Lecture 4

Bitcoin Consensus

Bitcoin consensus: theory & practice

- Bitcoin consensus: initially, seemed to work better in practice than in theory
- Theory has been steadily catching up to explain why Bitcoin consensus works [e.g., Garay-Kiayias-Leonardos'15,Pass-Shelat-Shi'17,Garay-Kiayias-Leonardos'17,…]
- Theory is important, can help predict unforeseen attacks

Some things Bitcoin does differently

Introduces incentives

- Possible only because it’s a currency!

Embraces randomness

- Does away with the notion of a specific end-point
- Consensus happens over long time scales — about 1 hour

Consensus without identity: the blockchain

Why identity?

Pragmatic: some protocols need node IDs

Security: assume less than 50% malicious

Why don’t Bitcoin nodes have identities?

Identity is hard in a P2P system — Sybil attack

Pseudonymity is a goal of Bitcoin

Weaker assumption: select random node

Analogy: lottery or raffle

When tracking & verifying identities is hard, we give people tokens, tickets, etc.

Now we can pick a random ID & select that node

Key idea: implicit consensus

In each round, random node is picked

This node proposes the next block in the chain

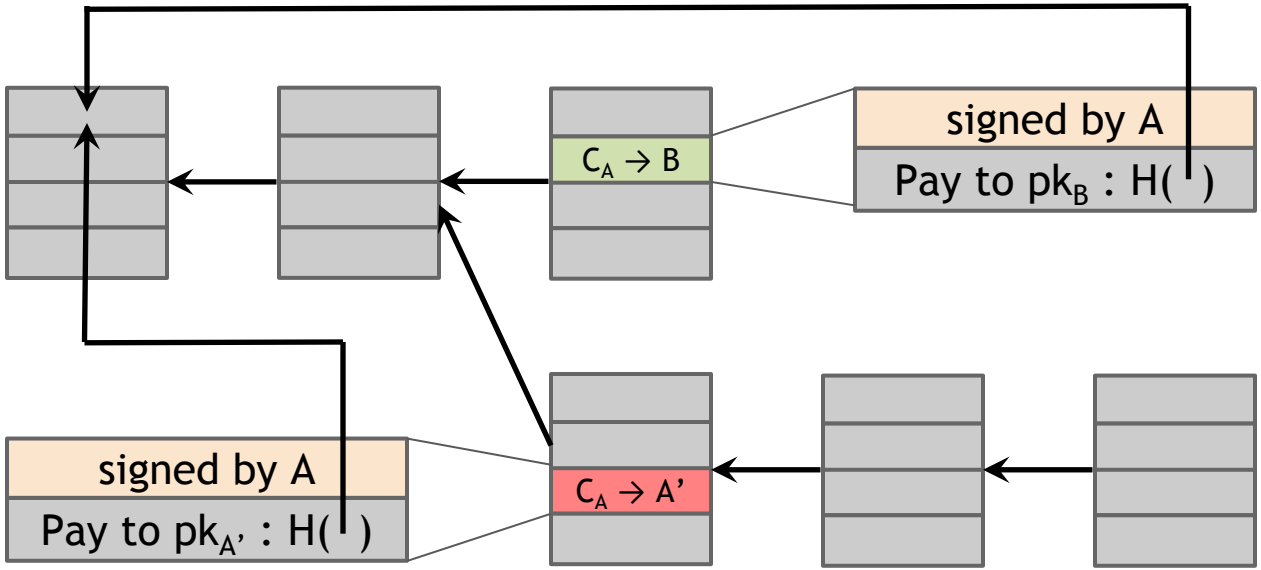
Other nodes implicitly accept/reject this block
- by either extending it
- or ignoring it and extending chain from earlier block

Every block contains hash of the block it extends

Consensus algorithm (simplified)

1. New transactions are broadcast to all nodes
2. Each node collects new transactions into a block
3. In each round a random node gets to broadcast its block
4. Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)
5. Nodes express their acceptance of the block by including its hash in the next block they create

