Algorand Community Study Group

2: L1s, L2s, and alternative blockchains

HashMapsData2Value

Digital Community Champion
Algorand Foundation

June 26, 2023

Welcome!

- Welcome!
- Study group? Discord: #cryptography-study-group
- GitHub Repo:
  https://github.com/HashMapsData2Value/Algorand-Community-Study-Group
- Please introduce yourselves
- Topic - L1s, L2s, and alternative blockchains

Agenda

1. Layer 1
2. "Alternative" Blockchains
3. Layer 2 solutions
4. Algorand

Anatomy of an L1 - A network of nodes

- Nodes
  - Block assembly and validation software
  - Networking layer, block propagation
  - Blockchain history
- Reach Consensus
  - Who? Whose block...
  - How? PoS, PoW, etc...
  - When? Block speed, finality...
- Security Model
  - Collusion and Nakamoto coefficient
  - Cost of corruption
- Abstraction: The World Computer
- Why need more than L1 in the first place?

"Alternative" Blockchains

Warning

These definitions are not 100% set in stone.

Co-Chains

- Separate, basically copy+paste blockchain of an established L1, e.g. Algorand
- I.e. own and independent history, node network, security consideration
- Interoperability: Two-way bridge
- Permissioned vs Non-Permissioned, licensing

Side-Chain

- Separate and independent network of nodes, own security considerations
- BUT! Goal is to assist another L1 from the side
- Validators on side-chain may pay attention to "parent chain" (or vice-versa)
- E.g. Milkomeda C1 (Cardano EVM Sidechain):
  - Token: mADA, wrap with Milkomeda Bridge
  - Has a built-in EVM $\rightarrow$ Solidity contracts
  - Tx on Cardano calls contract on Milkomeda C1

Oracles

- Goal: plug off-chain "truth" into on-chain
- Problem: Can't just trust one person's opinion/report
- Solution: "Ask the audience" - decentralized network of nodes, incentivized to reach consensus
- E.g. Goracle servicing Algorand:
  - Token: GORA
  - Main (on-chain) contract:
    1. Nodes stake their tokens, register public key
    2. Consumers request data, subscribe
  - Off chain: Nodes poll main contract for requests, prepare data in block, PPoS sortition
  - Votes for block tallied, data aggregated, presented to consumer
  - Summary: Nodes exist, but token and state is kept on-chain

Bridge Networks

- Similar to Oracles - report activity on chain A to chain B
- Bridge components:
  - "Deposit Box" contract on chain A, takes token X
  - "Withdrawal Box" contract on chain B, gives wrapped token X
  - Messaging service between the contract
- How to prove to WB that you deposited X into DB so you can extract the wrapped tokens?
- $\rightarrow$ Rely on *decentralized* network of nodes (with incentives)
- E.g., Portal/Wormhole (Messina.one), Flare
- (Better than "multi-sig"... Is there a better way?)

Layer 0s

- "Chain of Chains" approach
- Polkadot: Relay Chain + Parachains, bidding for slots
- Cosmos: Cosmos Hub + Zones, customized
- Avalanche: Primary Network Subnet: **C**ontract Chain + E**x**change Chain + **P**latform Chain, + custom subnets

Layer 2

Channels

- Open: Lock up funds in multi-sig contract/account/tx, then construct and sign txs off-chain. Many txs can be passed back-and-forth, like ledger of IOUs.
- Close: Final up-to-date settlement tx is broadcast, distributing funds from contract on-chain.
- Multi-sig + timelock
- Simple (Payment), Advanced (State)
- E.g.: Bitcoin's Lightning Network
  - Network of bidirectional payment channels
  - Onion routing
  - Cooperative vs Uncooperative Closing
  - Dispute period $\rightarrow$ Publish most up-to-date tx

Rollups

- Rather than 100 people signing 100 tx requiring 100 signature checks on L1, what if they were all bundled up?
  1. Send tx to rollup "sequencer" or group of "operators"
  2. Sequencer pools tx batch and posts to L1
  3. Rollup "validators" validate batch off-chain
  4. IF valid, validators post rollup block to L1 with proof
- *Crucially*, that proof can be verified *within* the L1! And, transaction data got posted.
- After requesting withdrawal, user posts merkle proof validating inclusion of user's withdrawal tx in verified tx batch.
- Types of proof?
  - Optimistic Rollups: Fraud Proofs
  - ZK Rollups: Validity Proofs

Optimistic Rollups

- We assume *optimistically* that validators are acting honestly/non-maliciously when posting rollup block. They don't need to *prove* anything, just provide stake.
- If malicious rollup block is posted, as long as there's at least 1 honest node auditing, a fraud proof will be provided as a challenge, the rollup block rejected and the stake lost to that challenger.
- "Challenge time" means long settlement times! To give adequate time for challengers even with some malicious nodes. One week wait to withdraw funds from L2 contract.
- Optimistic Rollups can incorporate many EVM specs, allowing for sophisticated EVM contracts to be deployed.

ZK Rollups

- Rather than optimistic, pessimistic and suspicious. Requires Zero-Knowledge Proofs of validity, which have their own considerations.
- Allows for very fast finality.
- ZK Rollups good for rolling up simple txs, research topic how to translate EVM opcodes over.
- Considered the "future" of rollups and scaling on e.g. Ethereum.

Plasma Chain

- Separate blockchain "anchored" to a "root" chain, but executing tx off-chain with own mechanism for block validation. Similar to side-chains but benefit from main chain security.
- Most often just one validation node, settling off-chain then posting merkle root "state commitments" to contract on Ethereum. Contract processes user fund entry/exit.
- Plasma contract accepts **fraud proofs**, like optimistic rollups, but stores the transaction data off-chain.

