CS251 Fall 2023

(cs251.stanford.edu)

Consensus in the Internet Setting

Ertem Nusret Tas

Recap of the Last Lecture

- Byzantine Generals Problem
- Definition of Byzantine adversary
  - **Byzantine:** Adversarial nodes can deviate from the protocol arbitrarily!
- Synchronous and asynchronous networks
  - **Synchronous network:** known upper bound Δ on network delay
- Byzantine Broadcast
- Dolev-Strong (1983)
- State Machine Replication (SMR)
- Security properties for SMR protocols: Safety and Liveness

Sybil Attack

How to select the nodes that participate in consensus?

Two variants:

- *Permissioned:* There is a *fixed* set of nodes (previous lecture).
- *Permissionless*: Anyone is free to join the protocol at any time.

Can we accept any node that has a signing key to participate in consensus?

Sybil Attack

How to select the nodes that participate in consensus?

Two variants:

- *Permissioned:* There is a *fixed* set of nodes (previous lecture).
- *Permissionless*: Anyone is free to join the protocol at any time.

Can we accept any node that has a signing key to participate in consensus?

In a **sybil attack**, a single adversary impersonates many different nodes, outnumbering the honest nodes and potentially disrupting consensus.

Sybil Resistance

Consensus protocols with Sybil resistance are typically based on a bounded (scarce) resource:

	Resource dedicated to the protocol	Some Example Blockchains
Proof-of-Work	Total computational power	Bitcoin, PoW Ethereum…
Proof-of-Stake	Total number of coins	Algorand, Cardano, Cosmos, PoS Ethereum…
Proof-of-Space/Time	Total storage across time	Chia, Filecoin…

How does Proof-of-Work prevent Sybil attacks?

We assume that the adversary controls a small fraction of the scarce resource!

Resource gives the power to influence the protocol.

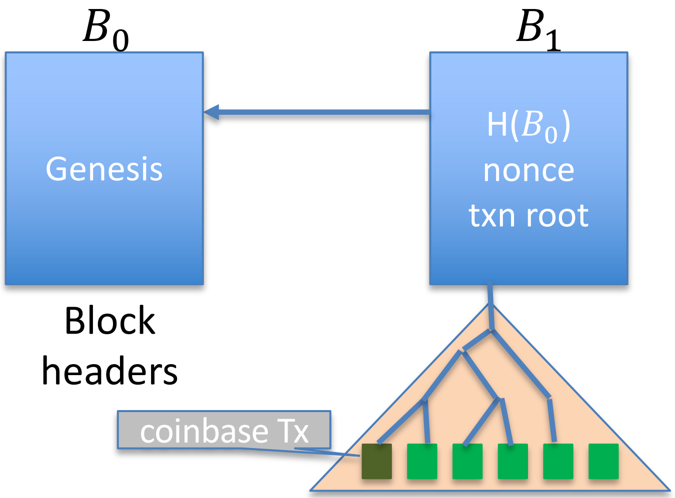
Adversary has less influence than honest nodes.

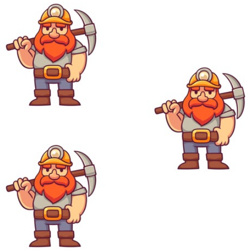
Bitcoin: Mining

To mine a new block, a miner must find $nonce$ such that

$$H(h_{prev}, \text{txn root}, \text{nonce}) < \text{Target} = \frac{2^{256}}{D}$$

Each miner tries different nonces until one of them finds a nonce that satisfies the above equation.





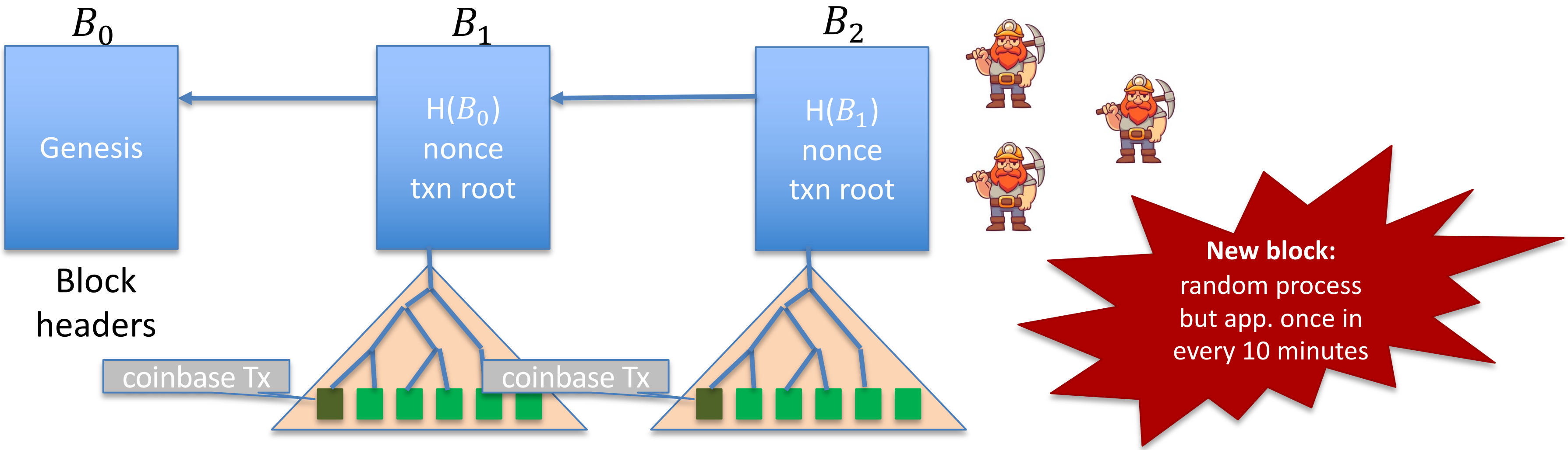
Bitcoin: Mining

To mine a new block, a miner must find *nonce* such that

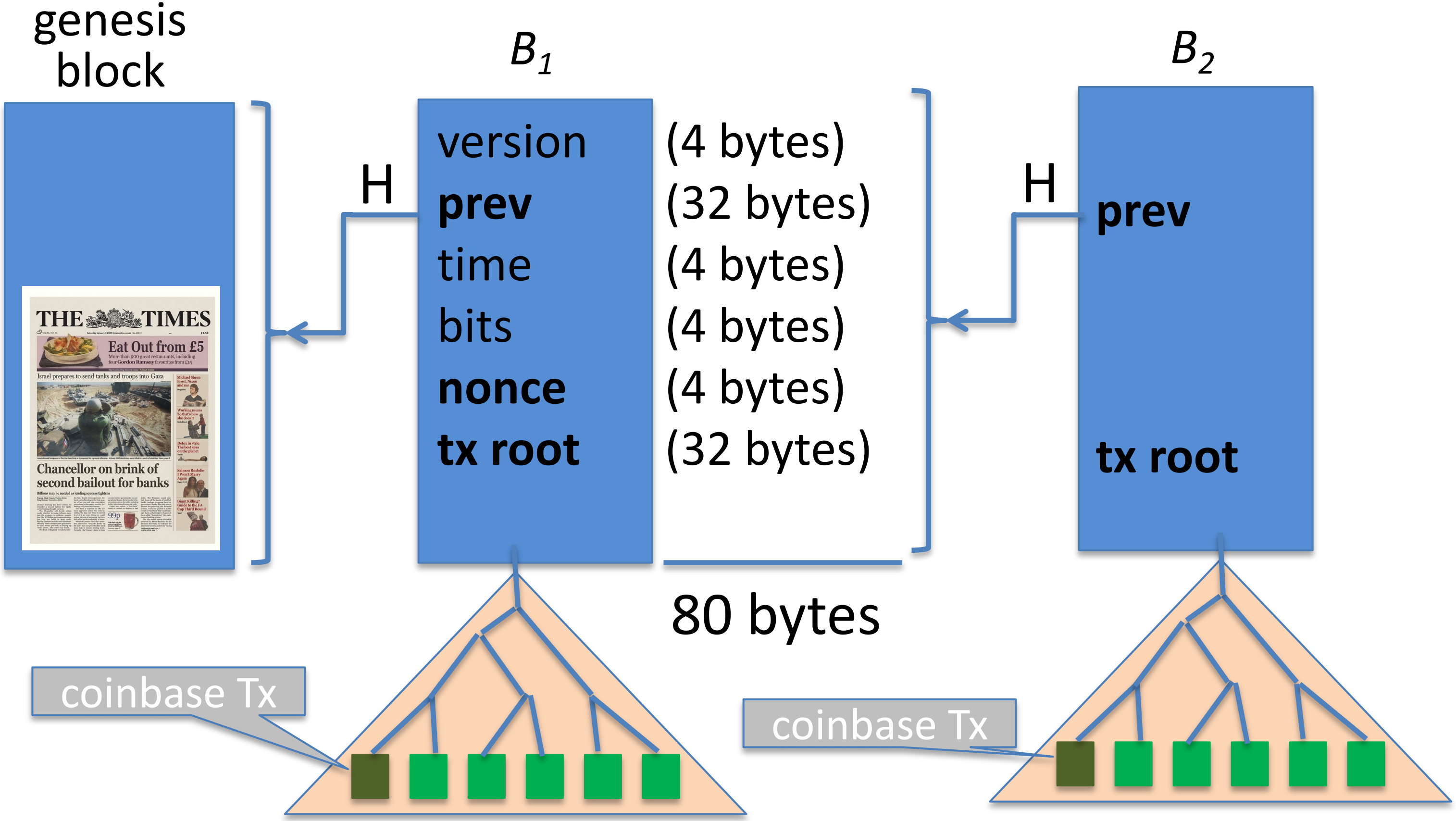
$$H(h_{prev}, \text{txn root}, \text{nonce}) < \text{Target} = \frac{2^{256}}{D}$$

Difficulty: How many nonces on average miners try until finding a block?

