Lecture 3: Proof of Work and Nakamoto Consensus

https://web3.princeton.edu/principles-of-blockchains/

Professor Pramod Viswanath
Princeton University

This lecture:
Decentralized Identity, Ownership, Transfer via Proof of Work Mining, Longest Chain Rule

Blockchain with Merkle Trees

Block: Header + Data

Header: Pointer to previous block = hash of the previous block header and Merkle root of data of previous block

Data: information specific to the block

Application: Centralized tamper evident information log with efficient proof of membership of any data entry

Head of the chain being known is enough to find tamper evidence in any internal block

Decentralized Blockchain

Block: Header + Data + Signature

Header: Pointer to previous block = hash of the previous block header and Merkle root of data of previous block

Data: information specific to the block

Signature: one of the users signs the block (header+data)

List of signatures known ahead of time: **permissioned** blockchains

Questions:

1. How is this list known ahead of time?
2. Which user in this list gets to add which block?
3. Who polices this?

This is the topic of this lecture

Distributed Consensus

Question: Who maintains the ledger of transactions?

Distributed Consensus

- Interactive Protocol
- Allows distributed non-trusting nodes to come to agreement
- Traditional area of computer science (Byzantine Fault Tolerance)

Bitcoin's consensus protocol is vastly different

- decentralized identity (permissionless setting)
- less pessimistic network assumptions

Decentralized Identity

Public keys are used as **identity**

Single entity can create vast number of identities

- Sybil
- Cannot do majority or super-majority voting

Network Assumption

Any node can **broadcast** to **all** nodes into the network

fully connected network

Every broadcast message **reaches every** node

albeit with some delay

Bitcoin: ten minutes

This is the focus of Lecture 4

Leader Election: Oracle

Time is organized into **slots**

Oracle selects one of the nodes (public identities)

- random
- everyone can verify the unique *winner*

The selected node is the proposer in that slot

- constitutes a block with transactions
- validates transactions
- includes hash pointer to previous block
- signs the block

