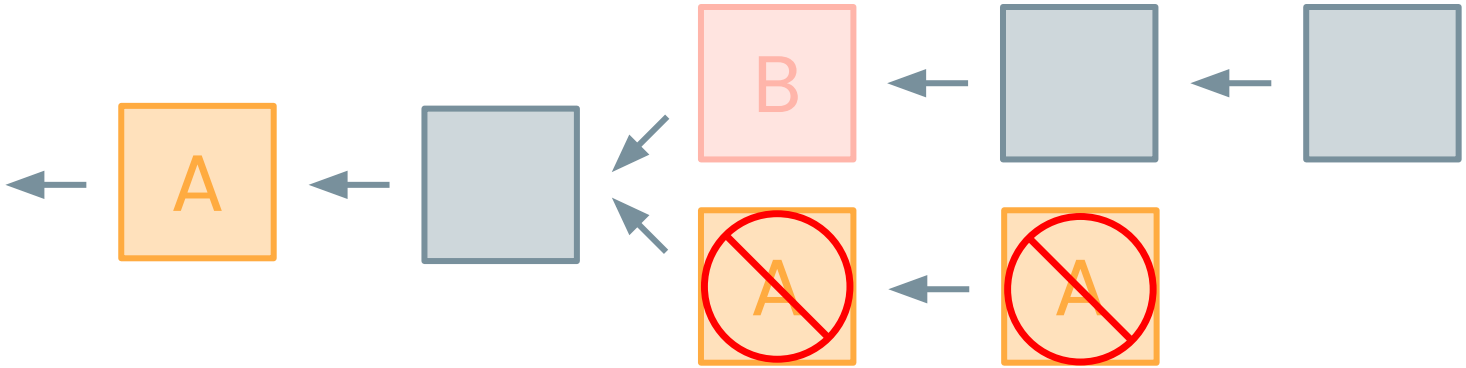
Miners outside of Alice's pools will stop including Bob's transactions when mining blocks, since Alice announced she would invalidate their blocks (and thus mining rewards)

Thus, we've shown that an entity with >51% of the network's hashrate can prevent someone from accessing their funds (or receiving any!)

Feather Forking

Punitive forking doesn't work unless Alice has >51% of hashpower. Is there another way? Yes! Called **Feather Forking**

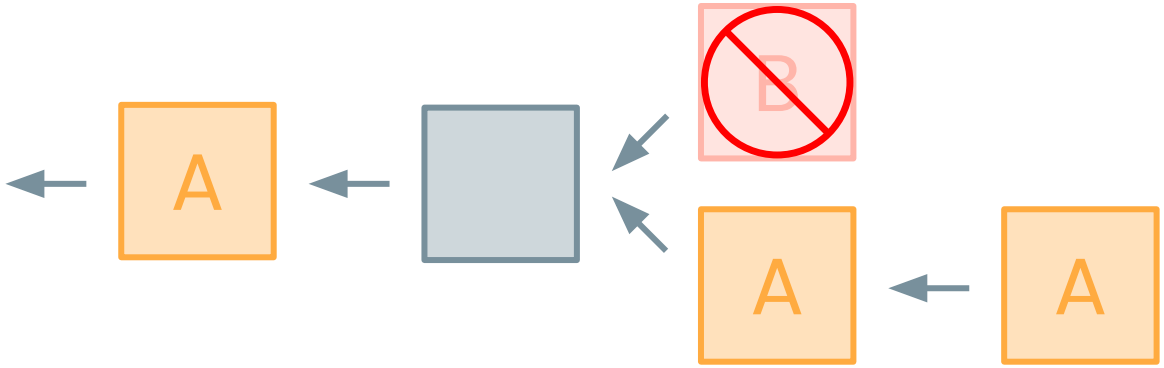
- New strategy: Alice announces that she will **attempt** to fork if she sees a block from Bob, but she will give up after a block with Bob's tx has **k** confirmations
  - As opposed to attempting to fork forever; doesn't work without >51%



Feather Forking

Let **q** equal the proportion of mining power Alice has, 0 < **q** < 1
Let **k** = 1: Alice will give up after 1 confirmation (one additional block)