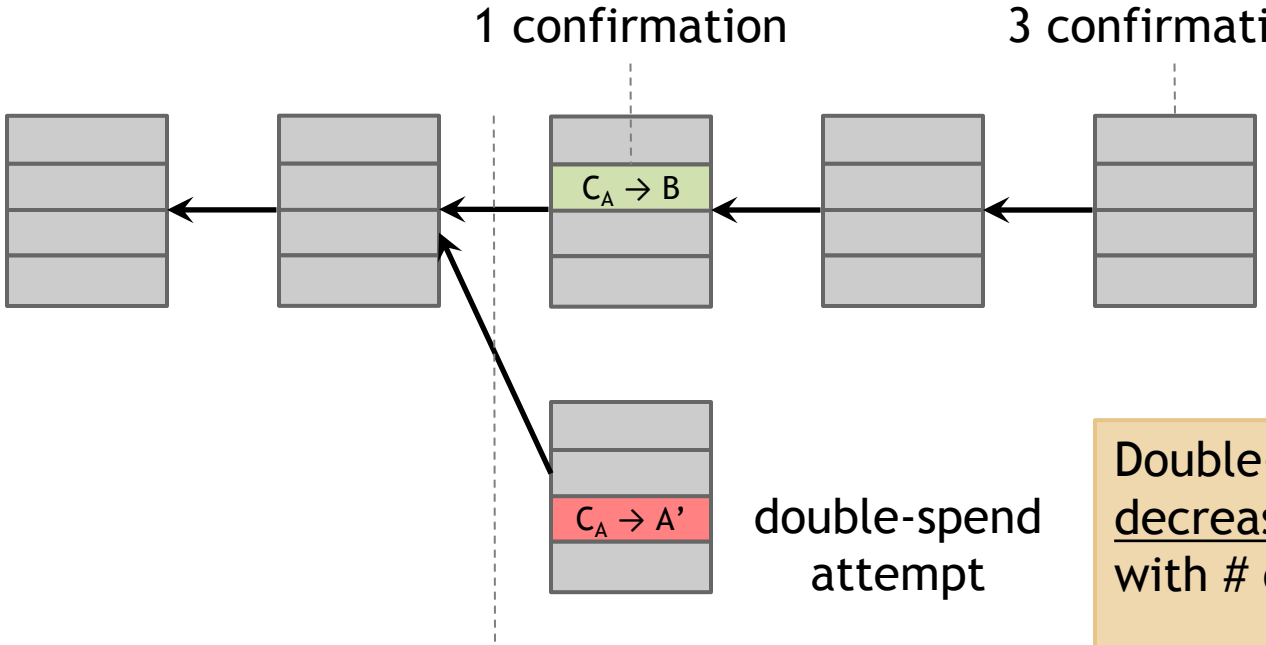
What can a malicious node do?



Double-spending attack

Honest nodes will extend the longest valid branch

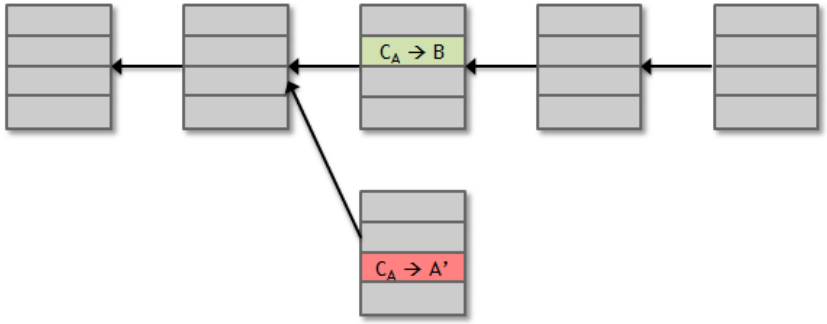
From Bob the merchant's point of view



Double-spend probability decreases exponentially with # of confirmations

Most common heuristic:
6 confirmations

Recap



Protection against invalid transactions is cryptographic, but enforced by consensus