Proof of Work

Practical method to simulate the Oracle

Mining

cryptographic hash function creates computational puzzle

Hash(nonce, block-hash) < Threshold

nonce is the proof of work

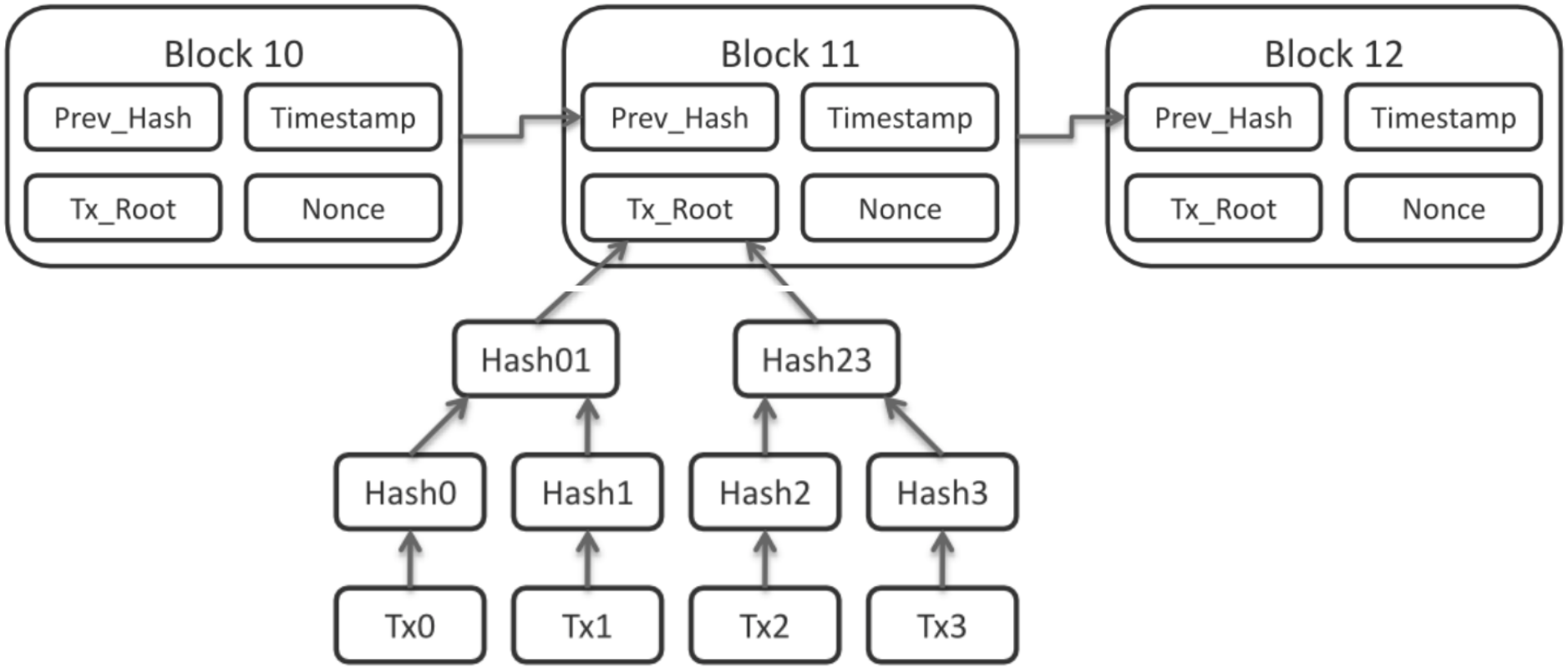
include nonce inside the block

Threshold

chosen such that a block is mined successfully on average once in 10 minutes

a successfully mined block will be broadcast to all nodes in the network

Block Constituents



Properties of Proof of Work Mining

1. Random miner selected at each time

2. Independent randomness across time and across miners

3. Probability of successful mining proportional to fraction of total hash power

4. Sybil resistance

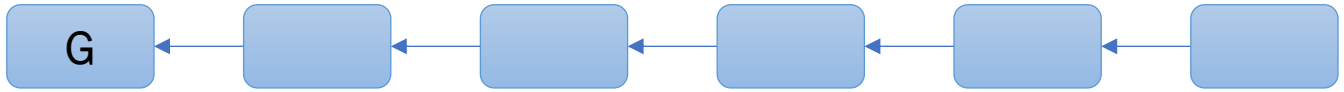
5. Spam resistance

6. Tamper proof – even by the proposer!

Longest Chain Protocol

Where should the mined block hash-point to?

The latest block



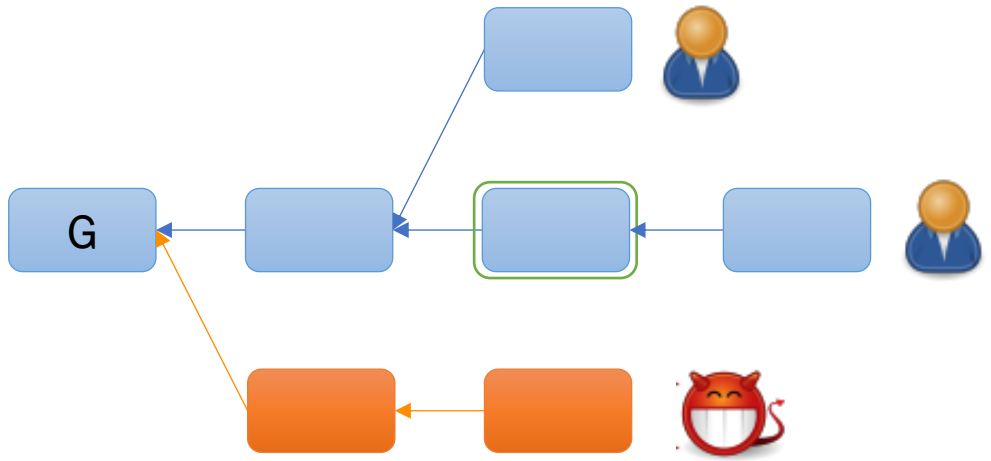
Longest Chain Protocol

Where should the mined block hash-point to?

