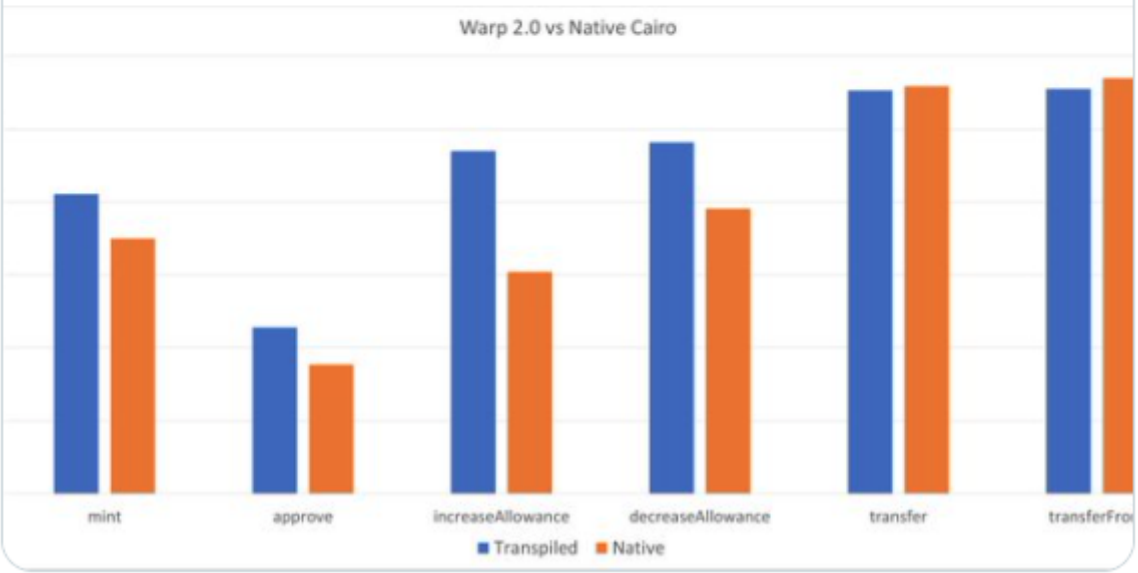
Warp update today

We've just made Warp 2.0 public:
github.com/NethermindEth/...

It's still absolutely not production-ready, there's still stuff we need to implement. PRs are welcome! You can see a link to our notion at the bottom of the README for tasks that are being worked on & need to be worked on

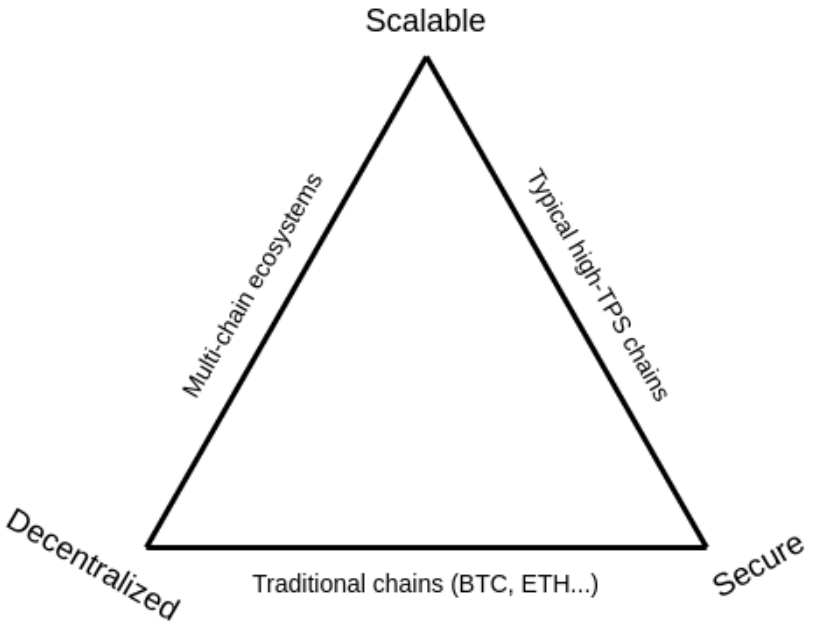
> **Greg Vardy** @0xGreg_ · Feb 17
> Dai transpiled from Solidity to Cairo with Warp 2.0 vs its hand-written Cairo counterpart. Efficiency is almost exactly the same, and the transpiled contract is 10KB smaller than the native one



Lesson 9 - ZK Rollups

ZK Rollups are an application of ZK technology focussing on the proof of computation aspect, rather than the zero knowledge aspect.
They rely on proofs being succint and non interactive.

Why do we need rollups ?

