Blockchains & Cryptocurrencies

Scaling II (Eth 2)

Instructor: Matthew Green
Johns Hopkins University - Fall 2020

Housekeeping

- Project presentations
  - Dec 2 + Dec 7: assigned slots will be sent out soon
  - No class Dec 9 (stupid JHU schedule)
- Final exam, take-home: will be given out December 16 or 17
  - Exact dates to come
  - Due December 18, 5pm (end of our scheduled slot)
  - Similar in structure to past exams (Gradescope etc.)

News?

Ethereum 2.0 Deposit Threshold Met: Proof-of-Stake 'Beacon' Chain Starts in 7 Days

Review stuff (from last time)

The problem

- Bitcoin transaction rate: 5-7 tx/sec
  - Bounded by block size (Segwit helps), TX size
  - All transactions must be globally verified, stored
- Ethereum: 15 transactions per second <u>if they're small</u>
- Visa: 24,000/sec peak (150M/day globally)
- WeChat 256,000/sec peak

Ethereum state channels

- In principle we can replicate Bitcoin's LN payment channels on Ethereum. Basic ideas are similar.
  - Initial "funding transaction" that locks up funds between two parties
  - Subsequent "update" transactions that change the balance of funds between two parties (not posted to chain)
  - Final "closure" transaction that goes on-chain + a dispute resolution procedure

Ethereum state channels cont'd

- The problem is that this idea works <u>only for specific cases where two users are affected by misbehavior:</u>
  - Imagine A & B have a state channel, and they're both colluding to do bad things
  - Can this hurt C?

Ethereum state channels cont'd

- The problem is that this idea works <u>only for specific cases where two users are affected by misbehavior:</u>
  - Imagine A & B have a state channel, and they're both colluding to do bad things while off-chain
  - Can this hurt C?
  - In Bitcoin payment channels, the answer is: **NO**. Payment channels only adjust the balance <u>between</u> A, B. Since C isn't involved, they have no "skin in the game".
  - But ETH contract state <u>might</u> matter to other people!!

Ethereum state channels cont’d

- Let’s assume that contract state updates (transactions) matter to <u>many parties</u>, i.e., not just Alice and Bob
  - Can we think of an example of such a contract?

Ethereum state channels cont'd

- Let's assume that contract state updates (transactions) matter to <u>many parties</u>, i.e., not just Alice and Bob
  - Can we think of an example of such a contract?
- Let's say we want to do a sequence "off chain" executions (transactions) of the smart contract, and not put them all on the chain to be verified by consensus nodes
  - How could we do this?

Two techniques

- Both go by the name "rollup": signifies that the idea is to take a chain of many sequential transactions and "compress" them into a small value that can be verified on chain
  - **Optimistic rollup:** Let's do all the transactions off-chain without verifying them, and publish them to the world in the hope that they're valid. If any transaction turns out to be *invalid*, we provide an incentivized mechanism to post a "proof" of invalidity.
  - **ZK rollup**: Let's use the magic of zero-knowledge (VC) to <u>prove</u> that we verified all the transactions, and produce a small proof.

Optimistic rollup (one concept)

- Imagine we have a series of transactions and we want to prove they are all valid (in a short on-chain transaction)
  - We designate a third party ("bonded aggregator") who locks up some currency (ETH) to pay for misbehavior
  - They collect all of the transactions people sent them, and execute the transaction **off chain**
  - For each TX they compute a Merkle tree over the TXes, and publish them too (Merkle root + transactions)
  - Finally they publish a single TX to the Ethereum chain, containing the Merkle root and some extra logic (—>)

Optimistic rollup (one concept)

- Imagine we have a series of transactions and we want to prove they are all valid (in a short on-chain transaction)
  - This extra logic supports "fraud proofs" of two types:
    - If anyone can provide a proof that <u>a single transaction</u> in the chain is **invalid**, they can "punish" the aggregator
    - If anyone can provide a receipt that says their own TX was included in the chain, but it isn't in the rollup, then they can "punish" the aggregator
  - Punishment means "take some or all of the bond"

What guarantees do we get?