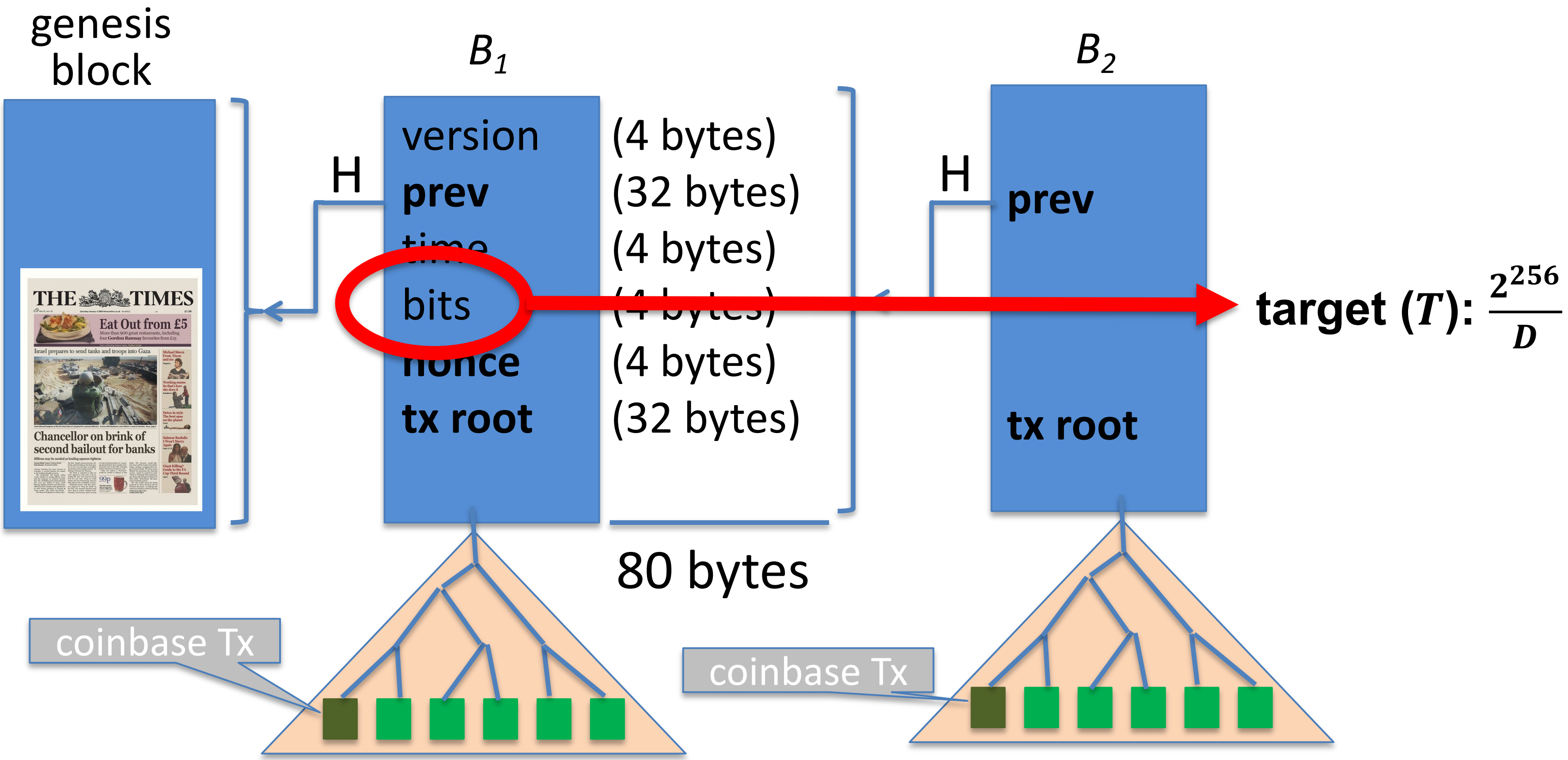
Each miner tries different nonces until one of them finds a nonce that satisfies the above equation.



Bitcoin: Mining



Bitcoin: Mining

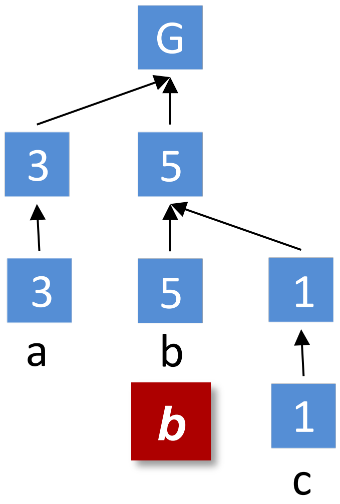


Nakamoto Consensus

Bitcoin uses **Nakamoto consensus**:

- **Fork-choice / proposal rule:** At any given time, each honest miner attempts to extend (i.e., mines on the tip of) the <u>heaviest</u> chain *held* in its view (Ties broken adversarially).

Difficulty values on the blocks



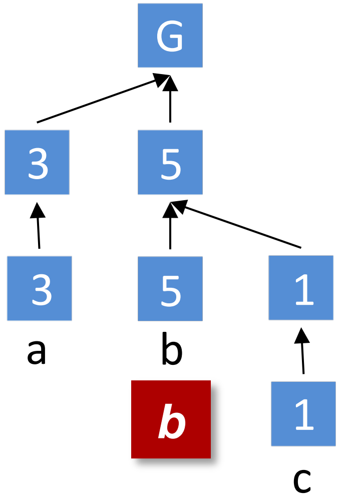
Nakamoto Consensus

Chain with the highest difficulty, i.e, largest sum of the difficulty D within blocks!

Bitcoin uses **Nakamoto consensus**:

- **Fork-choice / proposal rule:** At any given time, each honest miner attempts to extend (i.e., mines on the tip of) the heaviest chain *held* in its view (Ties broken adversarially).

Difficulty values on the blocks

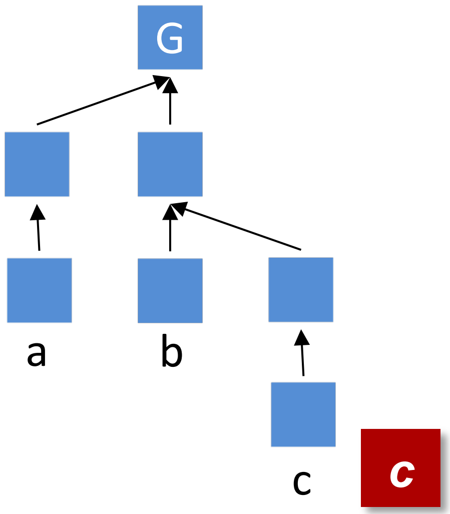


Nakamoto Consensus

Chain with the highest difficulty, i.e, largest sum of the difficulty D within blocks!

Bitcoin uses **Nakamoto consensus**:

- **Fork-choice / proposal rule:** At any given time, each honest miner attempts to extend (i.e., mines on the tip of) the <u>heaviest</u> (longest for us) chain *held* in its view (Ties broken adversarially).



Nakamoto Consensus

Bitcoin uses **Nakamoto consensus**:

- **Fork-choice / proposal rule:** At any given time, each honest miner attempts to extend (i.e., mines on the tip of) the heaviest (longest for us) chain *held* in its view (Ties broken adversarially).
- **Confirmation rule:** Each miner confirms the block (along with its prefix) that is *k*-deep within the longest chain in its view.
  - In practice, $k = 6$.
  - Miners and clients accept the transactions in the latest confirmed block and its prefix as their log.
  - Note that *confirmation* is different from *finalization*.
- **Leader selection rule:** Proof-of-Work.

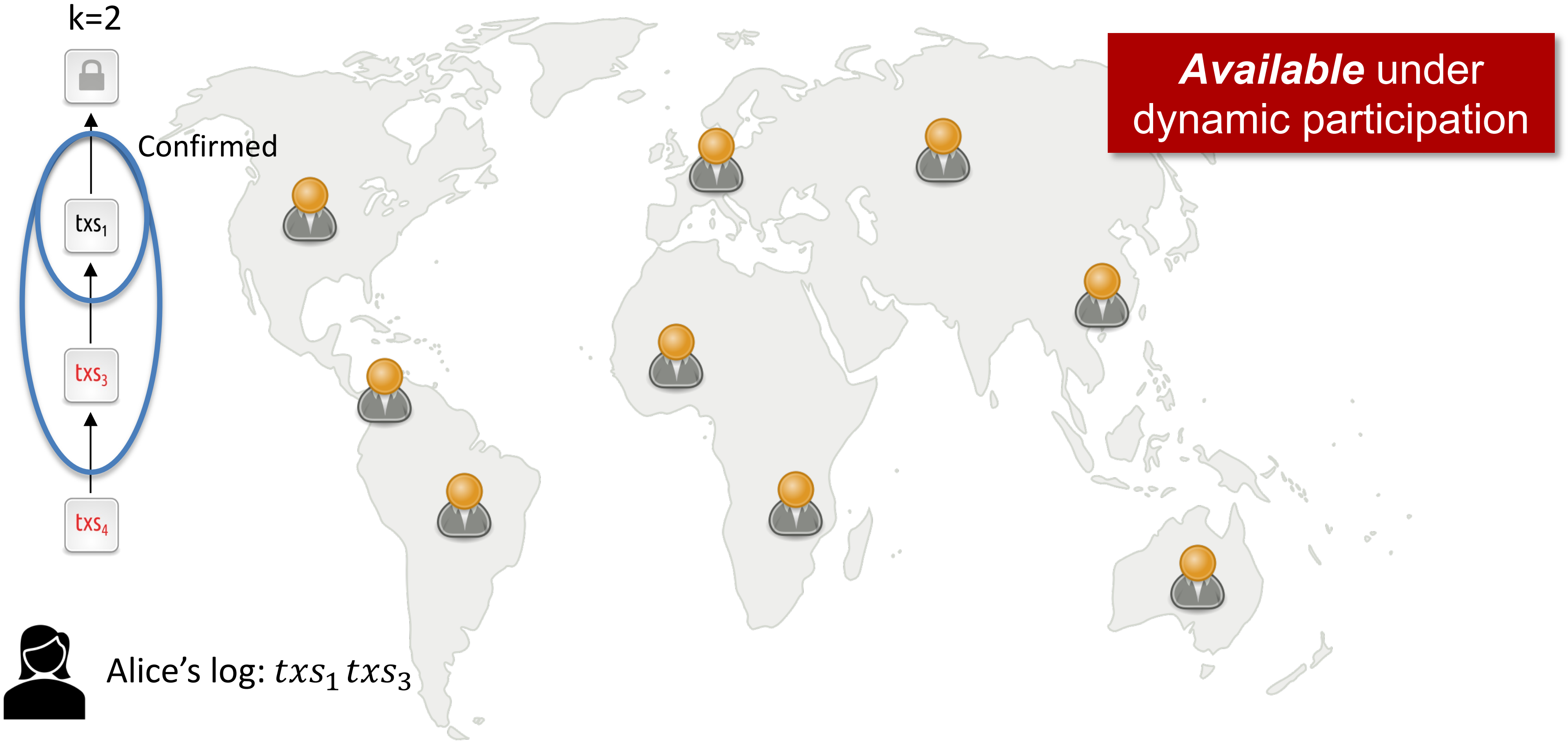
Nakamoto Consensus

Chain with the highest difficulty, i.e, largest sum of the difficulty D within blocks!