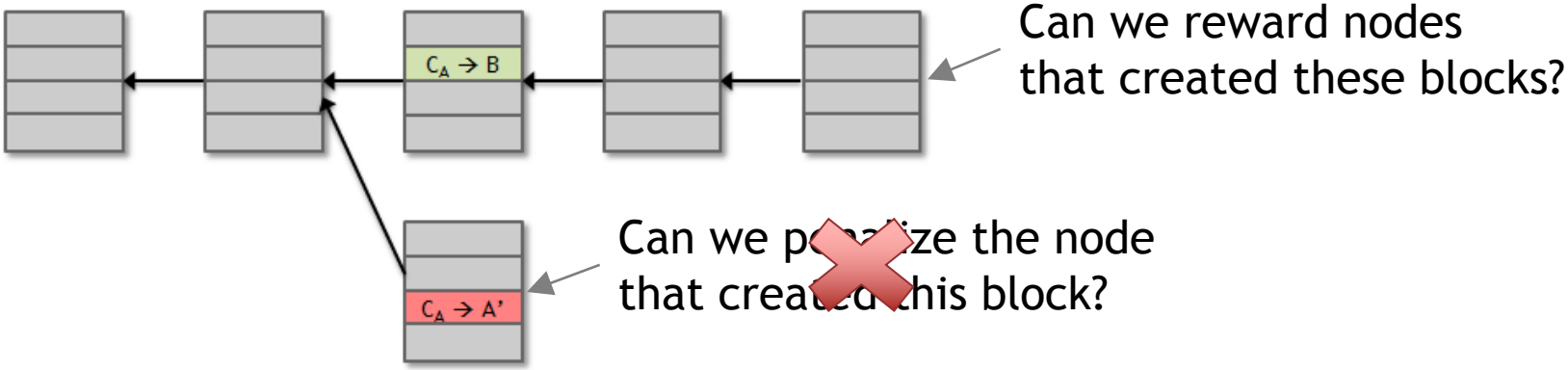
Protection against double-spending is purely by consensus

You're never 100% sure a transaction is in consensus branch. Guarantee is probabilistic

Incentives and proof of work

Assumption of honesty is problematic

Can we give nodes incentives for behaving honestly?



Everything so far is just a distributed consensus protocol
But now we utilize the fact that the currency has value

Incentive 1: block reward

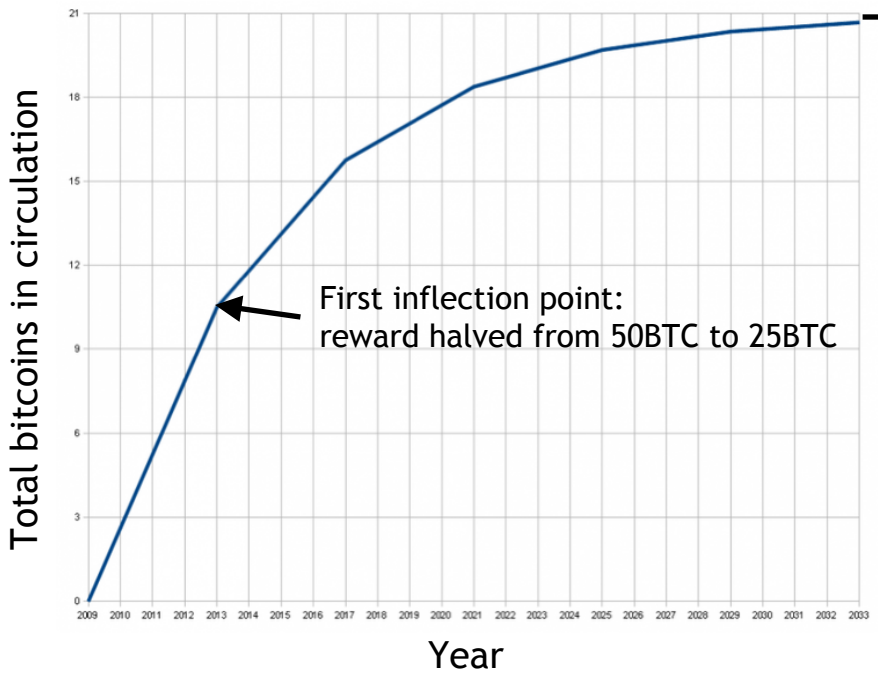
Creator of block gets to

- include special coin-creation transaction in the block
- choose recipient address of this transaction

Value is fixed: currently 12.5 BTC, halves every 4 years

Block creator gets to “collect” the reward only if the block ends up on long-term consensus branch!