- Imagine that an aggregator is malicious
  - Example: they want to inject invalid transactions into an ERC20 contract that gives them money they shouldn't have
  - Benefit of the attack (to malicious aggregator) is potentially quite high! A single invalid TX can be worth millions USD
  - Downside is potential for getting caught, and being "slashed" (punished)

What guarantees do we get?

- Imagine that an aggregator is malicious
  - Key requirement is that the transactions in the rollup chain are published widely enough that some honest node will discover malicious behavior
  - Might need incentive mechanisms to make sure people validate the whole chain. But who keeps the validators honest?
  - How does this work in Ethereum L1 (on chain?)

ZK Rollup

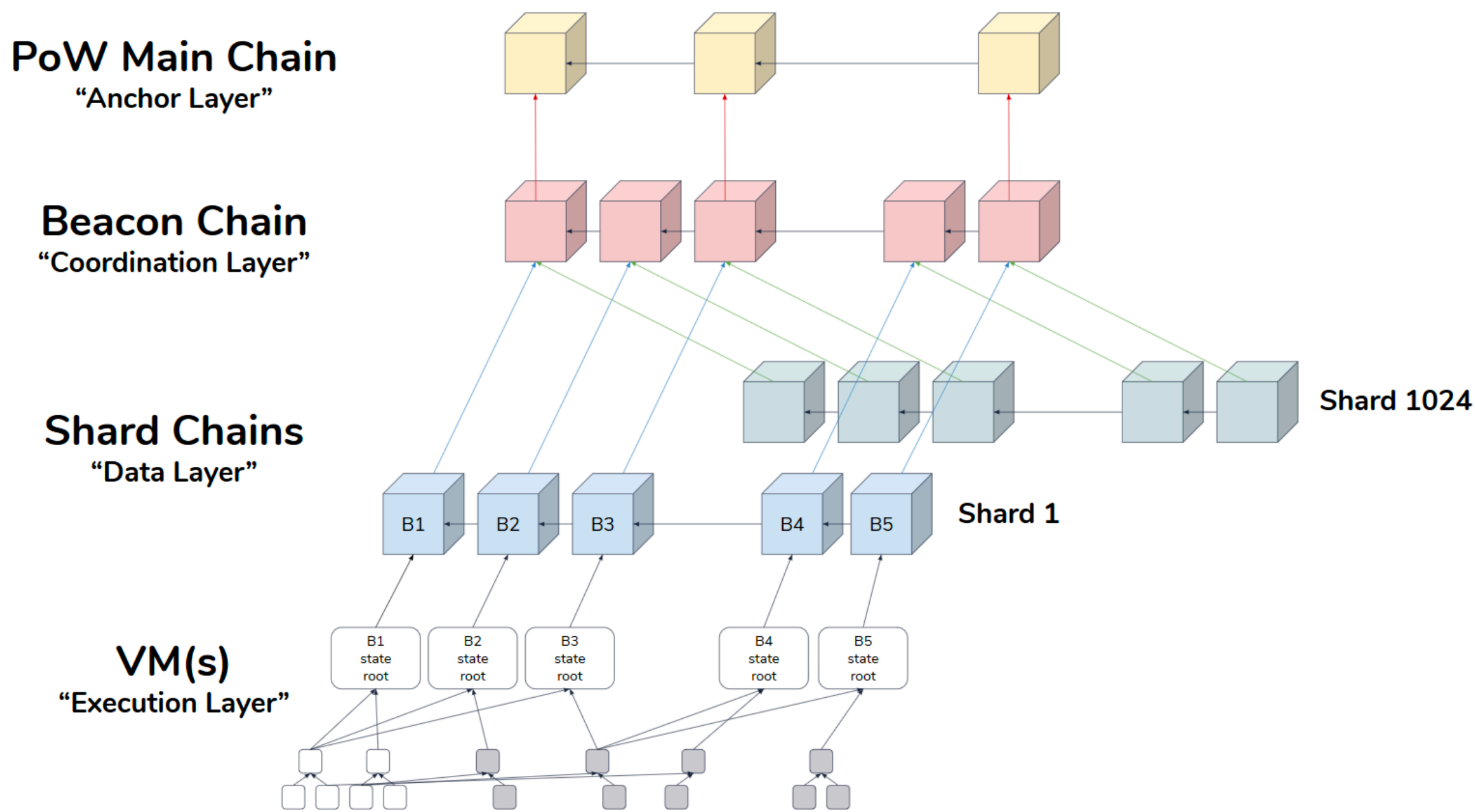
- A different property, uses the magic of "verifiable computation", and cryptographic "proving technology"
  - Basic assumption is that we have a "proving system" that can take the inputs and outputs of some program, and produce a **short** proof that the program has been executed correctly
  - There are many older and emerging technologies for this: SNARKs, STARKs, IOPs, PCPs, etc. etc.
  - Key property is that if a proof exists, then the program is (almost certainly) correctly-executed