However, blockchain may have **forks**

because of network delays

because of adversarial action



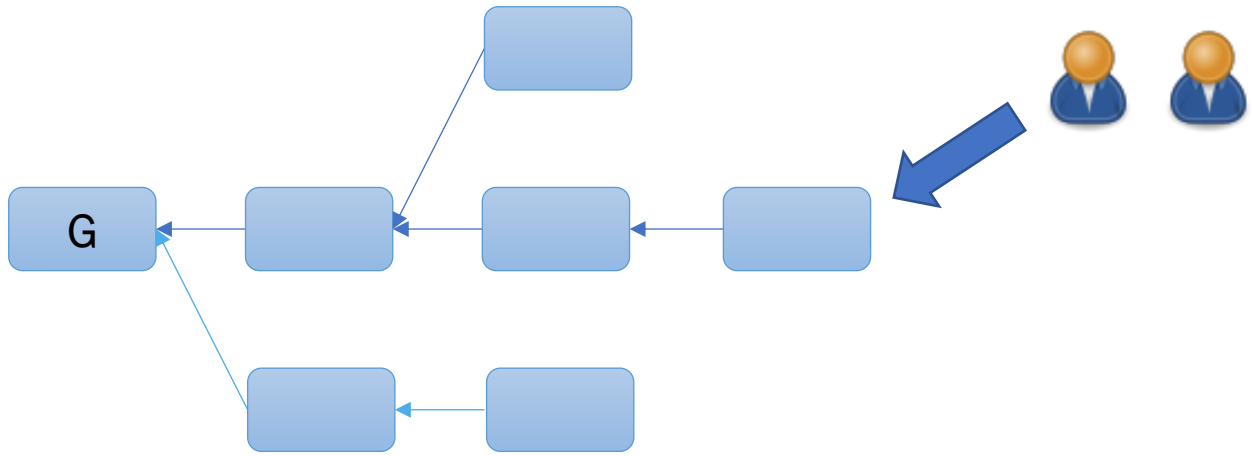
Longest Chain Protocol

Where should the mined block hash-point to?