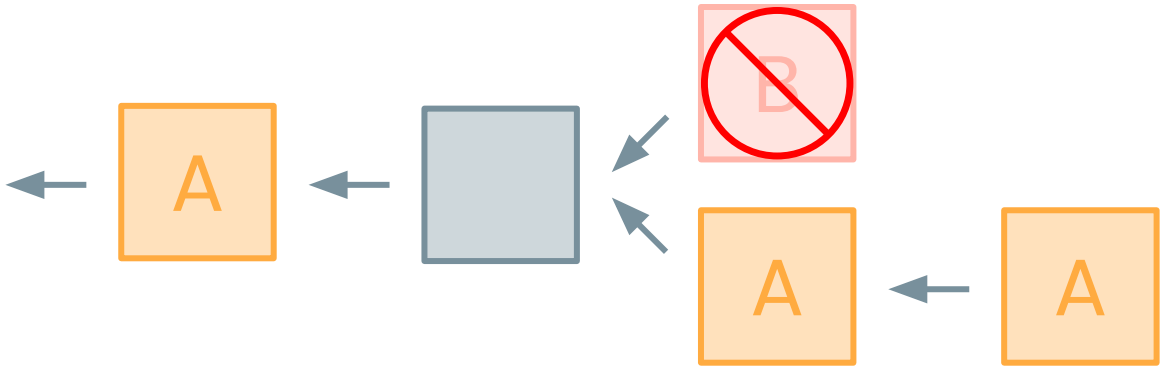
- Chance of successfully orphaning (invalidating) the Bob block = $\mathbf{q^2}$
  - Put up 2 blocks before 1 more gets put on top of the Bob block
- If **q** = 0.2, then $\mathbf{q^2}$ = **4%** chance of orphaning block. Not very good



AUTHOR: MAX FANG

Feather Forking

But other miners are now aware that their block has a $\mathbf{q^2}$ chance of being orphaned. Is it still worth it to include Bob's tx in their block?



Feather Forking

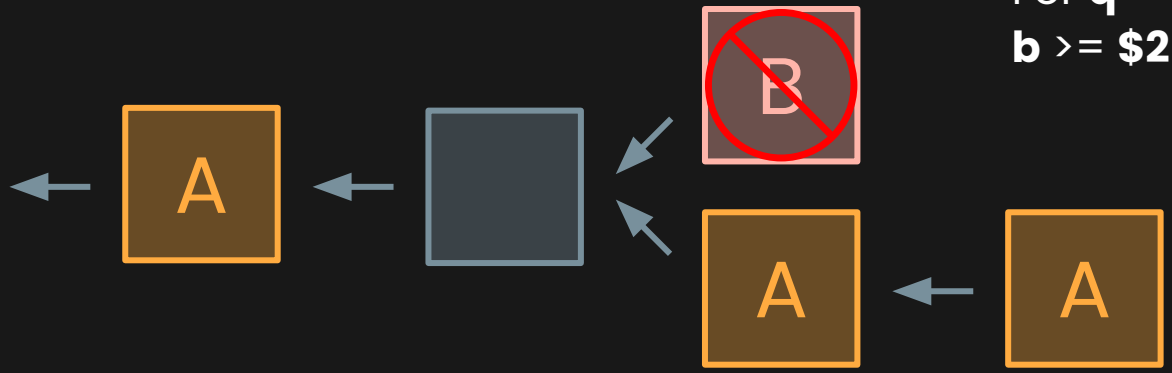
Let **b** be Bob's tx fees
Let **r** be the rest of the block rewards

E[include] = $(1 - \mathbf{q})^2 * (\mathbf{b} + \mathbf{r})$
E[exclude] = **r**

For Bob's tx to be included:

$(1 - \mathbf{q})^2 * (\mathbf{b} + \mathbf{r}) >= \mathbf{r}$
$\mathbf{b} >= \mathbf{q}^2 * \mathbf{r}$

For **q** = 0.2, **r** = 6.25 BTC, 1 BTC = $10k
b >= **$2500**



AUTHOR: MAX FANG

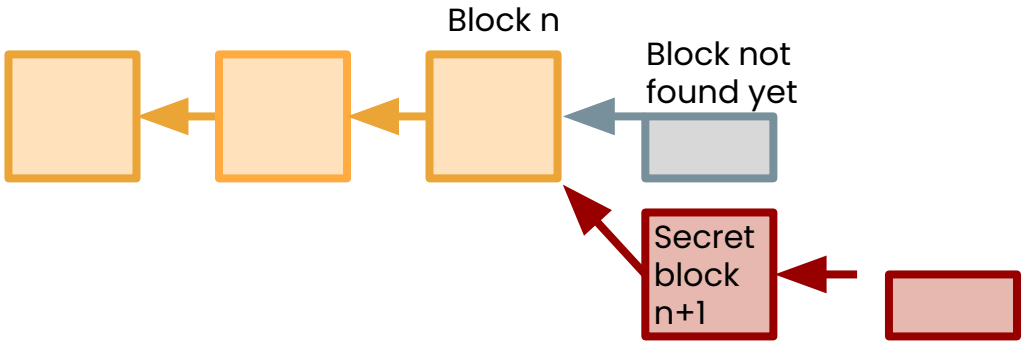
Selfish Mining

Block Withholding

You are a miner; suppose you have just found a block.