There’s a finite supply of bitcoins



Total supply: 21 million

Block reward is how new bitcoins are created

Runs out in 2040. No new bitcoins unless rules change

Incentive 2: transaction fees

Creator of transaction can choose to make output value less than input value

Remainder is a transaction fee and goes to block creator

Purely voluntary, like a tip

Remaining problems

1. How to pick a random node?

1. How to avoid a free-for-all due to rewards?

1. How to prevent Sybil attacks?

Proof of work

To approximate selecting a random node:
select nodes in proportion to a resource
that no one can monopolize (we hope)

- In proportion to computing power: proof-of-work
- In proportion to ownership: proof-of-stake

Equivalent views of proof of work

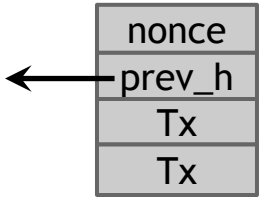
1. Select nodes in proportion to computing power

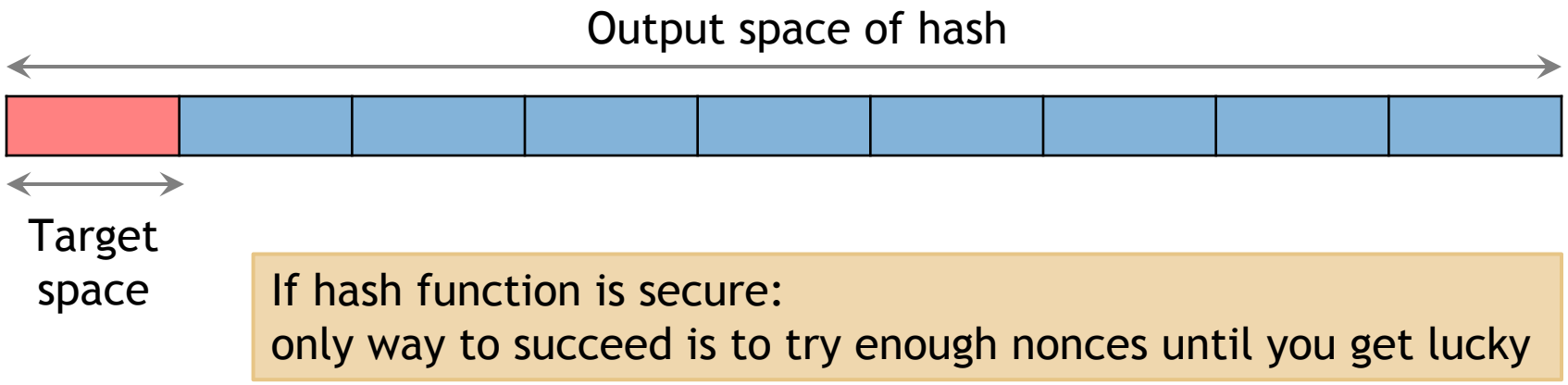
1. Let nodes compete for right to create block

1. Make it moderately hard to create new identities

Hash puzzles

To create block, find nonce s.t.
H(nonce ‖ prev_hash ‖ tx ‖ … ‖ tx) is very small





If hash function is secure:
only way to succeed is to try enough nonces until you get lucky