Validium

- Validiums, like ZK Rollups, post ZKPs (ZK-SNARKs or ZK-STARKs) to a contract on Ethereum. This allows near-instant withdrawals (once ZKP is generated!).
- Unlike ZK Rollups HOWEVER validium contracts do NOT keep tx data on-chain, instead off-chain, saving on cost. If data availability managers withhold off-chain state data from users, users cannot withdraw.
- E.g. StarkWare StarkEx runs in both ZK Rollup or Validium data-availability modes.

Future of Ethereum

- *A Rollup-centric Ethereum Roadmap* - Vitalik Buterin's forum post
- "[...] the Ethereum ecosystem is likely to be all-in on rollups (plus some plasma and channels) as a scaling strategy for the near and mid-term future."
- Dank-sharding: "instead of providing more space for transactions, Ethereum sharding provides more space for blobs of data [Ethereum] itself does not attempt to interpret. Verifying a blob simply requires checking that the blob is available - that it can be downloaded from the network. The data space in these blobs"

Comparison Table

L2 Name	On-Chain Data?	Fraud Proof?	Validity Proof?
Channel	No	No	No
Optimistic Rollup	Yes	Yes	No
ZK Rollup	Yes	No	Yes
Plasma	No	Yes	No
Validium	No	No	Yes

Algorand

Algorand - pros cons

- Algorand has **instant settlement finality**, soon 3.3s blocktime and 10k+ TPS
- PPoS leader selection $\rightarrow$ bottle-neck lies in networking
- Where can L2 come in?
  - Smart contracts limited by opcode budget
  - Application specific L2? (If co-chain "too much".)
- IMO the real improvements relate to relay nodes and archiving nodes...

Algorand Inc's L2 Research

- Algorand's Smart Contract Architecture - "Our Two Tier Architecture", Blog post by Silvio (May 2020)
- Speculative Smart Contracts in the Algorand Blockchain - Victor Luchangco (June 2022)
- Problem: Accomodating data-intensive and computation-intensive esoteric contracts
  - Clarity lang, a "decidable, predictable" language. Smart contracts kept in L2 off-chain but called from L1.
  - L2 nodes running PPoS run batch result of contract calls and collect signatures in compact certificates (state proofs) which can verified in AVM on L1.
- For now L1's AVM been able to grow very well. No recent update (AFAIK).

Milkomeda AVM Rollup

- Brings EVM to Algorand. Wrap Algo $\rightarrow$ mAlgo on Milkomeda A1, with bridge
- Still in development, not yet using BoxStorage but combined tx notes of 16 group tx (16 kb).

1. Batch Proposal Phase: Sequencer divides batch of txs into groups, create Merkle root of batch. Post Merkle root on-chain to declare "Block Proposal" and initiate next phase.
2. Batch Availability Phase: With Merkle root on-chain (in "block 0"), all batch data must be posted within 6 blocks. Tx groups + Merkle proof. If a new proposal is posted it'll be ignored, or if not all the data is posted within 6 blocks it'll be considered invalid. Rollup nodes observe everything, verify the Merkle tree and that the batch is "available".
3. Batch Validation Phase: Locally, rollup nodes verify that the transactions (which could be EVM calls) are valid. If they are, the state of the rollup advances a step. Otherwise they do not and state moves back into Batch Proposal.

Milkomeda AVM Rollup

- Who gets to be sequencer? Who gets to be validator?
  - Sequencer and Validator elections using auction to allocate "slots" of blocks on L1 to be sequencer
  - Proof of stake with stake locked up in rollup contract, VRF using L1 for randomness seed
  - Reward of mAlgo if honest, slashing if dishonest (can tolerate 1/3 malicious)
  - Currently in "basic" mode, relying on single centralized sequencer/batcher
- How to exit? Provide an exit tx signed by 2/3 of the node validators to contract.
- Problem: Algorand cannot verify EVM operations. Thus the L1 cannot be relied on the validate state of rollup.
  - On-going problem: "The current goal to be implement for the Algorand EVM Rollup is a fraud proof system." Optimistic rollup, page still under construction.
- On-going research into BoxStorage

Algorand as Data Availability Layer

- Tweet by Sebastien Guillemot (DcSpark, Milkomeda, PaimaStudios)
- AVM EC Math upgrade - coming soon
- Problem: Storing data off-chain AND verifying it is "still there".

Sebastien suggest the following:

1. Store data in BoxStorage from other chain
2. Create (off-chain) KZG commitments of data in BoxStorage
3. Wait for state proof, it will verify the successful inclusion of BoxStorage tx
4. Take state proof and into other chain
5. Some time later...
6. Request comes to validate data is this available
7. Trigger data availability sampling (randomly sample and check it exists in various places), leveraging KZG commitment scheme
8. Validate KZG commitment (pairing check) inside AVM
9. Take state proof into other chain

Mixers as L2s?

- Are mixers L2s? Mixers are a type of "one-shot" L2s.
- Mixer: smart contract that you (along with others) deposit a fixed chunk X of money into from account A. Later, you withdraw that money from account B, using a proof that you are *one of* the depositors, without revealing exactly *who*. Your money is mixed up and obfuscated, granting privacy through plausible deniability.
- When depositing you provide a "commitment" or public key; to withdraw you provide a "nullifier" or "key image". (Merkle Proof ZKP vs Ring Signatures.)
- Wallet Layers:
  1. First layer is account-based Algorand. You maintain your keys for accounts A and B.
  2. Second layer is UTXO-based mixer contract. E.g. 20 000 Algo in 100 Algo increments requires storing 200 withdrawal proofs, allowing 200 transactions.

Open discussion

Any other thoughts?

The End (for now!)