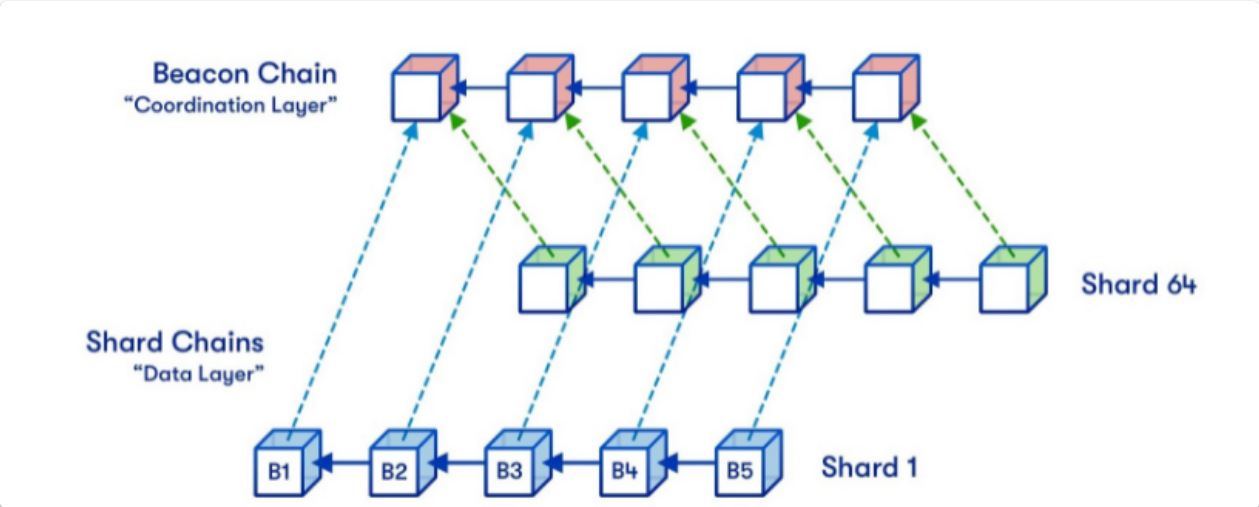


Approaches to increase scalability

ON CHAIN (L1) SOLUTIONS

- Consensus mechanism
  Using DPoS - EOS
  PoH - Solana
  Snow etc. - Avalanche
  For example moving from Proof of Work to Proof of Stake - Ethereum
  The Beacon chain is already live, in 2022 there will be the merge of main net and the beacon chain.

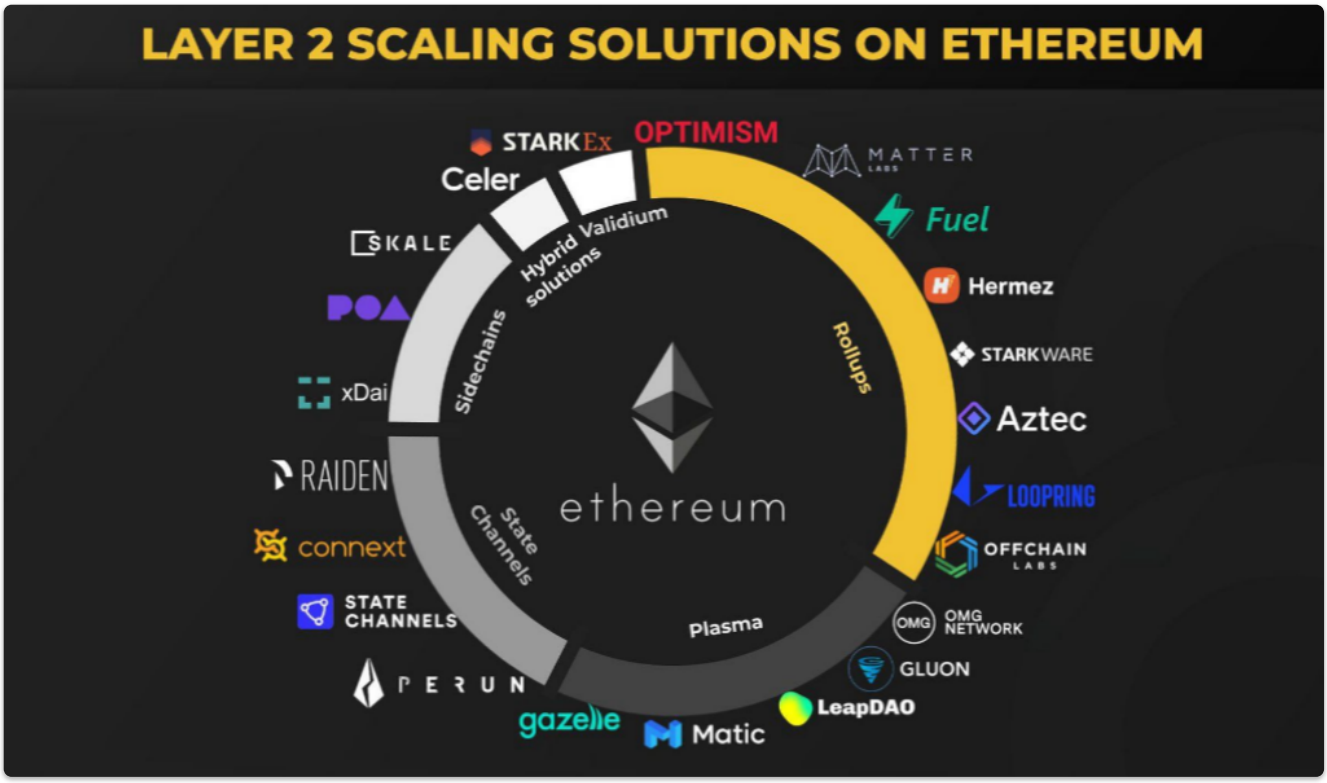
- Sharding



OFF CHAIN SCALING (LAYER 2)

In essence transactions are submitted to these layer 2 nodes instead of being submitted directly to layer 1 (Mainnet).
For some solutions the layer 2 instance then batches them into groups before anchoring them to

layer 1, after which they are secured by layer 1 and cannot be altered.
A specific layer 2 instance may be open and shared by many applications, or may be deployed by one project and dedicated to supporting only their application.