Blockchain may have **forks**

because of network delays

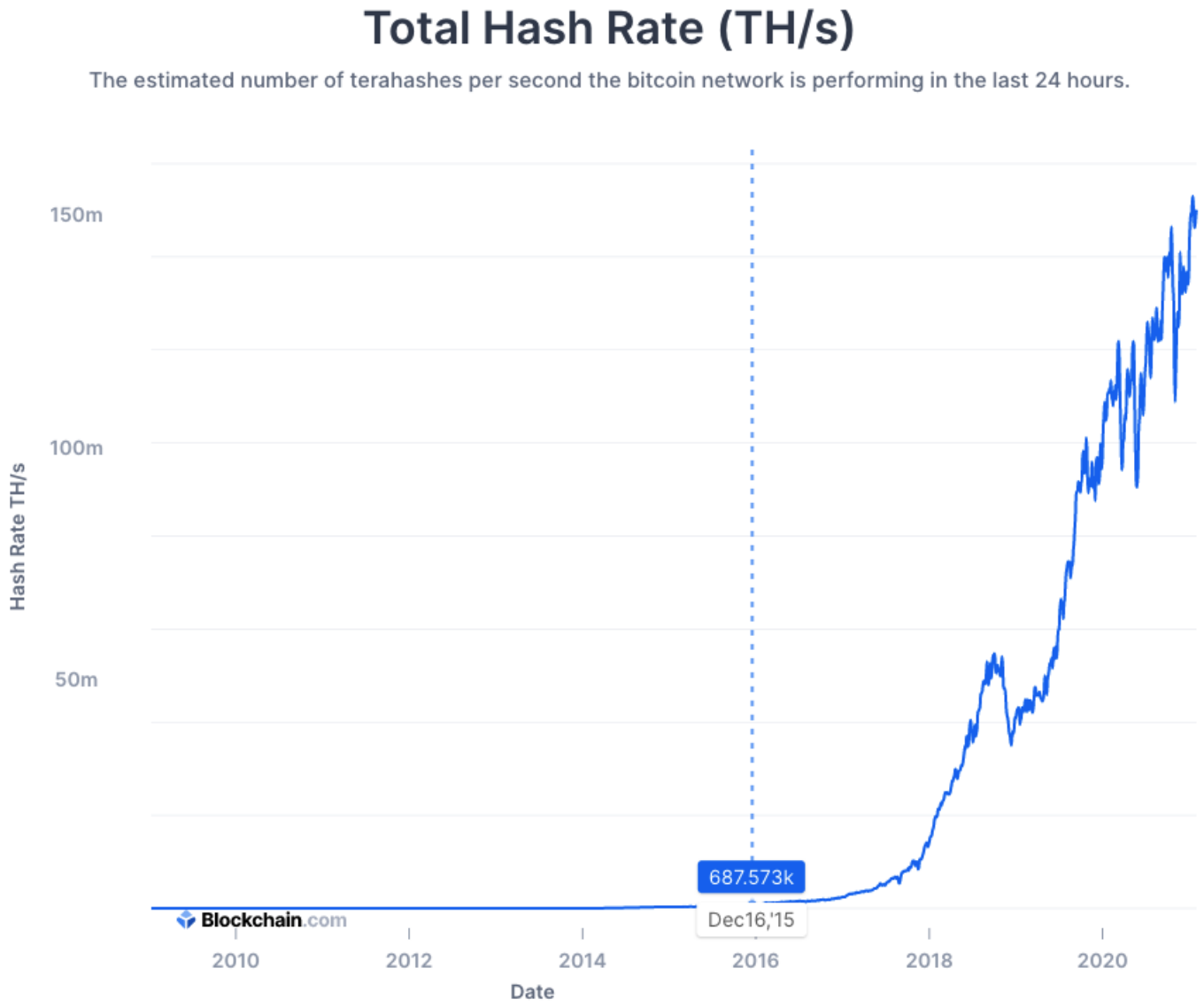
because of adversarial action



Longest chain protocol

attach the block to the leaf of the longest chain in the block tree

Why Variable Difficulty



Block Difficulty

Example: in September 2022 the mining target or threshold (in hexadecimal) is:

0x 00000000000000008c8940000000000000000000000000000000000

19 leading zeros

The hash of any valid block must be below this value ~ 8/16 . 16^{-19} = 2^{-77}

Difficulty of a block:

Block_difficulty = 1/mining_target

Bitcoin Rule

(a) The mining difficulty changes every 2016 blocks

next_difficulty = (previous_difficulty * 2016 * 10 minutes) / (time to mine last 2016 blocks)

(b) Adopt the heaviest chain instead of the longest chain

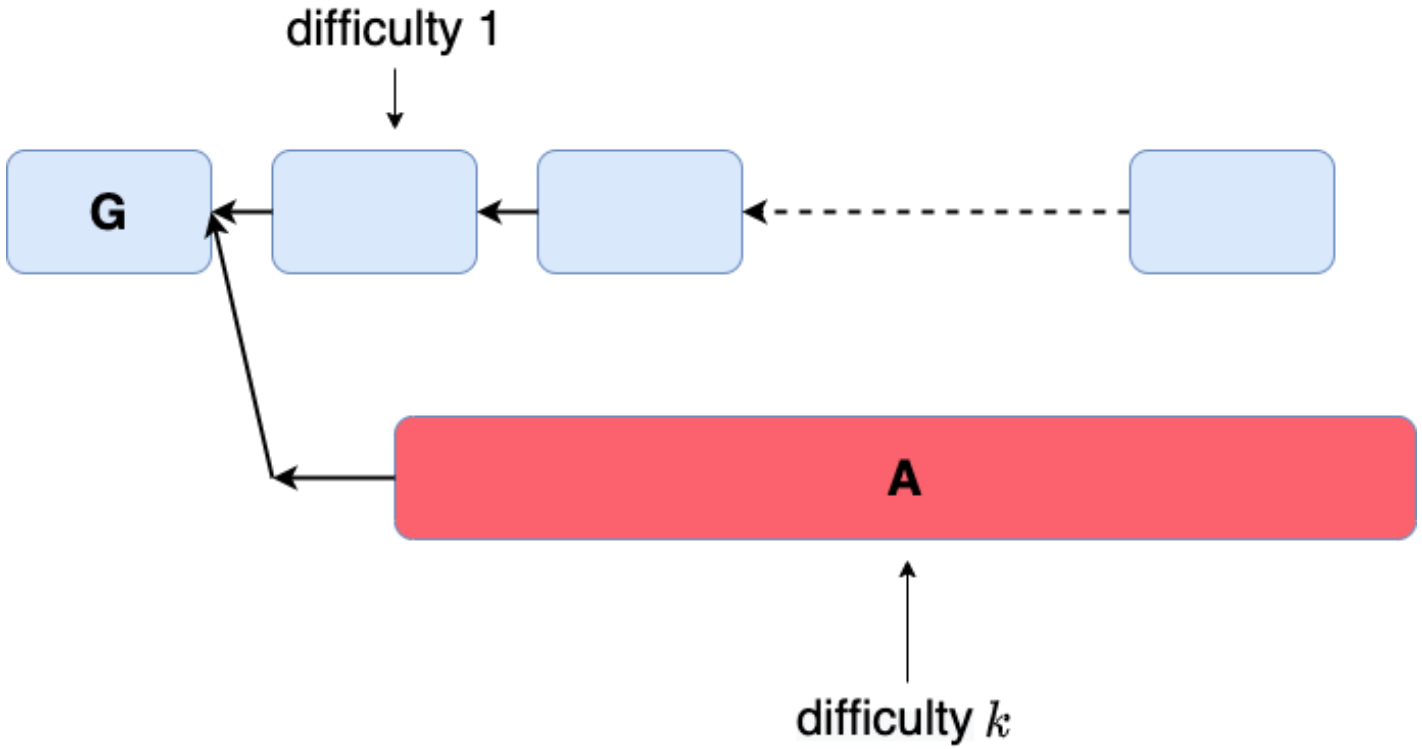
chain_difficulty = sum of block_difficulty