Bitcoin uses **Nakamoto consensus**:

- **Fork-choice / proposal rule:** At any given time, each honest miner attempts to extend (i.e., mines on the tip of) the heaviest (longest for us) chain *held* in its view (Ties broken adversarially).
- **Confirmation rule:** Each miner confirms the block (along with its prefix) that is *k*-deep within the longest chain in its view.
  - In practice, $k = 6$.
  - Miners and clients accept the transactions in the latest confirmed block and its prefix as their log.
  - Note that *confirmation* is different from *finalization*.
- **Leader selection rule:** Proof-of-Work.

Nakamoto Consensus



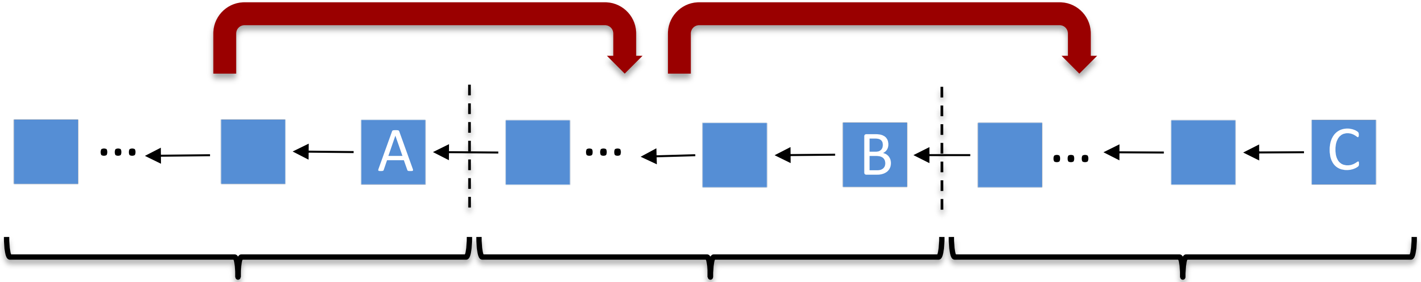
Available **under dynamic participation**

Alice's log: $txs_1\ txs_3$

Bitcoin: Difficulty Adjustment

New target: $T_2 = T_1 \frac{t_1}{2016 \times 10\ mins}$

New target: $T_3 = T_2 \frac{t_2}{2016 \times 10\ mins}$



2016 blocks
Time it took to mine: t_1(min)
Target: T_1

2016 blocks
Time it took to mine: t_2(min)
Target: T_2

2016 blocks
Time it took to mine: t_3(min)
Target: T_3

New target is not allowed to be more than 4x old target.
New target is not allowed to be less than ¼ x old target.

t_2: difference between the timestamps in B and A
t_3: difference between the timestamps in C and B

The Bitcoin Backbone Protocol with Chains of Variable Difficulty (2016)

Consensus in the Internet Setting

Characterized by *open participation*.

Challenges:

- Adversary can create many Sybil nodes to take over the protocol.
- Honest nodes can come and go at will.

Requirements:

Achieved by Bitcoin!

- Limit adversary's participation.
  - **Sybil resistance (e.g., Proof-of-Work)!**
- Maintain availability (liveness) of the protocol when the honest nodes come and go at will, resulting in changes in the number of nodes.
  - **Dynamic availability!**

Security?

Can we show that Bitcoin is a secure state machine replication (SMR) protocol (satisfies safety and liveness) under synchrony against a Byzantine adversary?

$\beta(t)$ $\in [0,1]$ for all t

Fraction of the mining power controlled by the adversary at time t.

What is the highest $\beta(t)$ for which Bitcoin is secure??

Model for Bitcoin

- Many different miners, each with *infinitesimal* power.

  Total mining rate (growth rate of the chain): λ (1/minutes). In Bitcoin, $\lambda = 1/10$.

- Suppose Adversary is Byzantine and controls $\beta < \frac{1}{2}$ fraction of the mining power.
  - **Adversarial mining rate**: $\lambda_a = \beta\lambda$
  - **Honest mining rate**: $\lambda_h = (1 - \beta)\lambda$

- Network is **synchronous** with a known upper bound Δ on delay.

Reminder: Why is safety important?

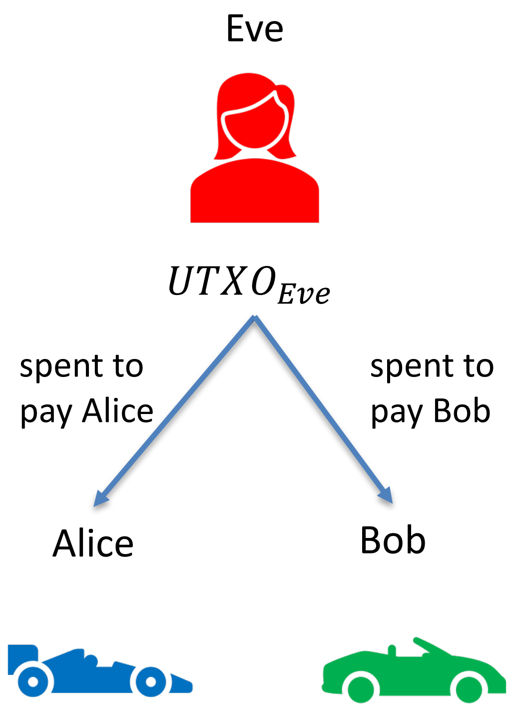
Suppose Eve has a UTXO.

- tx_1: transaction spending Eve's UTXO to pay to car vendor Alice.
- tx_2: transaction spending Eve's UTXO to pay to car vendor Bob.

$t_0 = 0$ $\quad t_1 \quad$ t_2

- Alice's ledger at time t_1 contains tx_1: $LOG_{t_1}^{Alice} =< tx_1 >$
- Alice thinks it received Eve's payment and sends over the car.

- Bob's ledger at time t_2 contains tx_2: $LOG_{t_2}^{Bob} =< tx_2 >$
- Bob thinks it received Eve's payment and sends over the car.

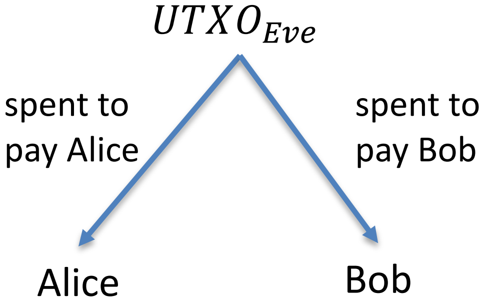


Reminder: Why is safety important?

Suppose Eve has a UTXO.

- tx_1: transaction spending Eve's UTXO to pay to car vendor Alice.
- tx_2: transaction spending Eve's UTXO to pay to car vendor Bob.

Eve



$t_0 = 0$ t_1 t_2

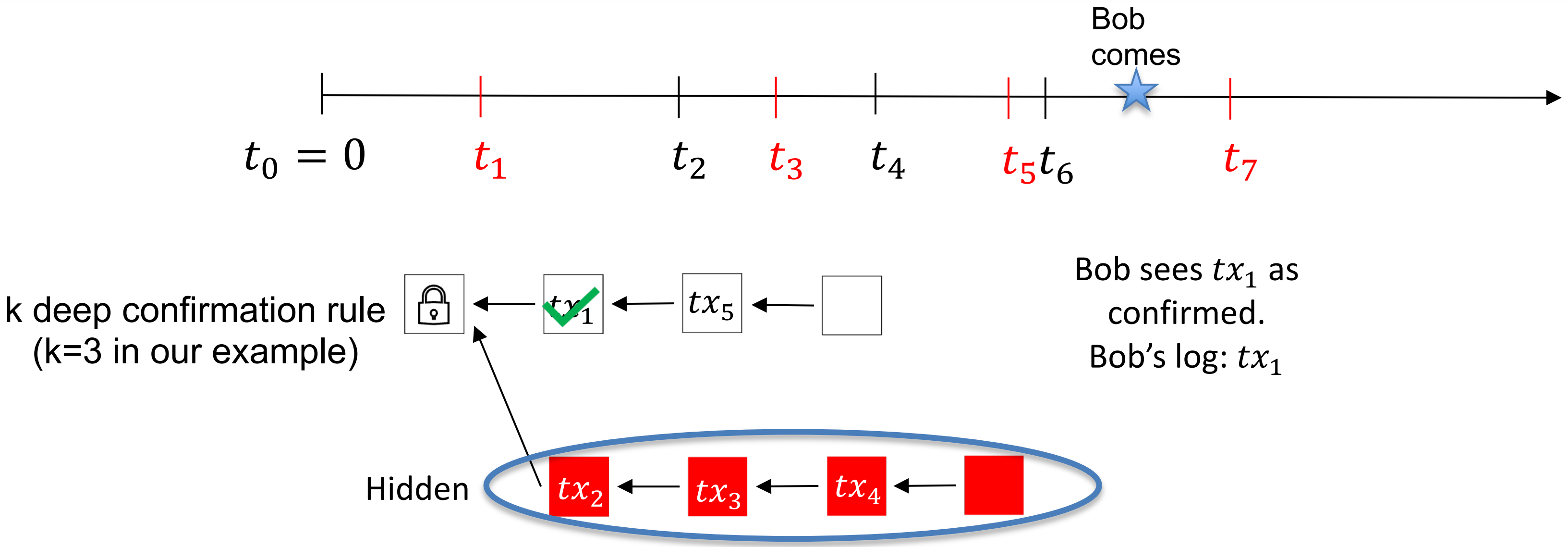
safety violation

- Alice's ledger at time t_1 contains tx_1: $LOG_{t_1}^{Alice} = < tx_1 >$
- Alice thinks it received Eve's payment and sends over the car.

- Bob's ledger at time t_2 contains tx_2: $LOG_{t_2}^{Bob} = < tx_2 >$
- Bob thinks it received Eve's payment and sends over the car.

When safety is violated, Eve can double-spend!