	State channels	Sidechains[0]	Plasma	Optimistic rollups	Validium	zkRollup
Security						
Liveness assumption (e.g. watch-towers)	Yes	Bonded	Yes	Bonded	No	No
The mass exit assumption	No	No	Yes	No	No	No
Quorum of validators can freeze funds	No	Yes	No	No	Yes	No
Quorum of validators can confiscate funds	No	Yes	No	No	Yes [1]	No
Vulnerability to hot-wallet key exploits	High	High	Moderate	Moderate	High	Immune
Vulnerability to crypto-economic attacks	Moderate	High	Moderate	Moderate	Moderate	Immune
Cryptographic primitives	Standard	Standard	Standard	Standard	New	New
Performance / economics						
Max throughput on ETH 1.0	1..∞ TPS [2]	10k+ TPS	1k..9k TPS [2]	2k TPS [3]	20k+ TPS	2k TPS
Max throughput on ETH 2.0	1..∞ TPS [2]	10k+ TPS	1k..9k TPS [2]	20k+ TPS	20k+ TPS	20k+ TPS
Capital-efficient	No	Yes	Yes	Yes	Yes	Yes
Separate onchain tx to open new account	Yes	No	No	No	No	No [5]
Cost of tx	Very low	Low	Very low	Low	Low	Low
Usability						
Withdrawal time	1 confirm.	1 confirm.	1 week [4] ([7])	1 week [4] ([7])	1..10 min [7]	1..10 min [7]
Time to subjective finality	Instant	N/A (trusted)	1 confirm.	1 confirm.	1..10 min	1..10 min
Client-side verification of subjective finality	Yes	N/A (trusted)	No	No	Yes	Yes
Instant tx confirmations	Full	Bonded	Bonded	Bonded	Bonded	Bonded
Other aspects						
Smart contracts	Limited	Flexible	Limited	Flexible	Flexible	Flexible
EVM-bytecode portable	No	Yes	No	Yes	Yes	Yes
Native privacy options	Limited	No	No	No	Full	Full

[0] Some researchers do not consider them to be part of L2 space at all, see https://twitter.com/gakonst/status/1146793685545304064

[1] Depends on the implementation of the upgrade mechanism, but usually applies.

[2] Complex limitations apply.

[3] To keep compatibility with EVM throughput must be capped at 300 TPS

[4] This parameter is configurable, but most researchers consider 1 or 2 weeks to be secure.

[5] Depends on the implementation. Not needed in zkSync but required in Loopring.

[7] Can be accelerated with liquidity providers but will make the solution capital-inefficient.