(c) Allow the difficulty to be adjusted only mildly every epoch

¼ < next_difficulty/previous_difficulty < 4

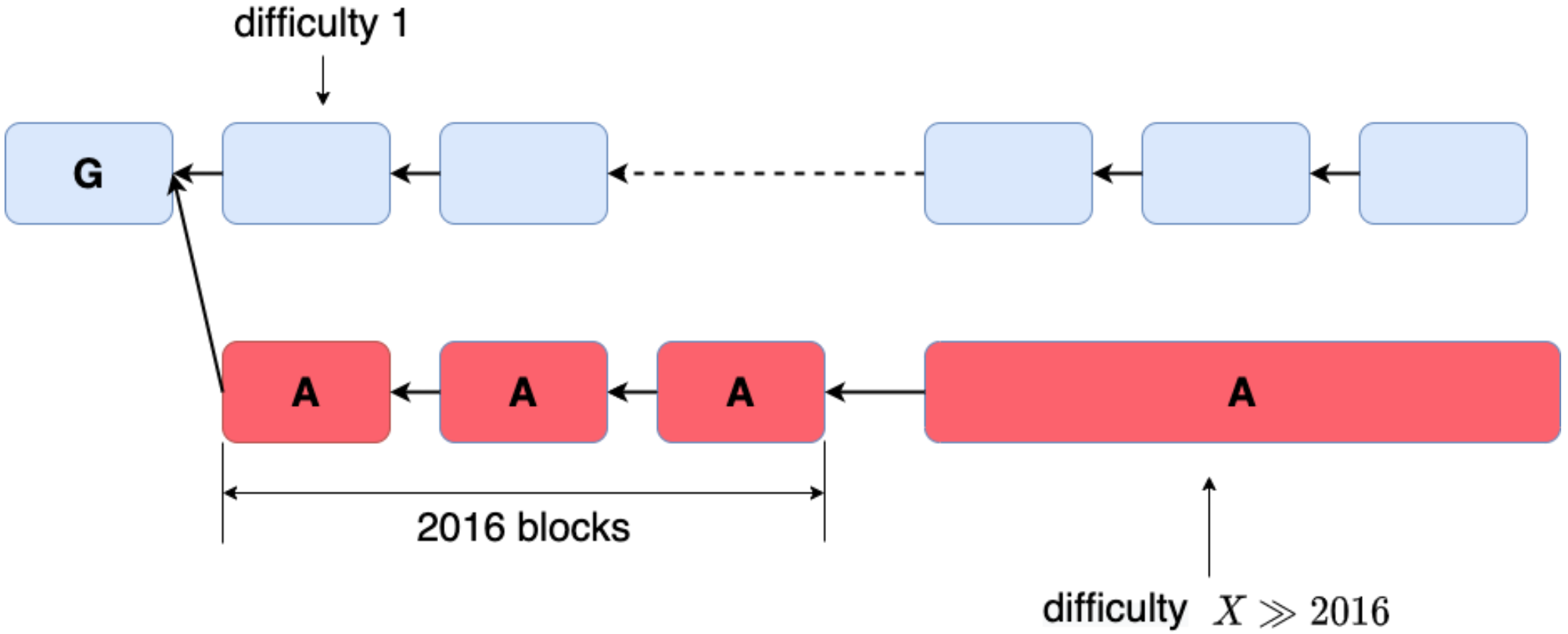
Alternate Bitcoin Rule (Only (b))