- Instead of announcing block to the network and receiving reward, keep it secret
- Try to find two blocks in a row before the network finds the next one

This is called **selfish mining** or **block-withholding**

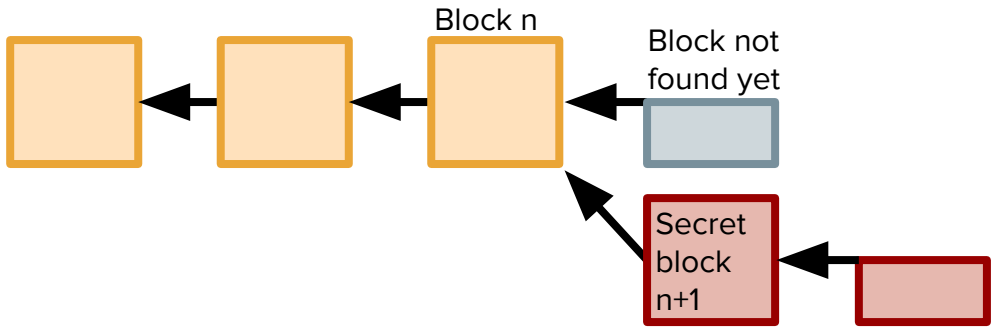


Note: "block-withholding" is also sometimes used in the context of mining pools - submitting shares but withholding valid blocks

Block Withholding

If you succeed in finding a second block, you have fooled the network

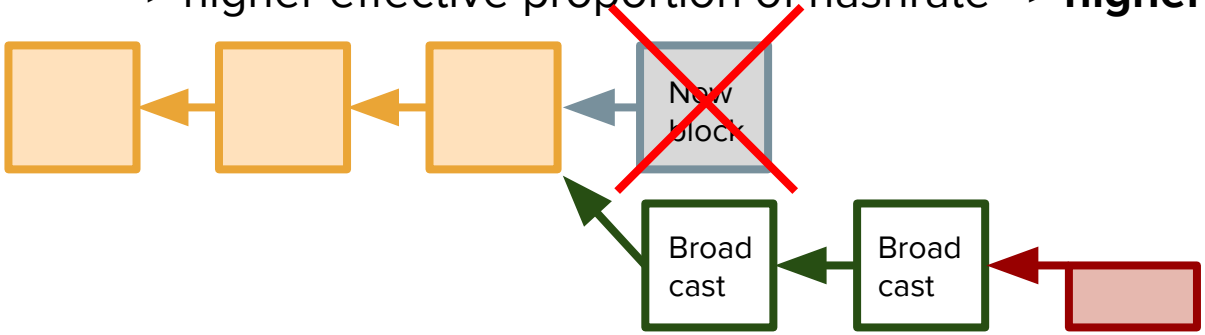
- Network still believes it is mining on the longest proof of work chain
- You continue to mine on your own chain



Block Withholding

If the network finds a block, you broadcast your two secret blocks and make the network block invalid

- While network was working on the invalid block, you got a bunch of time to mine by yourself... for free!
- Free time mining on network
  => higher effective proportion of hashrate **=> higher expected profits!**

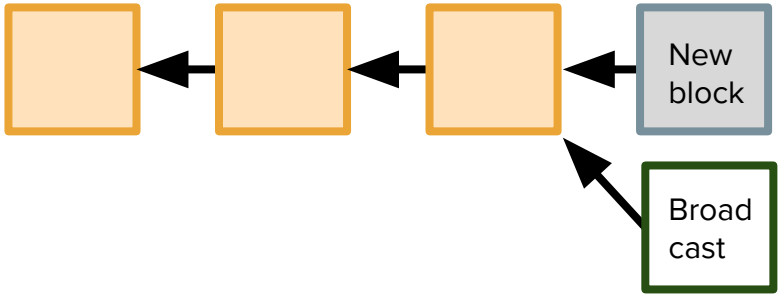


Block Withholding

But what if the network found their new block before you could find a second one?