Nakamoto's Private Attack: $\beta \geq 1/2$

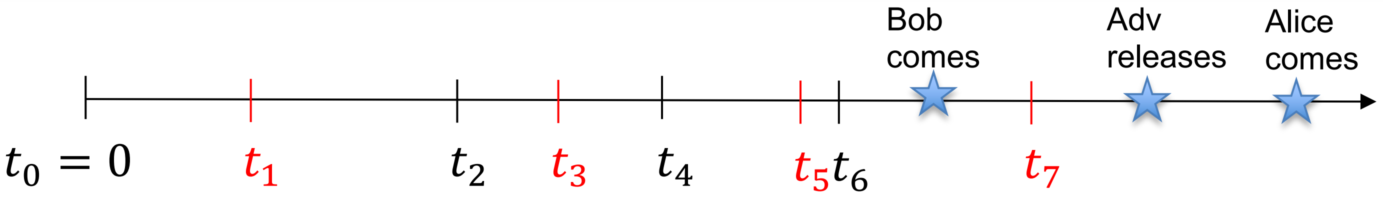


Let's show that Bitcoin is insecure if $\beta(t) \geq 1/2$

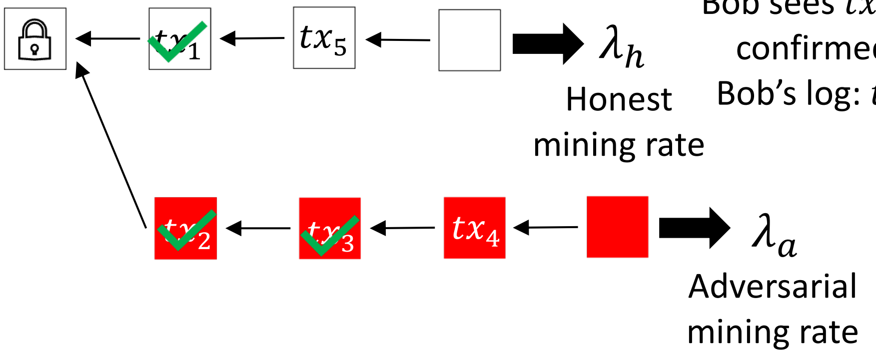
A Peer-to-Peer Electronic Cash System (2008)

Nakamoto’s Private Attack: $\beta \geq 1/2$

safety violation! double spend!



k deep confirmation rule (k=3 in our example)



Bob sees tx_1 as confirmed.
Bob’s log: tx_1

tx_1 got ‘reorged’: It was part of the longest chain before but not anymore!!

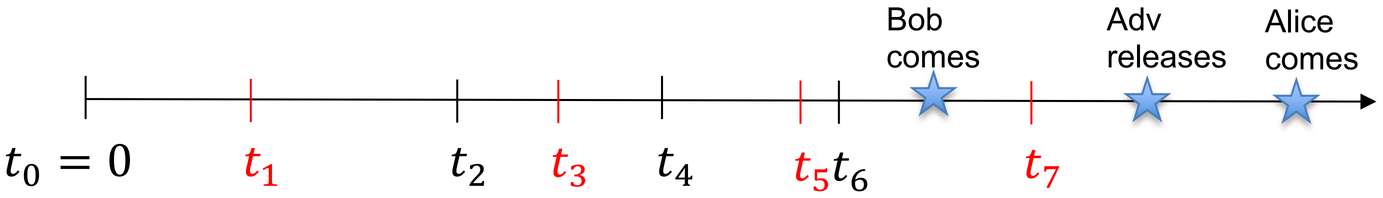
Alice sees the red chain as the longest chain.
tx_1 is not confirmed!
Alice’s log: $tx_2 tx_3$

Private attack succeeds!

A Peer-to-Peer Electronic Cash System (2008)

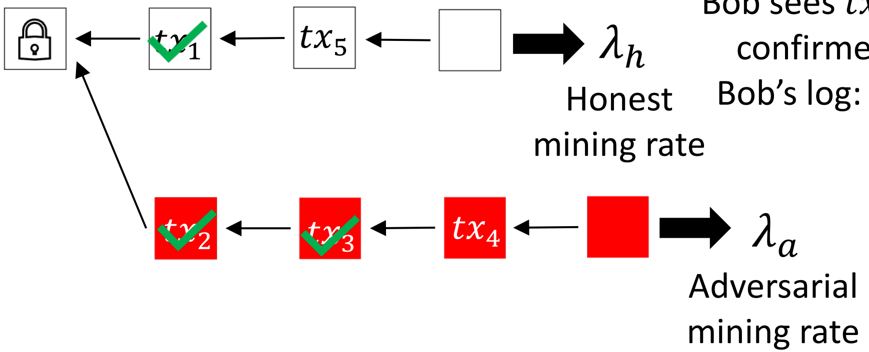
Nakamoto's Private Attack: $\beta \geq 1/2$

safety violation! double spend!



tx_1 got 'reorged': It was part of the longest chain before but not anymore!!

k deep confirmation rule (k=3 in our example)



Bob sees tx_1 as confirmed.
Bob's log: tx_1

Alice sees the red chain as the longest chain.
tx_1 is not confirmed!
Alice's log: $tx_2 tx_3$

Private attack succeeds!

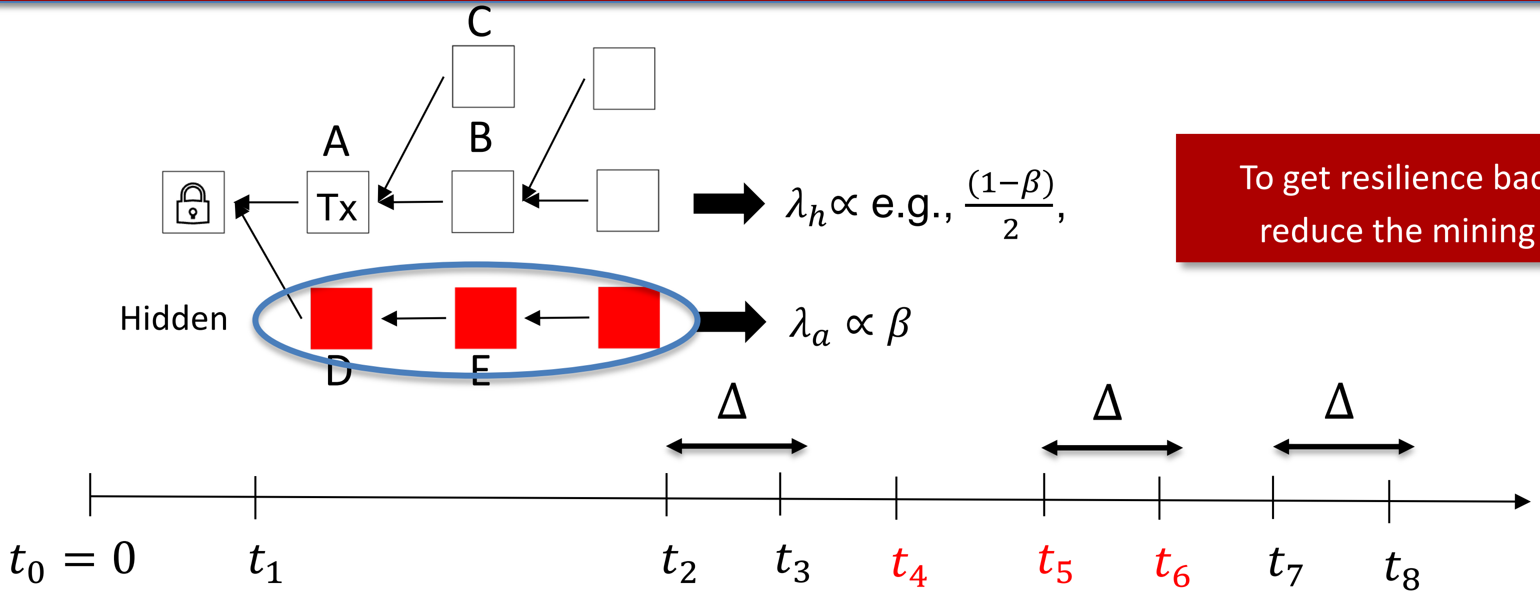
Private attack (mostly) succeeds if $\lambda_a \geq \lambda_h$, i.e., if $\beta \geq 1 - \beta$, i.e., if $\beta \geq \frac{1}{2}$.

Private attack (mostly) fails if $\lambda_a < \lambda_h$, i.e., if $\beta < 1 - \beta$, i.e., if $\beta < \frac{1}{2}$.

Can another attack succeed?

A Peer-to-Peer Electronic Cash System (2008)

Forking



To get resilience back to $\frac{1}{2}$, reduce the mining rate!

Multiple honest blocks at the same height due to network delay.

Adversary's chain grows at rate proportional to (shown by $\propto$) β!

Honest miners' chain grows at rate less than $1 - \beta$ because of forking!

Now, adversary succeeds if $\beta \geq \frac{(1-\beta)}{2}$, which implies $\beta \geq \frac{1}{3}$!!

Reminder for SMR Security

Let's recall the security definition for state machine replication (SMR) protocols. Let ch_t^i denote the confirmed (i.e., k-deep) of a client i at time t.

Safety (Consistency):

- For any two clients i and j, and times t and s: $\boldsymbol{ch_t^i} \preccurlyeq \boldsymbol{ch_s^j}$ (prefix of) or vice versa, i.e., chains are consistent.

No double spend

Liveness:

- If a transaction tx is input to an honest miner at some time t, then for all clients i, and times $s \geq t + T_{conf}$: $tx \in \boldsymbol{ch_s^i}$.

No censorship

Security Theorem

Theorem: If $\beta < 1/2$, there exists a small enough <u>mining rate</u> $\lambda(\Delta, \beta) = \lambda_a + \lambda_h$ such that Bitcoin satisfies safety and liveness <u>except with error probability $\epsilon = e^{-\Omega(k)}$</u> under synchronous network (recall that k is used in the k deep confirmation rule).

- $e^{-\Omega(k)}$ is the error probability for confirmation.
- Latest result for bounding the error probability as a function of k:

$$\epsilon \leq \left(2 + 2\sqrt{\frac{1-\beta}{\beta}}\right)\left(4\beta(1-\beta)\right)^k$$

- We say 'confirmation' instead of finalization because when you *confirm* a block or transaction, you *confirm* it with an error probability…
- …unlike *finalizing* a block where there is no error probability*.

The Bitcoin Backbone Protocol: Analysis and Applications (2015)
Analysis of the Blockchain Protocol in Asynchronous Networks (2016)
Analysis of Nakamoto Consensus (2019)
Everything is a Race and Nakamoto Always Wins (2022)
Bitcoin's Latency–Security Analysis Made Simple (2022)

Security Theorem

Theorem: If $\beta < 1/2$, there exists a small enough mining rate $\lambda(\Delta, \beta) = \lambda_a + \lambda_h$ such that Bitcoin satisfies safety and liveness except with error probability $\epsilon = e^{-\Omega(k)}$ under synchronous network (recall that k is used in the k deep confirmation rule).