Matter Labs

Updated 2021–02–18

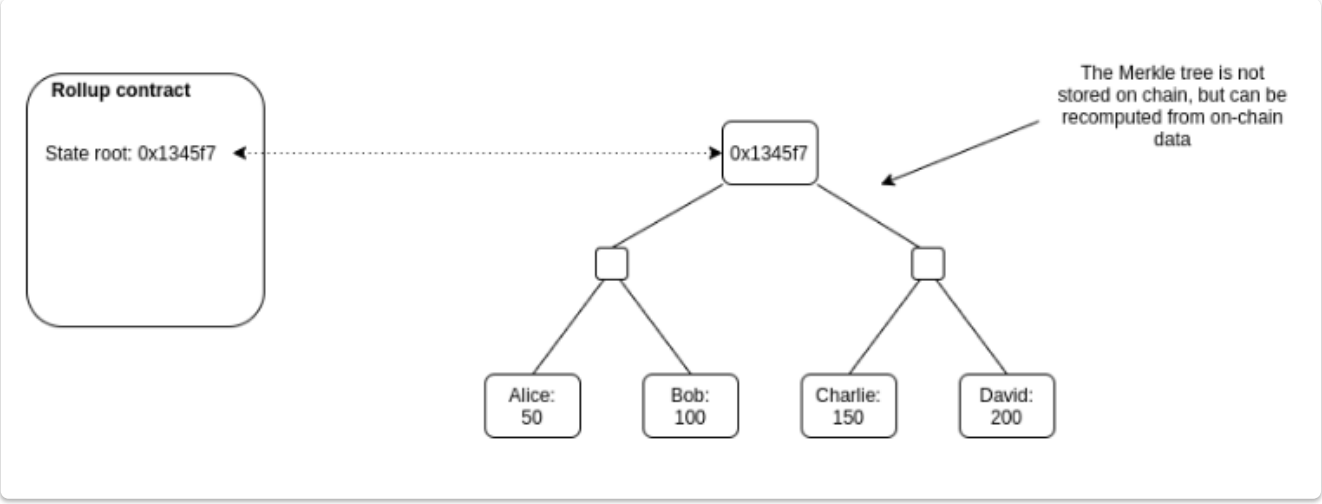
Rollups

Rollups are solutions that have

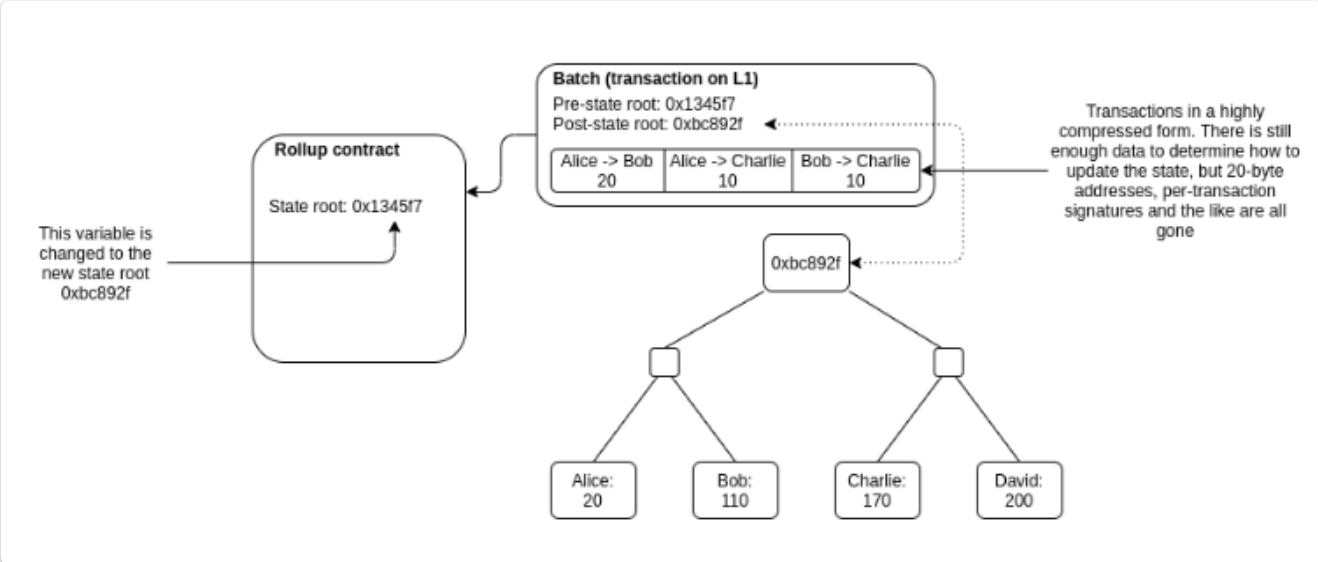
- transaction execution outside layer 1
- transaction data and proof of transactions is on layer 1
- a rollup smart contract in layer 1 that can enforce correct transaction execution on layer 2 by using the transaction data on layer 1

The main chain holds funds and commitments to the side chains
The side chain holds additional state and performs execution
There needs to be some proof, either a fraud proof (Optimistic) or a validity proof (zk)

Rollups require “operators” to stake a bond in the rollup contract. This incentivises operators to verify and execute transactions correctly.



Anyone can publish a batch, a collection of transactions in a highly compressed form together with the previous state root and the new state root (the Merkle root after processing the transactions). The contract checks that the previous state root in the batch matches its current state root; if it does, it switches the state root to the new state root.



There are currently 2 types of rollups

- Zero Knowledge Proof rollups
- Optimistic rollups

OPTIMISTIC ROLLUPS

The name Optimistic Rollups originates from how the solution works. 'Optimistic' is used because aggregators publish only the bare minimum information needed with no proofs, assuming the aggregators run without commiting frauds, and only providing proofs in case of fraud.
For more details see Arbitrum research

ZKP Rollups

An excellent report

An overview from Ethereum
The ZK-Rollup scheme consists of two types of users: transactors and relayers.

- Transactors create their transfer and broadcast the transfer to the network. The transfer data consists of an indexed “to” and “from” address, a value to transact, the network fee, and nonce. A shortened 3 byte indexed version of the addresses reduces processing resource needs. The value of the transaction being greater than or less than zero creates a deposit or withdrawal respectively. The smart contract records the data in two Merkle Trees; addresses in one Merkle Tree and transfer amounts in another.
- Relayers collect a large amount of transfers to create a rollup. It is the relayers job to generate the SNARK proof. The SNARK proof is a hash that represents the delta of the blockchain state. State refers to “state of being.” SNARK proof compares a snapshot of the blockchain before the transfers to a snapshot of the blockchain after the transfers (i.e. wallet values) and reports only the changes in a verifiable hash to the mainnet.