Race to propagate!

- If on average you manage to tell 50% of the network about your block first:
  - Malicious strategy is more profitable if you have >25% mining power
- If you have >33% mining power, **you can lose the race every time and malicious strategy is still more profitable!**



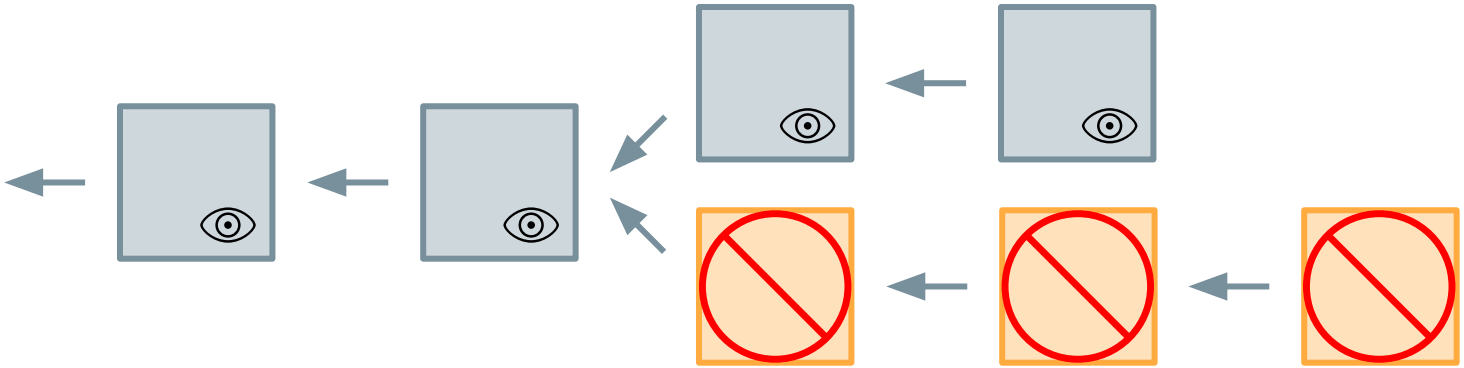
  - (actual math omitted due to complexity)

Selfish Mining Defenses

Certification of Witness

Proposed by Schultz (2015)

- Accompany each block with a set of signatures from other nodes
  - Select a random subset of nodes
- Signifies that they were aware the block was being mined



Certification of Witness

Needs:

- Way to choose a random subset of nodes
- Way to see which nodes were assigned to a given block

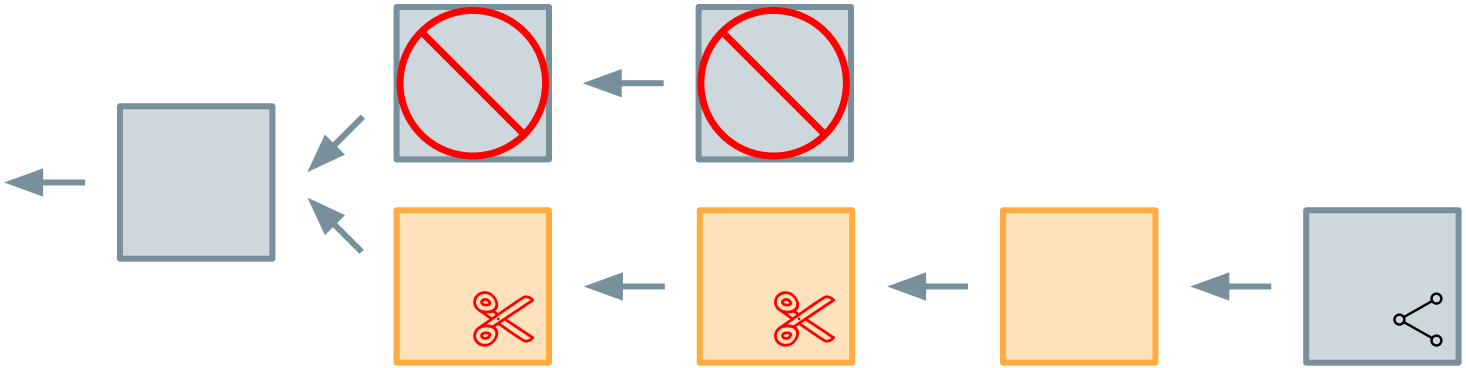
Otherwise, this scheme is not **Sybil-resistant** (can be taken advantage of by an attacker creating multiple identities)