- $e^{-\Omega(k)}$ is the error probability for confirmation.
- Latest result for bounding the error probability as a function of k:

$$\epsilon \leq \left(2 + 2\sqrt{\frac{1-\beta}{\beta}}\right)\left(4\beta(1-\beta)\right)^k$$

- We say 'confirmation' instead of finalization because when you *confirm* a block or transaction, you *confirm* it with an error probability…
- …unlike *finalizing* a block where there is no error probability*.

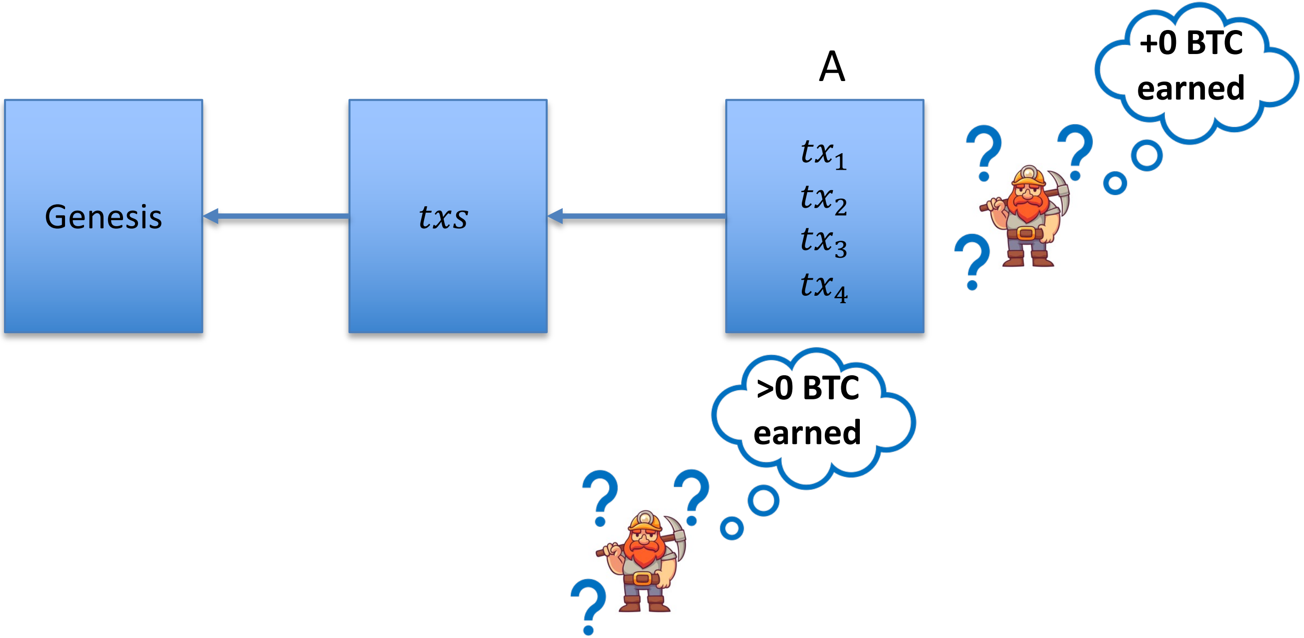
The Bitcoin Backbone Protocol: Analysis and Applications (2015)
Analysis of the Blockchain Protocol in Asynchronous Networks (2016)
Analysis of Nakamoto Consensus (2019)
Everything is a Race and Nakamoto Always Wins (2022)
Bitcoin's Latency–Security Analysis Made Simple (2022)

Now, we see why Bitcoin has 1 block every 10 minutes, instead of 1 block every second…

Proof of the Security Theorem

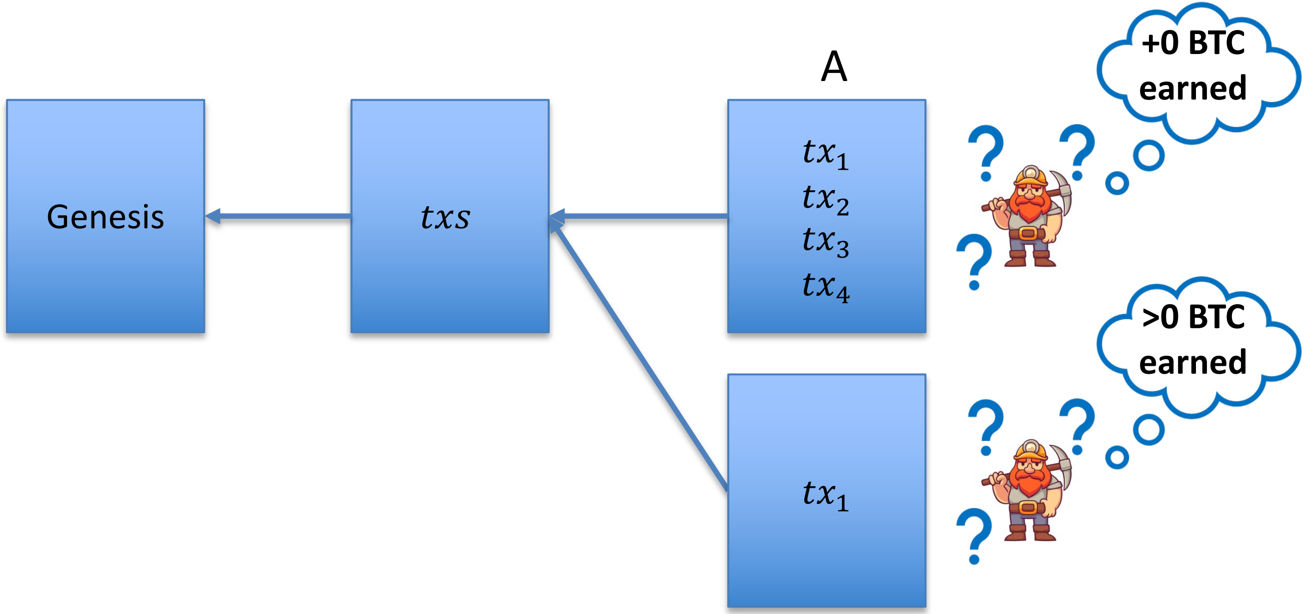
See the optional slides at the end of the deck …

Would $\beta < 1/2$ hold in practice?



Transaction fees
(paid to the miner):
tx_1: 4 BTC
tx_2: 3 BTC
tx_3: 2 BTC
tx_4: 1 BTC

Would $\beta < 1/2$ hold in practice?



Transaction fees (paid to the miner):

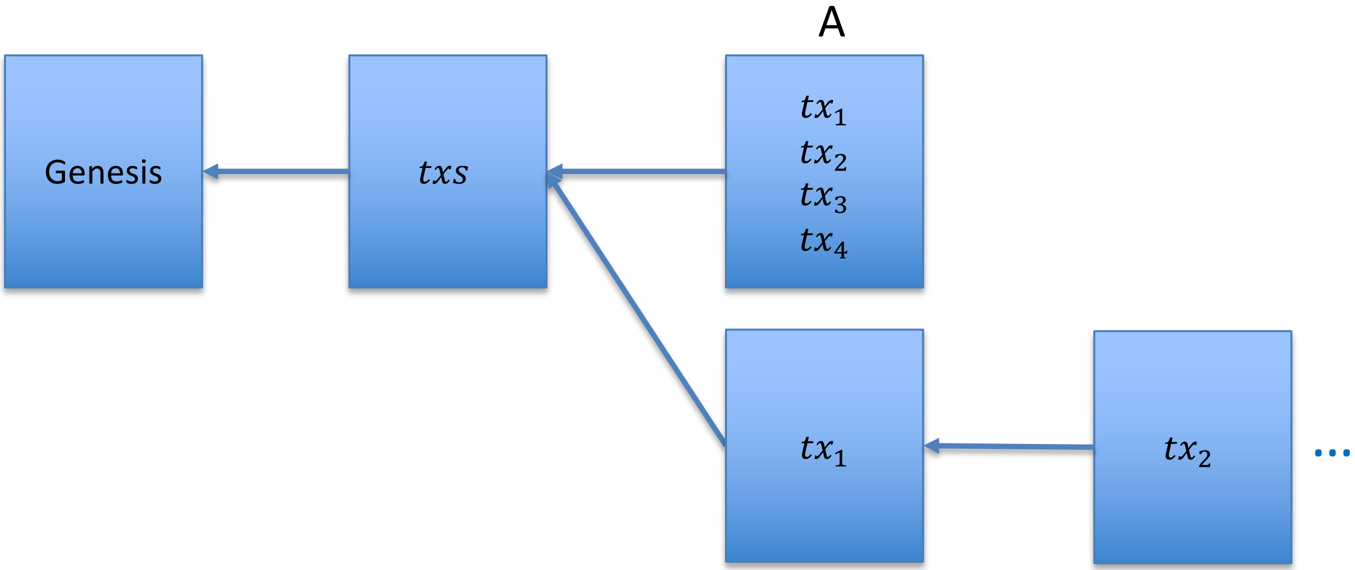
tx_1: 4 BTC

tx_2: 3 BTC

tx_3: 2 BTC

tx_4: 1 BTC

Would $\beta < 1/2$ hold in practice?



Transaction fees (paid to the miner):

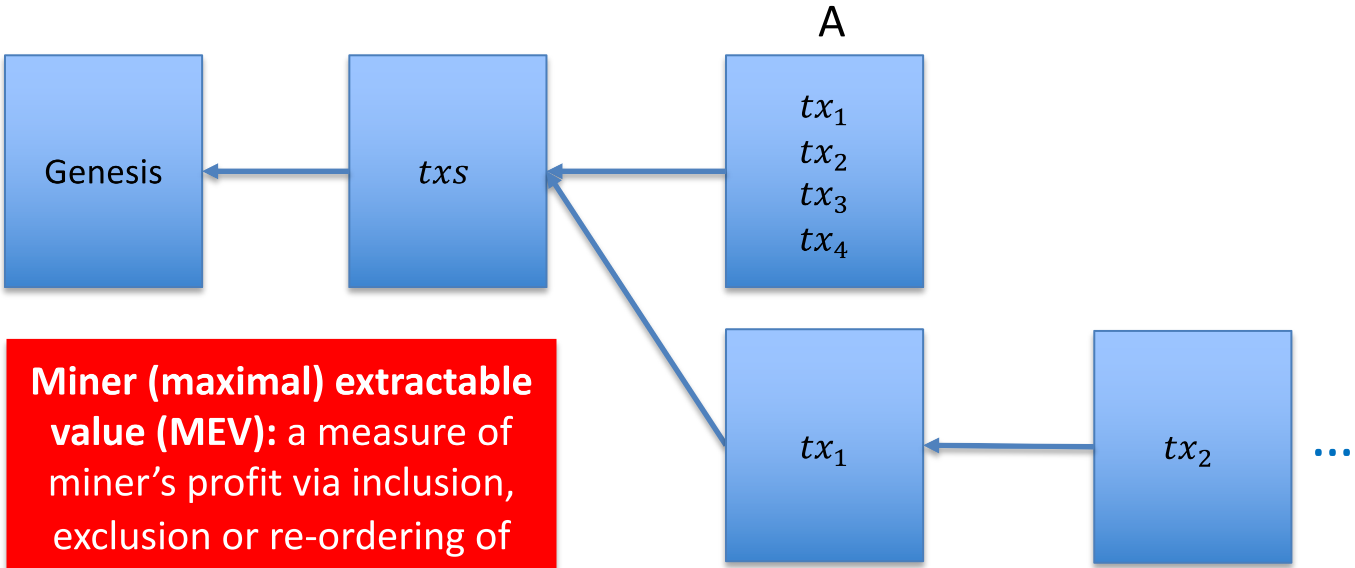
tx_1: 4 BTC

tx_2: 3 BTC

tx_3: 2 BTC

tx_4: 1 BTC

Would $\beta < 1/2$ hold in practice?