Let the miners choose their own difficulty and then use (b) the heaviest chain rule.

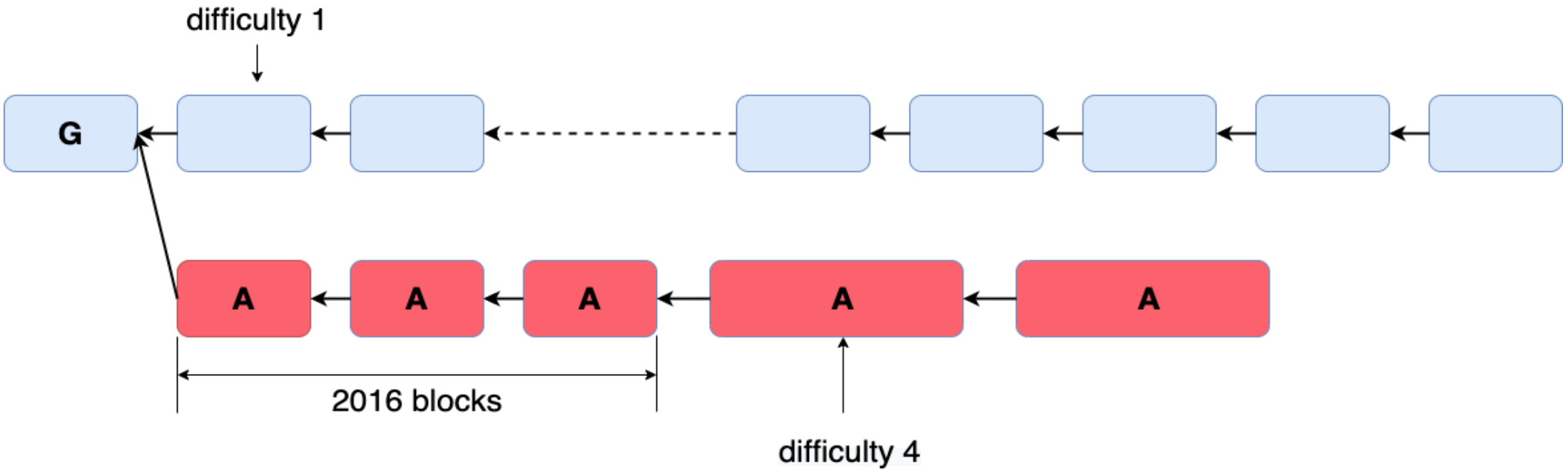


Alternate Bitcoin Rule ((a) + (b))

Difficulty rising attack



Bitcoin Rule ((a) + (b) + (c))