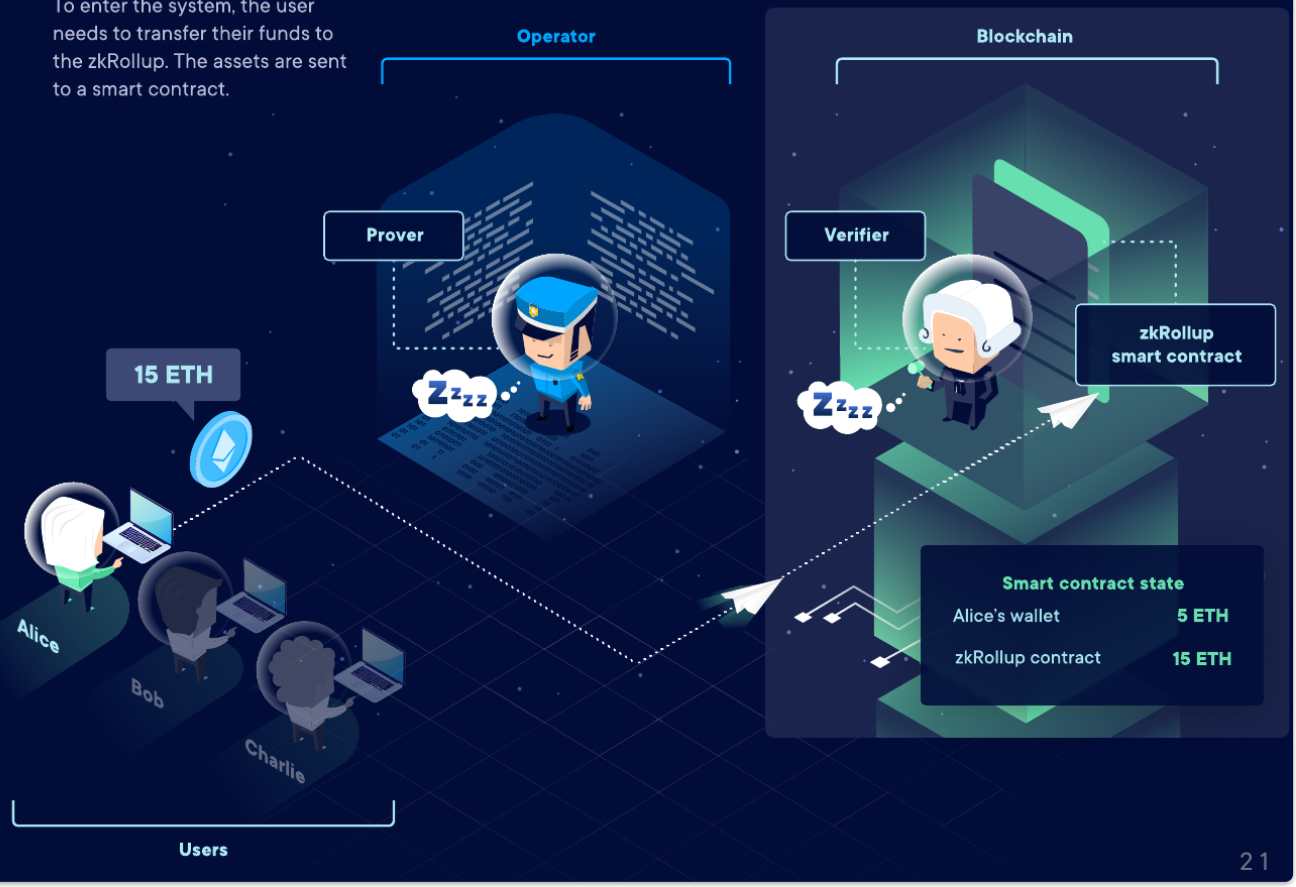
It is worth noting that anyone can become a relayer so long as they have staked the required bond in the smart contract. This incentivises the relayer not to tamper with or withhold a rollup.

ZK Rollup Process

From Ethworks

02. Alice's Enter

To enter the system, the user needs to transfer their funds to the zkRollup. The assets are sent to a smart contract.



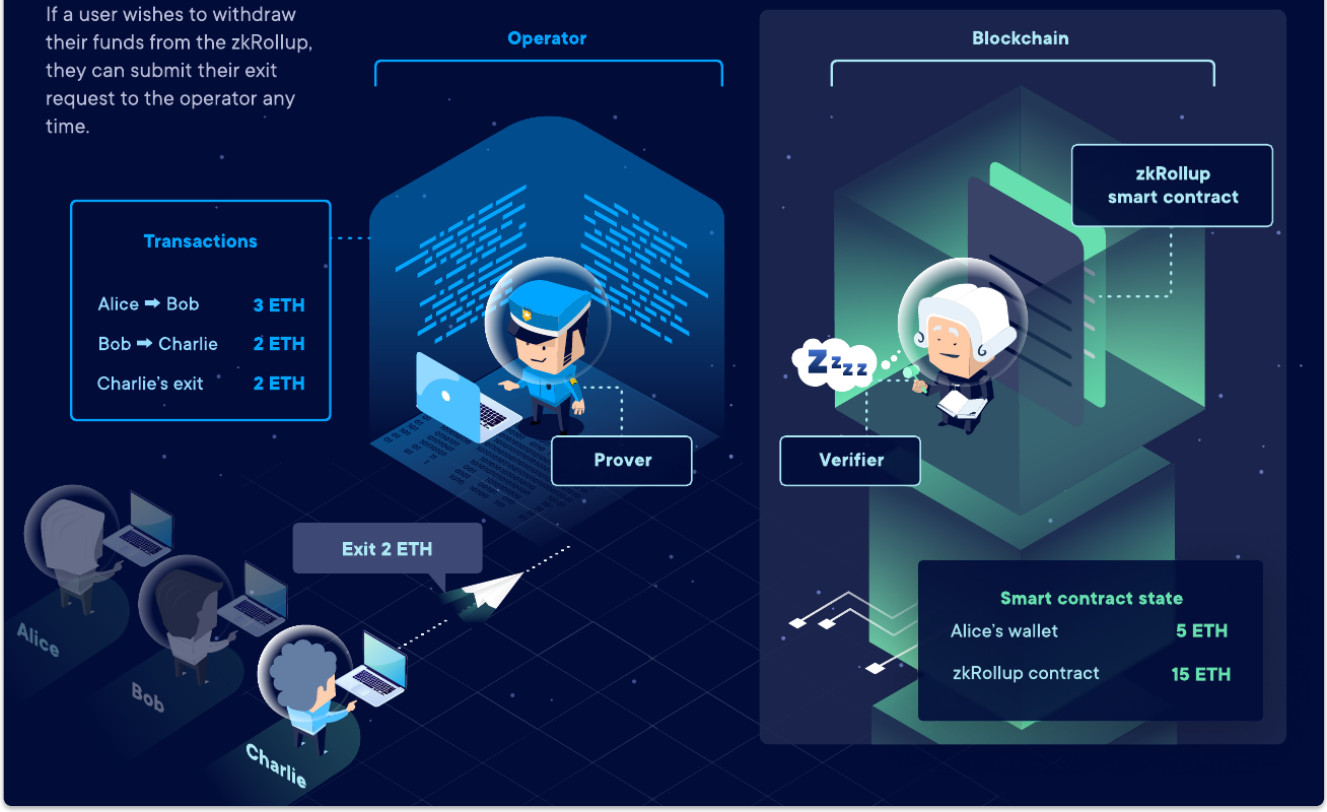
03. Alice's Transfer

The user can now transfer their funds to another person. They sign the transaction and submit it to the zkRollup operator.