This requires a hard fork of the protocol, and relies on **oracles** (3rd-party services that provide external, or “off-chain,” data to a blockchain network)

Fork Punishment

Proposed by Lear Bahack (2013)

- New approach: use economic incentives
- The first miner to incorporate proof of a fork in their block gets half of the total block rewards in that fork, while the other half are destroyed



AUTHOR: MAX FANG, APARNA KRISHNAN

Fork Punishment

Drawbacks

- Honest miners are caught in the crossfire
- Makes fundamental changes to Bitcoin's monetary policy
- Also requires a hard fork
- Doesn't completely prevent selfish mining
    - If an attacker publishes a selfishly-mined fork that is >1.5x as long as the competing one, they'll still be profitable

Uniform Tie-Breaking

Proposed by Eyal and Sirer (2014)

- In the event of a tie, miners **choose randomly** which chain to mine on
  - Attackers can't benefit from superior network connectivity
- Attackers would need to command **>=25%** of network hashrate for selfish mining to be more profitable than honest mining
  - Right now, selfish mining is better even for miners with **0%** of the hashrate, if they're optimally connected
  - Selfish mining will **always** be better for miners with **>= 33%** of hashrate

Problem Statement

How do we disincentivize selfish mining even when the selfish miner has a longer chain?

Zhang and Preneel, “Publish or Perish” (2017)

Ren Zhang and Bart Preneel (Apr 2017) claim the best-yet defense of selfish mining

- Backwards compatible: No hard fork
- Disincentivizes selfish mining even when the selfish miner has a longer chain

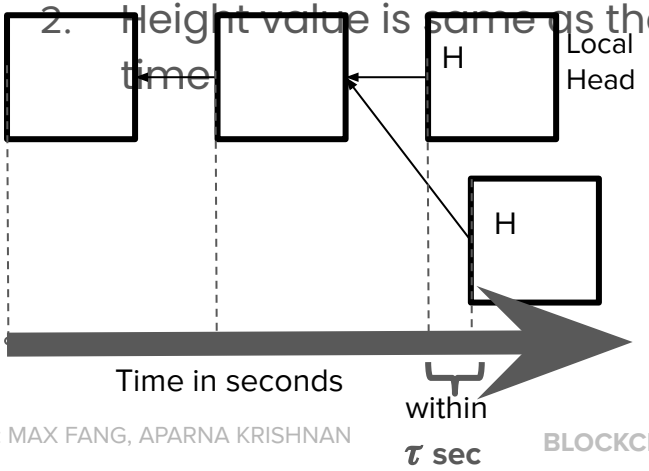
Approach: A novel **F**ork-**R**esolving **P**olicy (FRP)

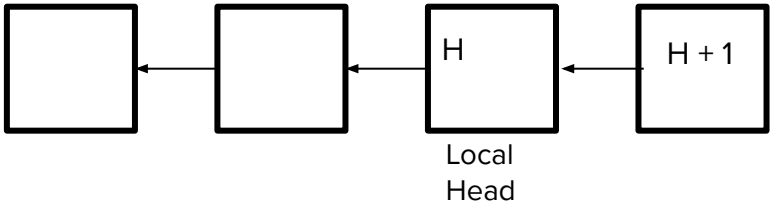
- Replace the original Bitcoin FRP (length FRP), with a **weighted FRP**
  - Embed in the working block the hashes of all its uncle blocks
- Note that selfish mining is premised on the idea of first building a secret block
- Idea: Make sure this secret block does not help the selfish miner win the block race

Zhang and Preneel, “Publish or Perish” (2017)

Definitions

- τ: An assumed upper bound on the amount of time it takes to propagate blocks across the Bitcoin network
- **In time.** Evaluated from the miner’s local perspective.
  1. Height value is greater than that of the local head OR
  2. Height value is same as that of the local head, but was propagated within τ time