Transaction fees
(paid to the miner):
tx_1: 4 BTC
tx_2: 3 BTC
tx_3: 2 BTC
tx_4: 1 BTC

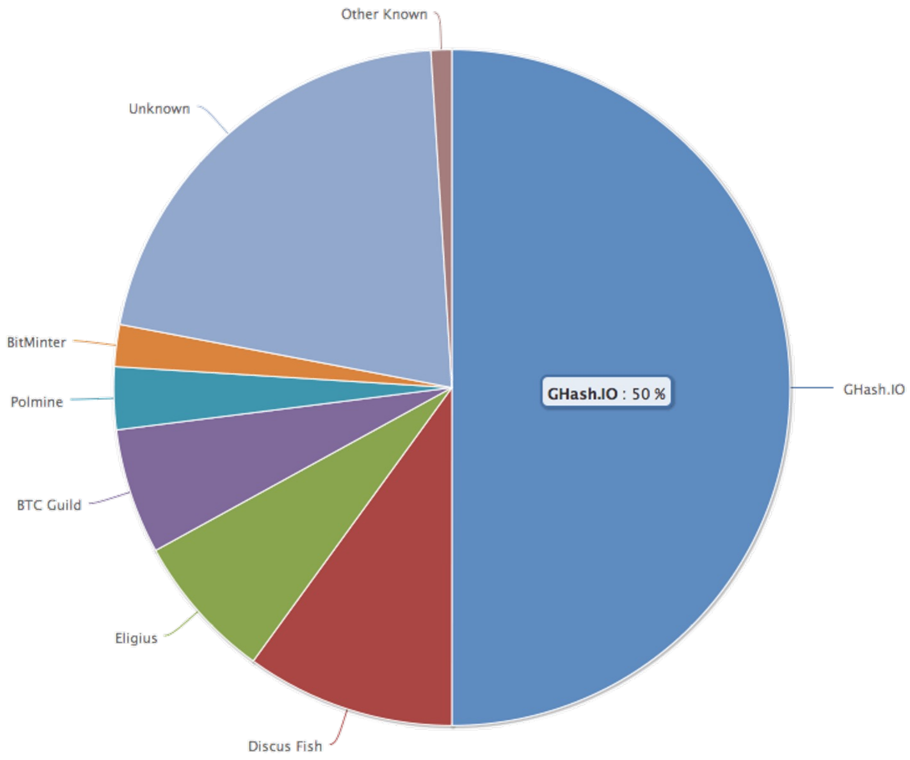
Miner (maximal) extractable value (MEV): a measure of miner's profit via inclusion, exclusion or re-ordering of transactions within its block

- Miners have incentive to violate the protocol!
- Can lead to double spends!

Need to think about incentives!!

MEV gives even more incentive to violate the protocol!!

No Attacks on Bitcoin?



Ghash.IO had >50% in 2014

- Gave up mining power

Why are visible attacks not more frequent?

Miners care about the Bitcoin price?

- Not a valid argument.
- They can ‘short’ the chain for profit!

Might not always be rational to attack.

No guarantees for the future!

Is Bitcoin the Endgame?

Bitcoin provides Sybil resistance and dynamic availability.

Is it the Endgame for consensus?

No!

Bitcoin is secure only under synchrony and loses security during periods of asynchrony.

It *confirms* blocks with an error probability depending on k, i.e., blocks are not finalized.

Energy consumption?

Next lecture: low-energy consensus using proof-of-stake

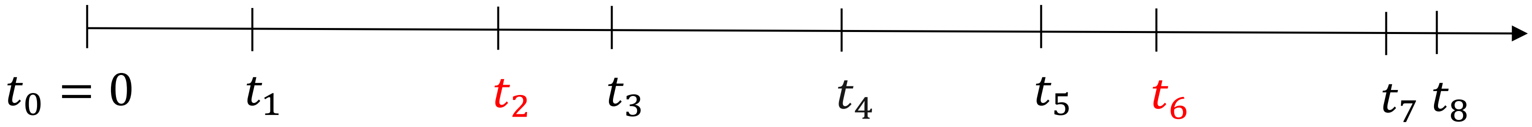
END OF LECTURE

Next lecture: Incentives and Accountability in Consensus

Optional: Security Proof

Loner block:

- An honest block such that no other honest block is mined within Δ time of the loner block.

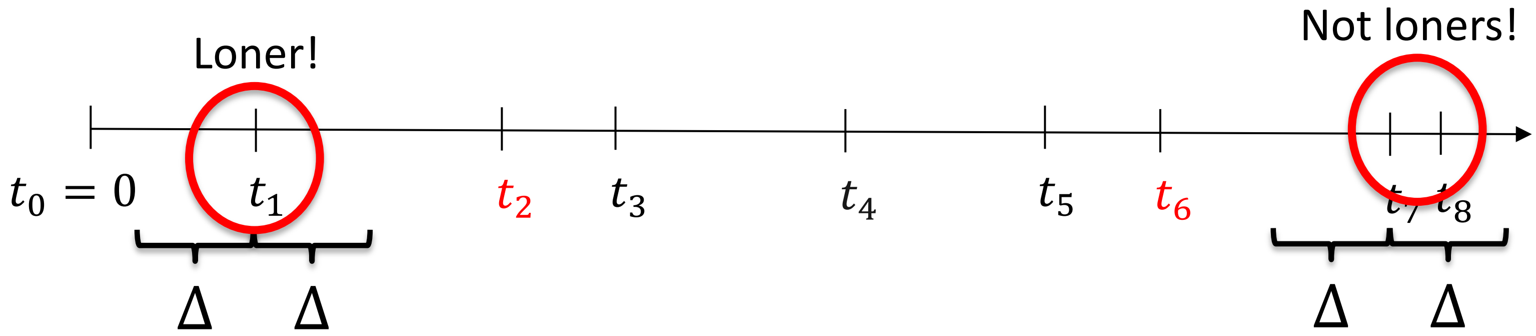


Analysis of Nakamoto Consensus (2019)
The Sleepy Model of Consensus (2016)

Optional: Security Proof

Loner block:

- An honest block such that no other honest block is mined within Δ time of the loner block.



Length of the shortest chain among the longest chains observed by the clients at time t:

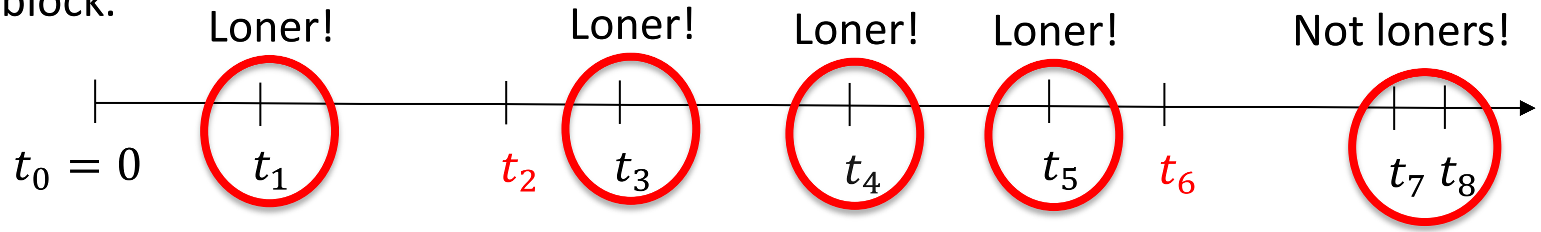
$$L(t)$$

Analysis of Nakamoto Consensus (2019)
The Sleepy Model of Consensus (2016)

Optional: Security Proof

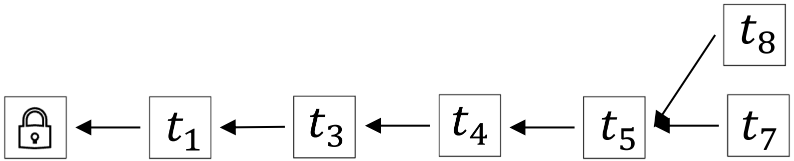
Loner block:

- An honest block such that no other honest block is mined within Δ time of the loner block.



Lemma: For any $s > t$, $L(s) - L(t) \geq$ "number of loners mined in the interval $(t + \Delta, s - \Delta]$".