PoW property 1: difficult to compute

As of Aug 2014: about 10^{20} hashes/block

Only some nodes bother to compete — miners

PoW property 2: parameterizable cost

Nodes automatically re-calculate the target every two weeks

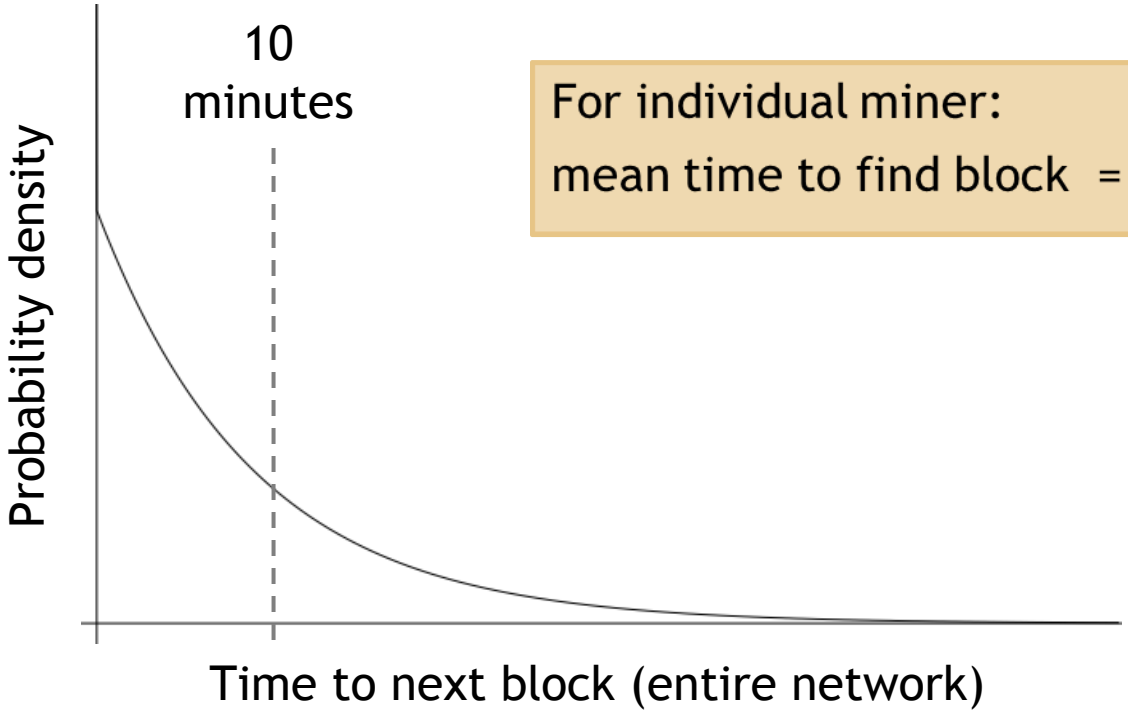
Goal: average time between blocks = 10 minutes

Prob (Alice wins next block) =
fraction of global hash power she controls

Key security assumption

Attacks infeasible if majority of miners weighted by hash power follow the protocol

Solving hash puzzles is probabilistic



For individual miner:

mean time to find block $= \frac{\text{10 minutes}}{\text{fraction of hash power}}$

PoW property 3: trivial to verify

Nonce must be published as part of block

Other miners simply verify that
H(nonce || prev_hash || tx || … || tx) < target

Mining economics

If mining reward (block reward + Tx fees)	>	hardware + electricity cost	→	Profit

Complications:

- fixed vs. variable costs
- reward depends on global hash rate

Putting it all together

Recap

Identities

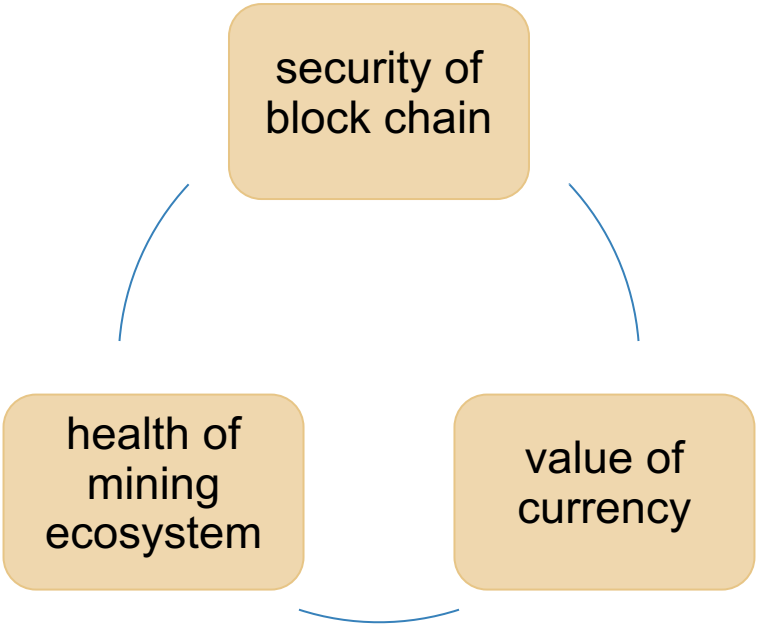
Transactions

P2P network

Block chain & consensus

Hash puzzles & mining

Bitcoin is bootstrapped



What can a “51% attacker” do?

Steal coins from existing address? ✗

Suppress some transactions?

- From the block chain ✓
- From the P2P network ✗

Change the block reward? ✗

Destroy confidence in Bitcoin? ✓ ✓