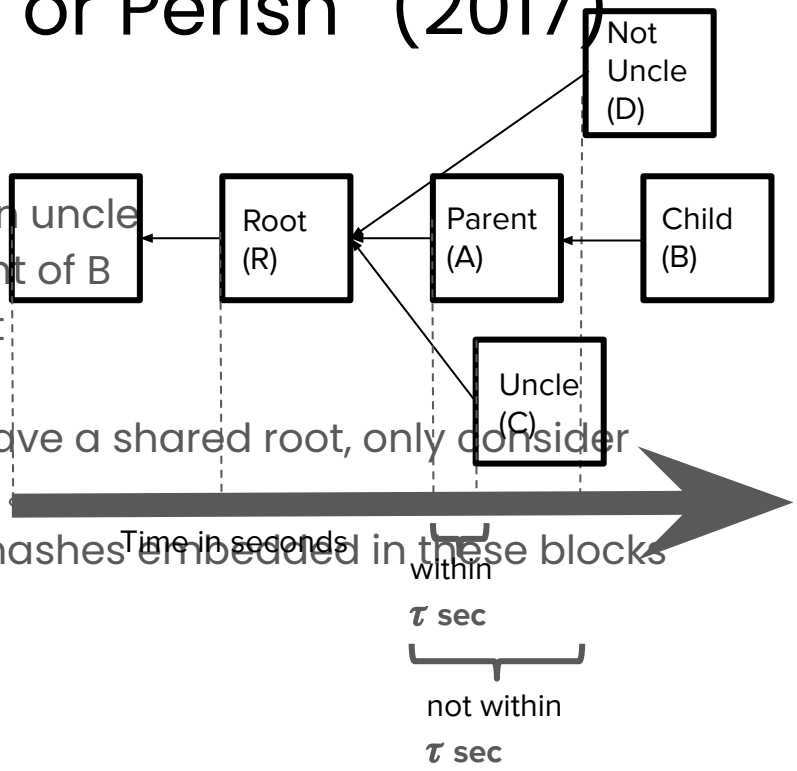


Zhang and Preneel, “Publish or Perish” (2017)

Definitions

- **Uncle**. Different from Ethereum’s definition of an uncle
  1. The uncle of a block B is one less the height of B
  2. The uncle has to be in time with B’s parent

- **Weight**. Since two competing chains always have a shared root, only consider blocks after that
  - weight = # of in time blocks + # of uncle hashes embedded in these blocks



Zhang and Preneel, “Publish or Perish” (2017)

Zhang and Preneel's **Weighted Fork Resolving Policy**:

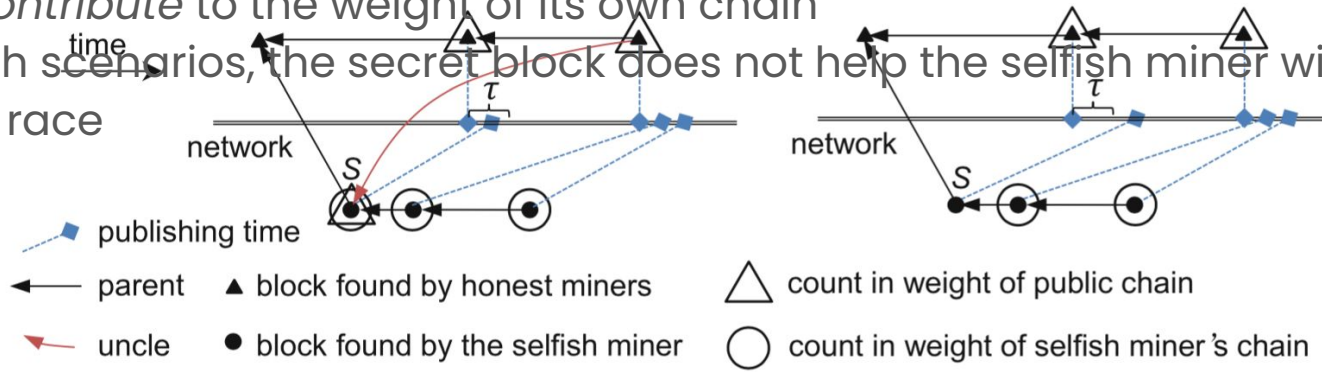
1. If one chain is longer *height-wise* than the other(s) by **k** or greater blocks*
   a. The miner will mine on this chain
2. Otherwise, the miner will choose the chain with the largest *weight*
3. If the largest weight is achieved by multiple chains simultaneously, then the miner chooses one among them randomly

Aside*: ***k* *is a "fail-safe parameter" that gauges the allowed amount of network partition. Note that when* ***k*** *= ∞ the first rule never applies.*

Zhang and Preneel, “Publish or Perish” (2017)

Miner has one secret block. A competing block is published. **Block race**! Miner has two options:

- Option 1: If the selfish miner **publishes** their block, *the next honest block gains a higher weight* by embedding a proof of having seen this block
- Option 2: If the selfish miner **keeps their block secret**, the secret block *does not contribute* to the weight of its own chain
- In both scenarios, the secret block does not help the selfish miner win the block race



Choice 1: Publish

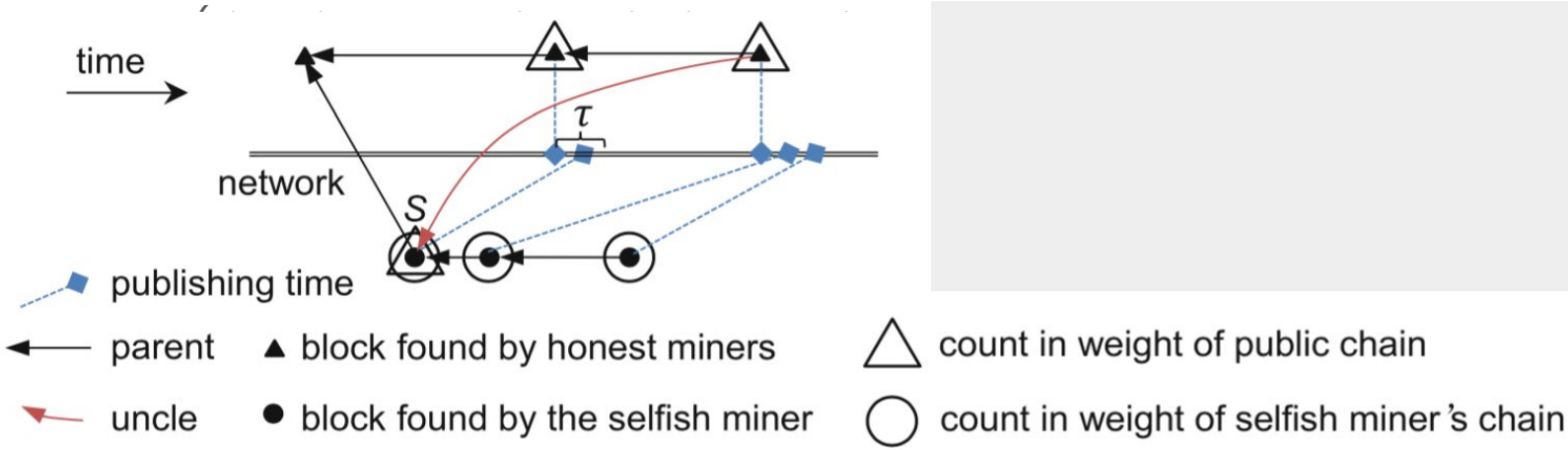
Choice 2: Don't publish

Zhang and Preneel, “Publish or Perish” (2017)

Same scenario revisited. More rigorously, let S be the first selfish block

Option 1: Selfish miner publishes S

- S will be an uncle of the next honest block

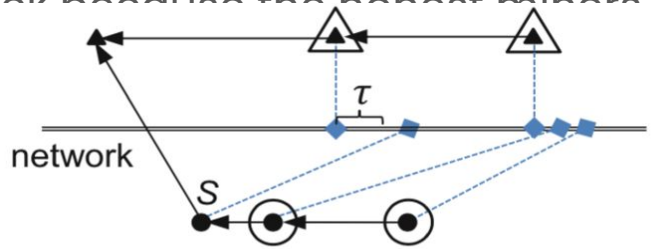


Choice 1: Publish

Zhang and Preneel, “Publish or Perish” (2017)

Option 2: Selfish miner doesn't publish S

- Selfish miner waits, and publishes it later as a part of the selfish chain
- Honest miners do not count S into the weight of the selfish chain because S is not *in time.*
  - It is a **late block**
- S is not an *uncle* of the next honest block because the honest miners did not