Proof sketch: Each loner increases the length of the longest chains observed by the clients by one block. For instance;

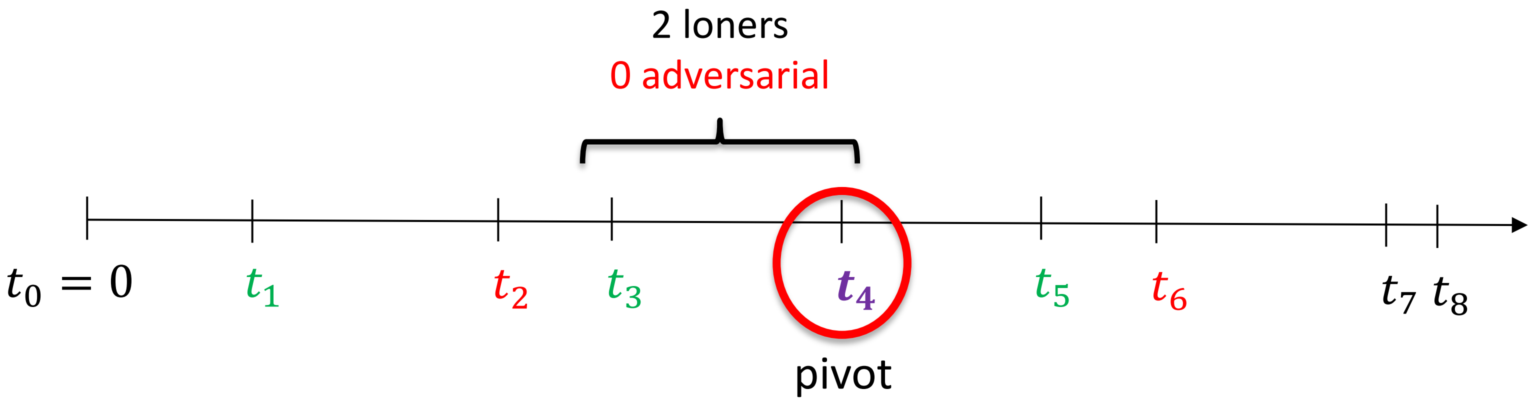


Analysis of Nakamoto Consensus (2019)
The Sleepy Model of Consensus (2016)

Optional: Security Proof

Pivot block:

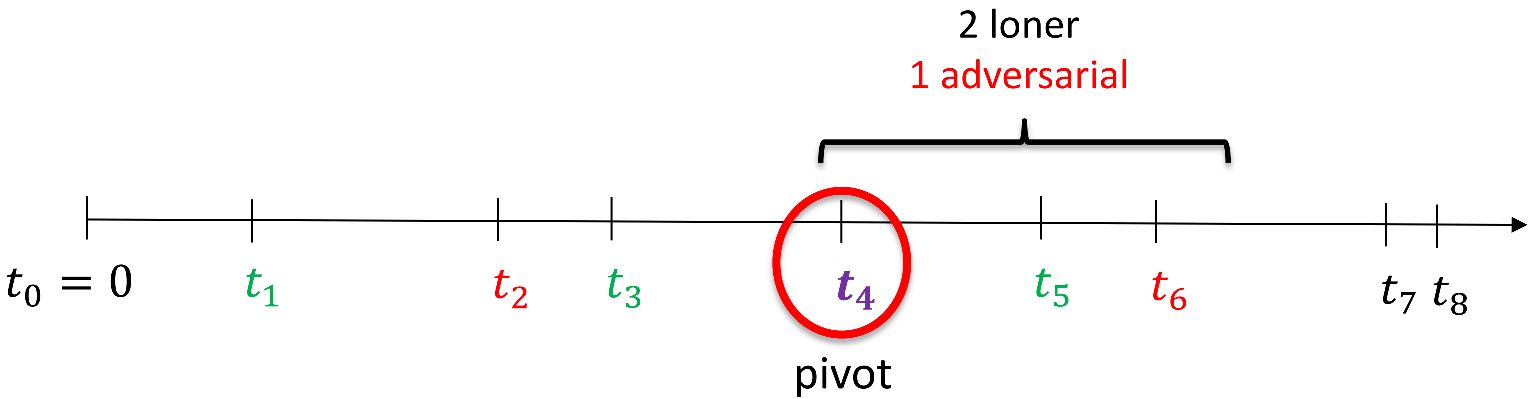
- In any interval covering the mining time of the pivot block, more loner blocks are mined than adversarial blocks.
- Pivot block is a loner.



Optional: Security Proof

Pivot block:

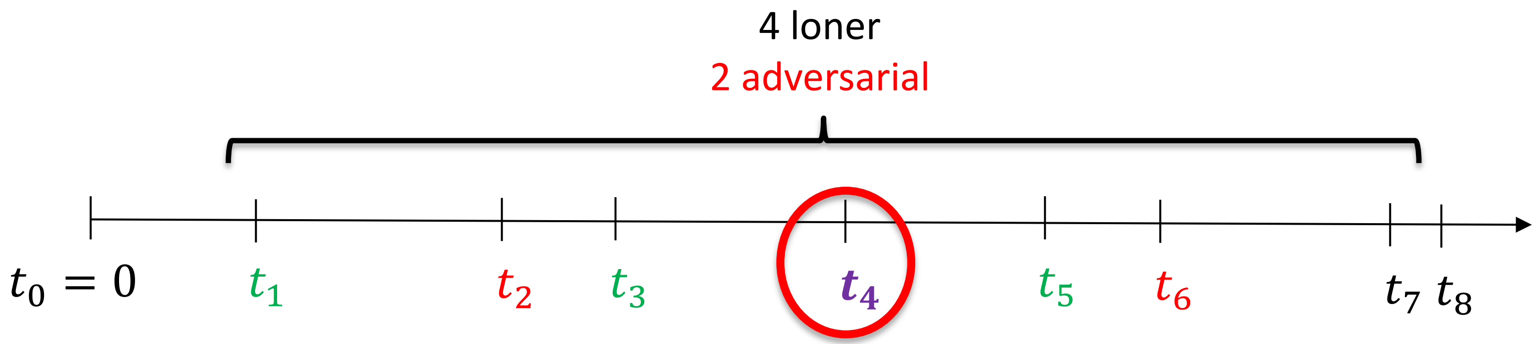
- In any interval covering the mining time of the pivot block, more loner blocks are mined than adversarial blocks.
- Pivot block is a loner.



Optional: Security Proof

Pivot block:

- In any interval covering the mining time of the pivot block, more loner blocks are mined than adversarial blocks.
- Pivot block is a loner.



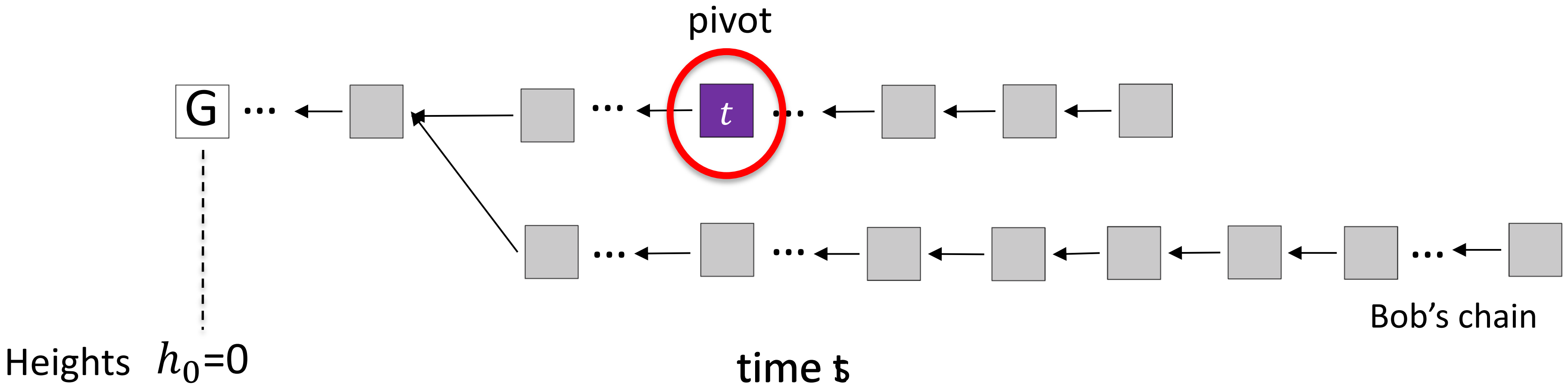
Theorem: If $\beta < 1/2$, there exists a small enough mining rate $\lambda(\Delta, \beta)$ such that any time interval of T have a pivot except with probability $e^{-\Omega(\sqrt{T})}$.

Proof: Probability theory

Optional: Security Proof

Theorem: Suppose a block mined at time t is a pivot. Then, the pivot block is on every (longest) chain held by any client at all times $\geq t$.

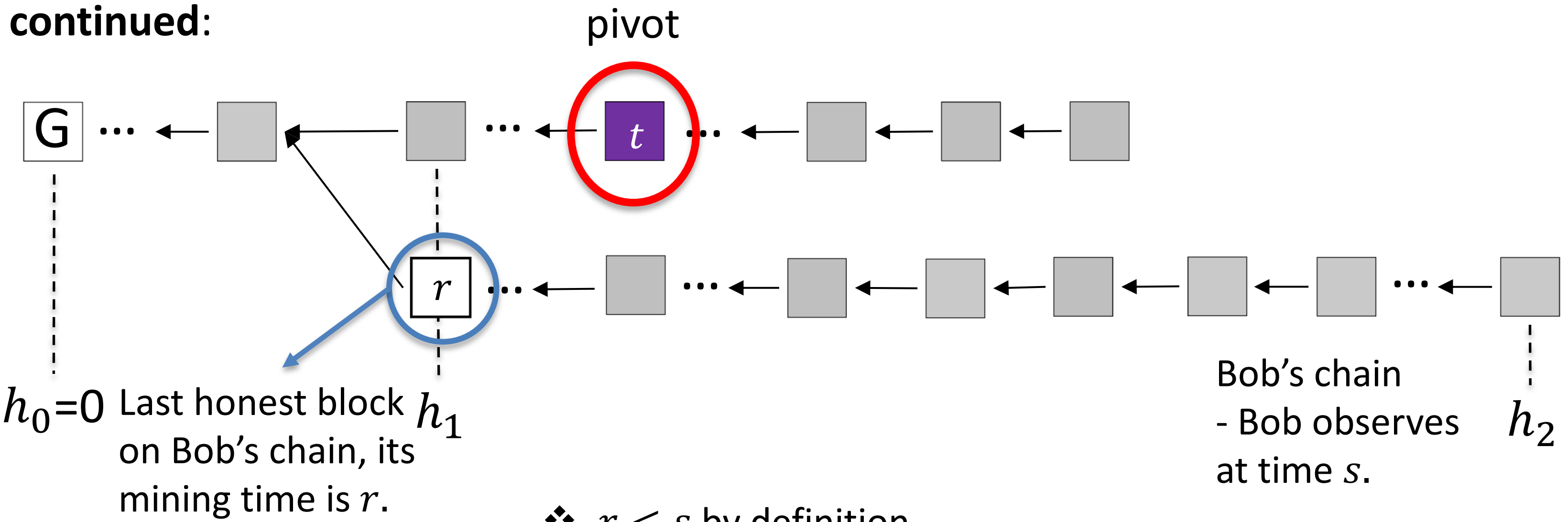
Proof: For contradiction, suppose there exists a minimum time $s \geq t$ such that a client Bob holds a chain conflicting with the pivot block.



Optional: Security Proof

Theorem: If a client holds a chain containing a pivot block, then no client can hold a chain conflicting with the pivot block after the pivot block is mined.