Security Analysis: Private Attack

Adversary can point its block to an older part of the chain

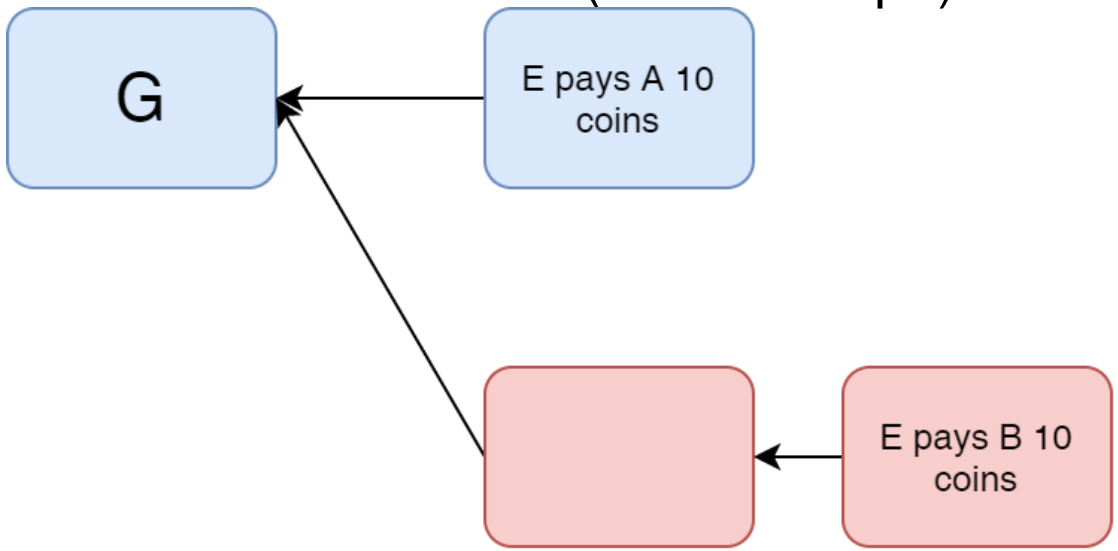
Duplicate transaction inserted

Plausible Deniability

network latency

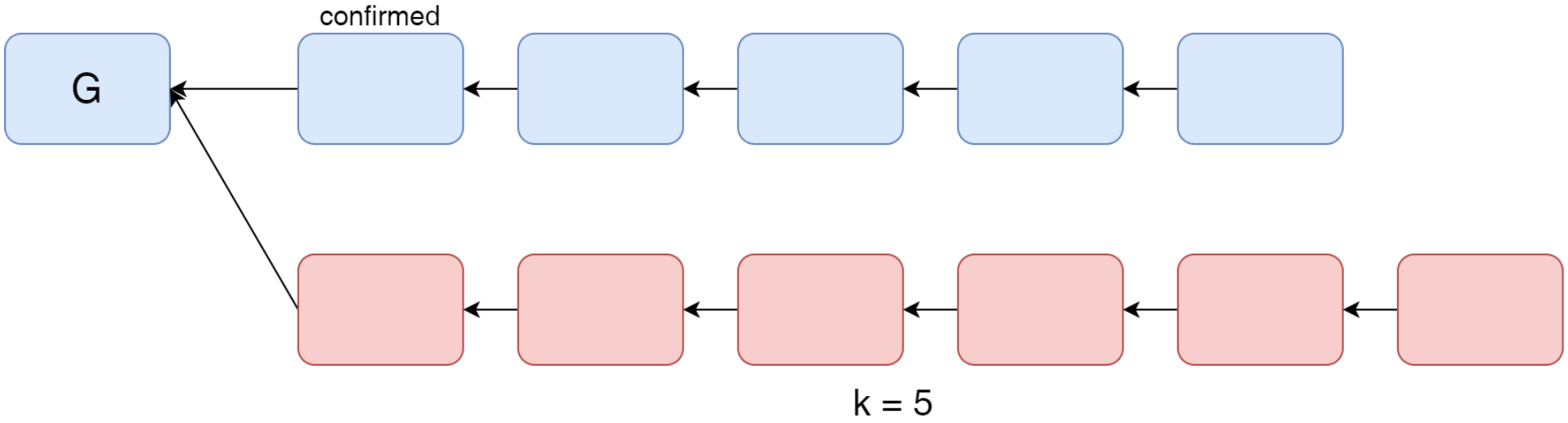
an offline user will not know which block came earlier

blocks have no wall clock reference (time stamps).



Security Analysis: k Deep Confirmation Rule

- A block is confirmed if it is buried k-deep in the longest chain
- An attacker would need more than k blocks to double spend



Security vs Latency with Private Attack