04. Bob's Transfer

05. Charlie's Exit

If a user wishes to withdraw their funds from the zkRollup, they can submit their exit request to the operator any time.



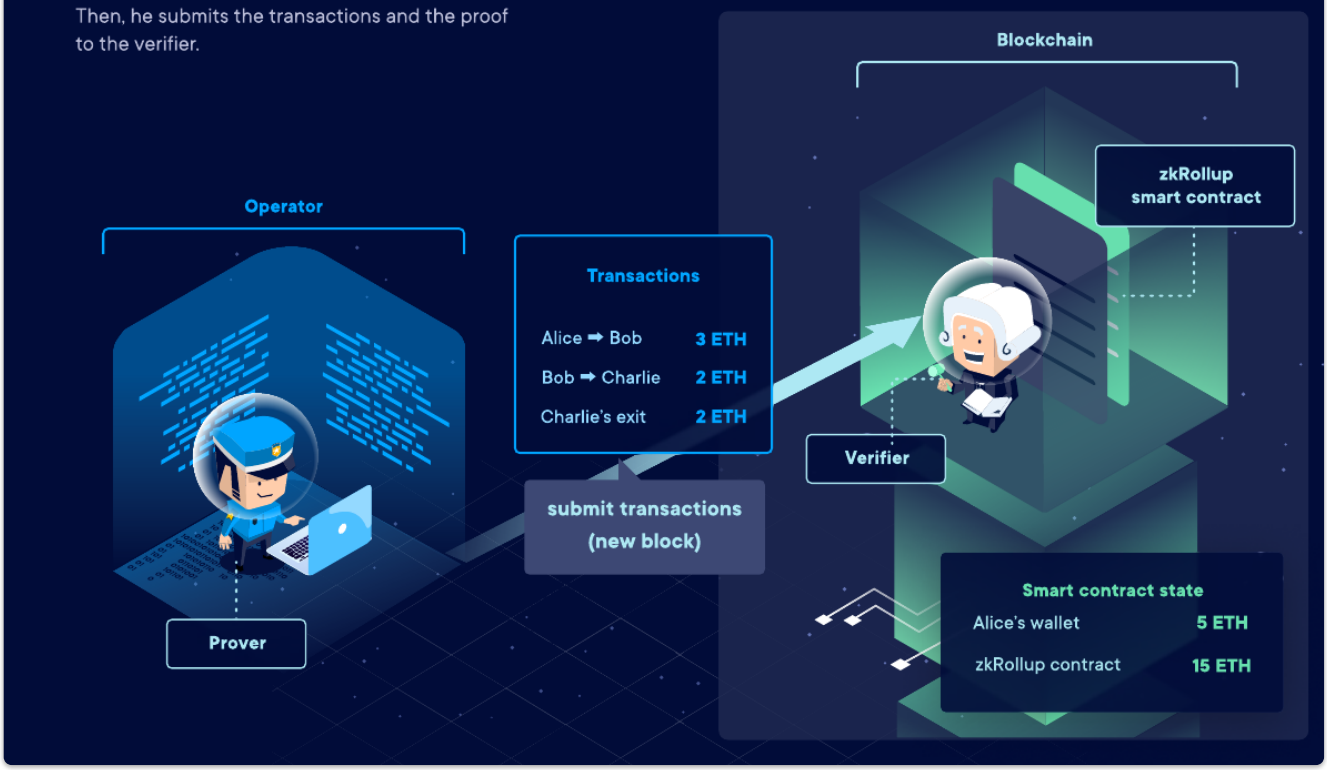
06. Collecting Transactions

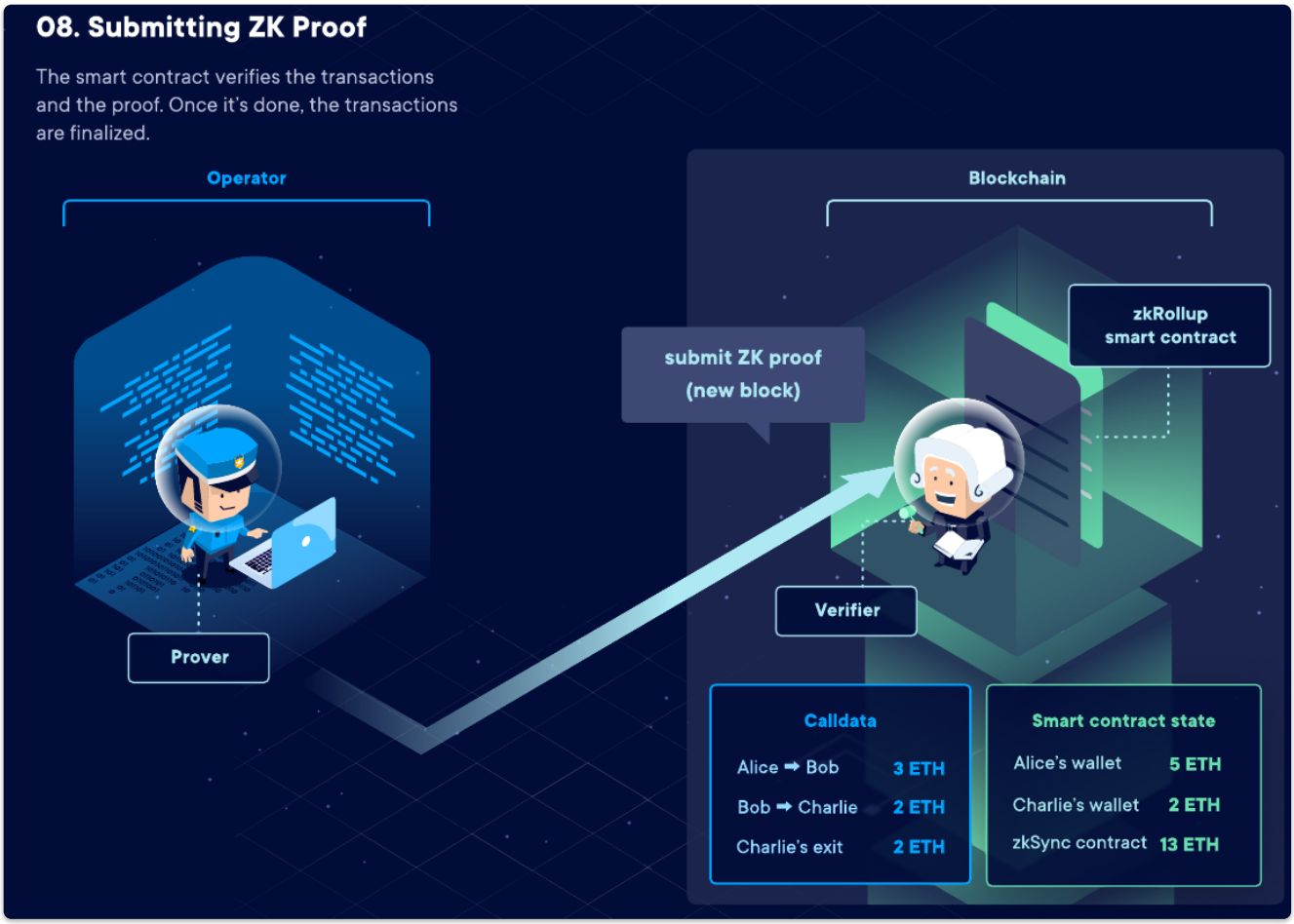
In the meantime, the operator collects transactions and exit requests from many users.

* Note that even if Bob and Charlie didn't have any funds on the zkRollup, they could still receive transfers from other users.

07. Submitting Transactions

Once in a while, the operator bundles the collected transactions together and generates a ZK proof. Then, he submits the transactions and the proof to the verifier.





Transaction Compression

How does compression work?

A simple Ethereum transaction (to send ETH) takes ~110 bytes. An ETH transfer on a rollup, however, takes only ~12 bytes:

Parameter	Ethereum	Rollup
Nonce	~3	0
Gasprice	~8	0-0.5
Gas	3	0-0.5
To	21	4
Value	~9	~3
Signature	~68 (2 + 33 + 33)	~0.5
From	0 (recovered from sig)	4
Total	~112	~12

Part of this is simply superior encoding: Ethereum's RLP wastes 1 byte per value on the length of each value. But there are also some very clever compression tricks that are going on:

Comparison of the types

Property	Optimistic rollups	ZK rollups
Fixed gas cost per batch	~40,000 (a lightweight transaction that mainly just changes the value of the state root)	~500,000 (verification of a ZK-SNARK is quite computationally intensive)
Withdrawal period	~1 week (withdrawals need to be delayed to give time for someone to publish a fraud proof and cancel the withdrawal if it is fraudulent)	Very fast (just wait for the next batch)
Complexity of technology	Low	High (ZK-SNARKs are very new and mathematically complex technology)
Generalizability	Easier (general-purpose EVM rollups are already close to mainnet)	Harder (ZK-SNARK proving general-purpose EVM execution is much harder than proving simple computations, though there are efforts (eg. Cairo) working to improve on this)
Per-transaction on-chain gas costs	Higher	Lower (if data in a transaction is only used to verify, and not to cause state changes, then this data can be left out, whereas in an optimistic rollup it would need to be published in case it needs to be checked in a fraud proof)
Off-chain computation costs	Lower (though there is more need for many full nodes to redo the computation)	Higher (ZK-SNARK proving especially for general-purpose computation can be expensive, potentially many thousands of times more expensive than running the computation directly)

StarkEx

How does this relate to StarkNet ?

StarkNet is a permissionless decentralized ZK-Rollup that supports independent deployment of smart contracts. Any developer can write and deploy their smart contract permissionlessly. StarkNet also supports composability.