ZK Rollup

- Basic idea:
  - Aggregator (may be malicious) collects transactions from participants, writes a "receipt" for each TX it receives
  - Aggregator verifies each (sequential) TX using EVM, updates state
  - Aggregator submits a Merkle root over all the transactions, plus a **short verifiable proof** of the following:
    1. Each TX verifies w.r.t. input state
    2. Merkle root is computed over all TXes and state
  - Blockchain (L1) simply verifies this proof

ETH 2, where are we?

ETH 2, where are we?



Ethereum 2.0 overall architecture. Original diagram by [Hsiao-Wei Wang](#).
From *https://media.consensys.net/state-of-ethereum-protocol-2-the-beacon-chain-c6b6a9a69129*

Sharding

- Right now a major property of Nakamoto-style blockchains is that all nodes see <u>all</u> transactions
  - This means we have no horizontal scaling
  - Adding nodes to the network increases decentralization (and possibly security) but does not increase throughput
  - How do we improve this?

Sharding (idea)

- Let's have multiple separate portions of the blockchain that interoperate
  - What are the challenges here?
  - What are the security consequences?
    - Especially consider Proof of Work
    - (Think about sidechains and merge mining)

Sharding (ETH2)

- Feature not yet launched, but here's the goal
  - There will be 64 shard chains
  - Designed to support weaker "validators", e.g., lightweight computers
    - Early phases will just add some extra data storage, won't support transactions
    - Ethereum Project doesn't really have this worked out!
    - Beacon chain will <u>somehow</u> assign stakers to validate shards

Proof of Stake

- We've discussed this before
  - Overall goal is to use coin stake, rather than hashpower, to determine who makes blocks
  - Second benefit is that we can obtain (non-probabilistic) finality, unlike Nakamoto PoW
  - What are the downsides?

How the Beacon PoS works

- This is basically the only part of Ethereum 2 that exists
  - So one expects it to be amazing!
  - Let’s dig in a little…

How the Beacon PoS works

- This is basically the only part of Ethereum 2 that exists
  - So one expects it to be amazing!
  - Let’s dig in a little…

How the Beacon PoS works

- Basic idea is to provide a "randomness beacon" to the rest of the system
  - Goal is to ensure that nobody can predict the decisions made by the blockchain before they are made
  - (This includes stuff like sharding committee assignments)

How the Beacon PoS works

- Step 1: "stakers" send 32 ETH to a special contract on the existing Ethereum1 main net
- Step 2: Staking contractor records this participant as a validator, so the beacon chain can see this
- Step 3: Active validators are selected (somehow, more soon) to propose new blocks on the Beacon chain — and later, on the shard chains
- Step 4: Validators can get their stake back but (and this is bananas) **only on one of the shard chains**

  did I mention that the shard chains don't exist?

Selecting randomness

- Holy cow this is nuts
- Main goal of the Beacon chain is for validators to pick random numbers
  - Why?

Selecting randomness

- Holy cow this is nuts
- Main goal of the Beacon chain is for validators to pick random numbers
  - For selecting validator nodes to propose blocks
  - For selecting "committees" of validators to verify shard chains
  - For things the Ethereum people haven't yet considered!

Selecting randomness II

- How do I make a group of people pick a random number?
  - (Board)

Proposing blocks

- Idea (cont'd)
  - Use this randomness to select a subset of the validators to form a committee (for the beacon chain) as well as a proposer that formulates each block
  - The validators publish signed attestations (votes) confirming previous blocks
  - The proposer collects these into a new block and sends them to the network

Discussion