publishing time

parent

uncle

block found by honest miners

block found by the selfish miner

count in weight of public chain

count in weight of selfish miner ’s chain

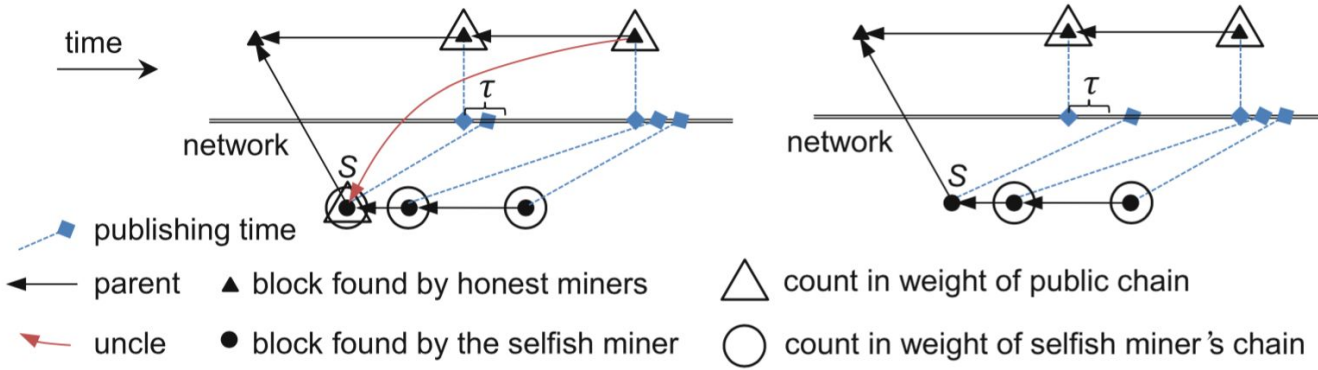
Choice 2: Don't publish

AUTHOR: MAX FANG

Zhang and Preneel, “Publish or Perish” (2017)

Result: Regardless of which option is chosen...

- S will **not** contribute to **only** the weight of the selfish chain.
  - Will only contribute to **both** or **neither**
- Completely nullifies the advantage of the secret block S!



Choice 1: Publish

Choice 2: Don't publish

Zhang and Preneel, “Publish or Perish” (2017)

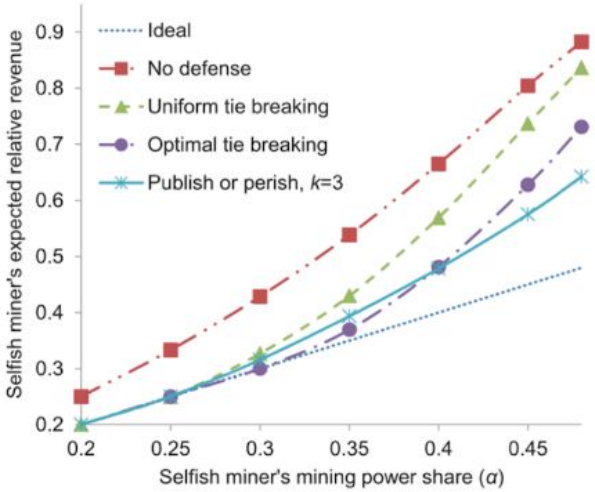


Fig. 5. Comparison with other defenses

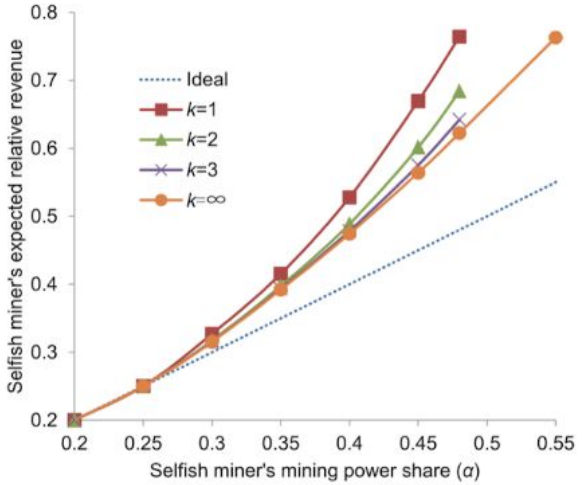


Fig. 4. Relative revenue of the selfish miner within our defense

Zhang and Preneel, “Publish or Perish” (2017)

Limitations

- Bitcoin aims be asynchronous, Publish and Perish assumes synchronicity
  - (Because it assumes an upper bound of block propagation time)
  - Because of this, it's basically useless
- When the fail-safe parameter k > 1, an attacker may broadcast blocks right before they are late to cause inconsistent views among the honest miners
  - Several other selfish mining defenses also require a fixed upper bound on the block propagation time in order to be effective
- During the transition period to weighted FRP, an attacker can launch double-spend attacks
- Neglects real world factors:
  - Does not permit the occurrence of natural forks
  - Does not consider transaction fees on the selfish miner’s strategy
  - Does not consider how multiple selfish miners could collude and compete with each other
- Does not achieve incentive compatibility, but is the closest scheme to date