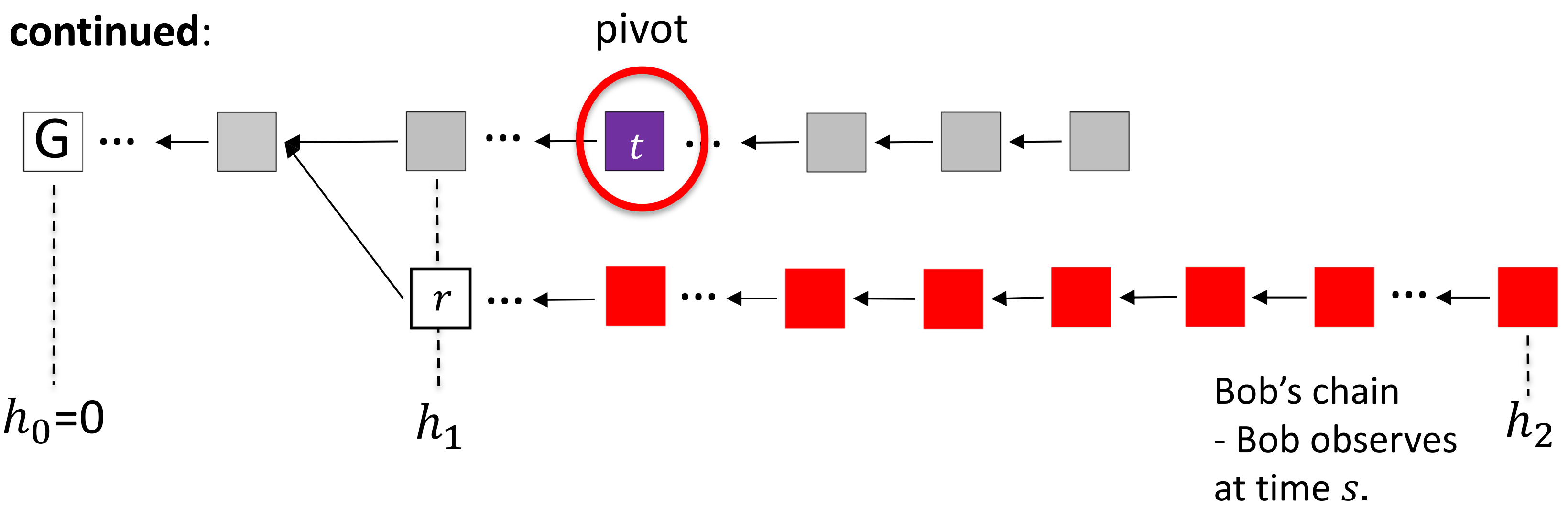
Proof continued:



- $r < s$ by definition.
- $r < t$ because otherwise s is not the first time a conflicting chain is held a client or honest miner.

Optional: Security Proof

Proof continued:



❖ $h_2 - h_1 <$ "blocks mined by the adversary in the interval $(r, s]$"

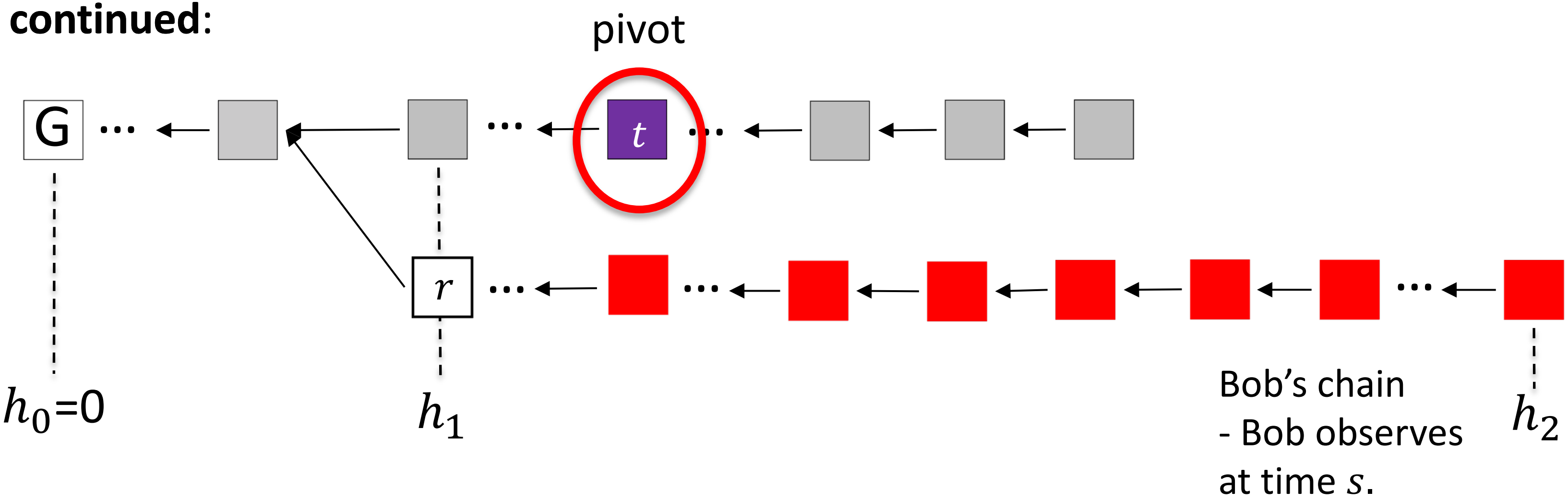
❖ length of the shortest 'longest chain' held by any client at time r, $L(r) \leq h_1$

❖ length of Bob's chain at time $s, h_2 \geq L(s)$

Hence, $h_2 - h_1 \geq L(s) - L(r) \geq$ "number of loners mined in the interval $(r + \Delta, s - \Delta]$" by the lemma.

Optional: Security Proof

Proof continued:



Finally, "blocks mined by the adversary in the interval $(r, s]$" $> h_1 - h_2$

$h_1 - h_2 \geq L(s) - L(r) \geq$ "number of loners mined in the interval $(r + \Delta, s - \Delta]$".

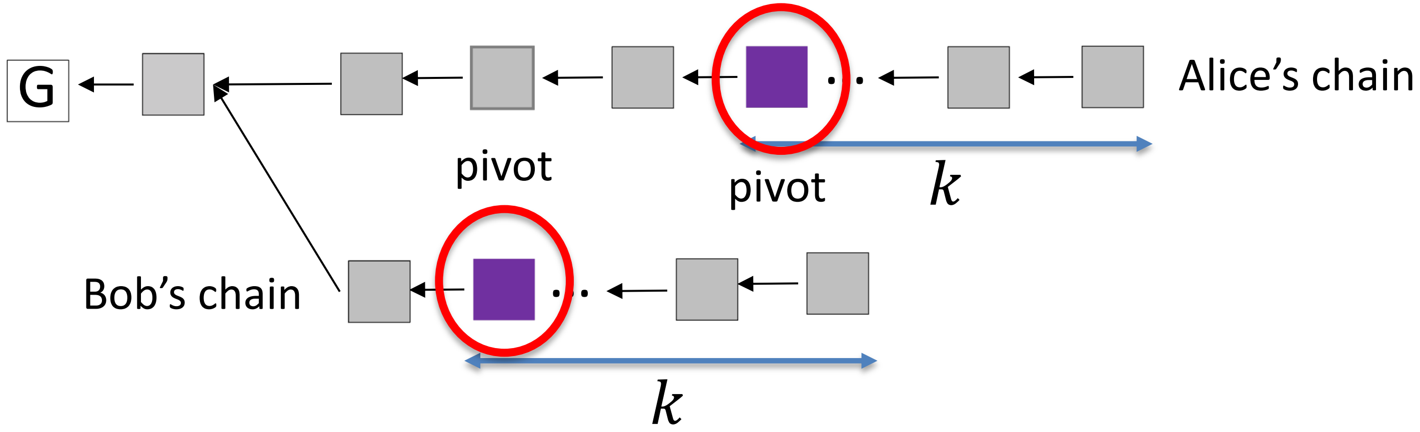
In the interval $(r, s]$ covering t, more adversary blocks are mined than loners!

Contradiction with the definition of pivot!!

Optional: Security Proof

Proof Sketch of Liveness: The pivot is mined by an honest miner and contains all transactions input to the honest miners. Since it is on all chains held by all clients at all times, liveness is satisfied.

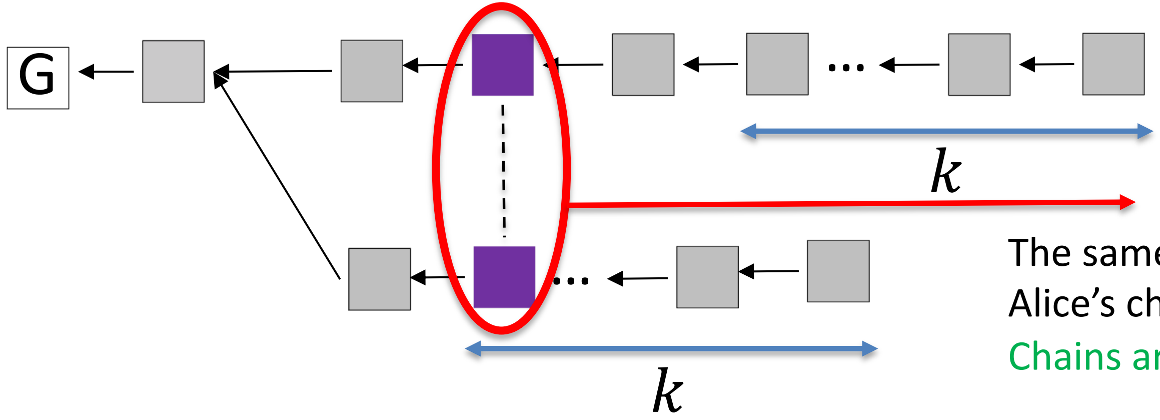
Proof Sketch of Safety: Consider two clients that confirm two chains after chopping off the last k blocks on their chains. One of the last k blocks is a pivot on both chains except with probability $e^{-\Omega(\sqrt{k})}$ (follows from probability theory). Thus,



Optional: Security Proof

Proof Sketch of Liveness: The pivot is mined by an honest miner and contains all transactions input to the honest miners. Since it is on all chains held by all clients at all times, liveness is satisfied.

Proof Sketch of Safety: Consider two clients that confirm two chains after chopping off the last k blocks on their chains. One of the last k blocks is a pivot on both chains except with probability $e^{-\Omega(\sqrt{k})}$ (follows from probability theory). Thus,



Pivots:

The same block appears in Alice's chain by the Theorem.

Chains are consistent!!