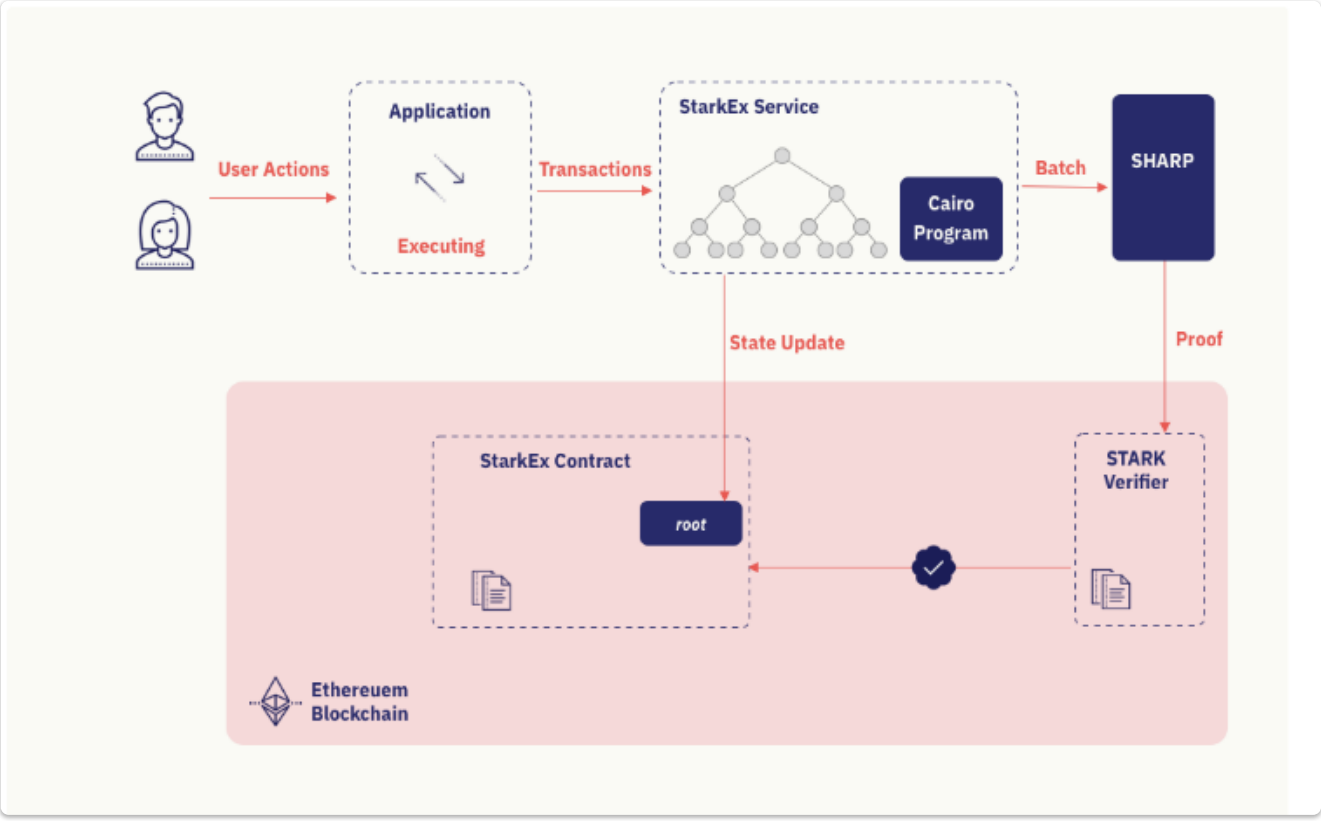
StarkEx is a permissioned tailor-made scaling engine, designed by StarkWare to fit the specific needs of apps.

Both StarkNet and StarkEx provide scalability and L1 security by using STARK-based validity proofs, and both are designed to support general computation, allowing any use case to be scaled

Data location and privacy

StarkEx can store data according to a range of user and application needs.
In ZK-Rollup mode data is published on-chain. In Validium mode data is stored off-chain.
Volition is a hybrid data availability mode, where the user can choose whether to place data on-chain or off-chain.

There are on chain and off chain components



OFF CHAIN

The off-chain component holds the state of orders, performs transaction executions in the system, and sends state updates to the on-chain component.

ON CHAIN

The on-chain component holds the state commitments and the system assets; and is responsible for enforcing the validity of state transition. In the case of StarkEx for spot trading, it also manages the on-chain accounts, which are useful in the context of Layer 1 (L1) dApp interoperability and DeFi pooling.

Process

All the transactions in the system are executed by the Application and sent to the StarkEx Service.

- The StarkEx Service batches transactions and sends the batch to SHARP, a shared proving service, to generate a proof attesting to the validity of the batch.
- SHARP sends the STARK proof to the STARK Verifier for verification.
- The service then sends an on-chain state update transaction to the StarkEx Contract, which will be accepted only if the verifier finds the proof valid.

Users interact with the system in two ways:

- by sending on-chain transactions to the StarkEx Contract
- off-chain transactions to the Application.

For an off-chain account, users

- Register their starkKey to their Ethereum Address in the StarkEx Contract.
- The starkKey is used to authenticate the user's off-chain transactions.
- The user then deposits their funds to the StarkEx Contract.
- After the Application accepts the deposit, the user can use their funds off-chain.
- Users submit Transfers or Limit Orders, signed by their starkKey, directly to the Application.
- Limit orders are matched together and sent as a Settlement to the StarkEx Service.

To withdraw their funds,

- users submit an off-chain withdrawal request to the Application.
- This request is sent to the StarkEx Service, and the user can access their funds on-chain once the state update containing the withdrawal is accepted.

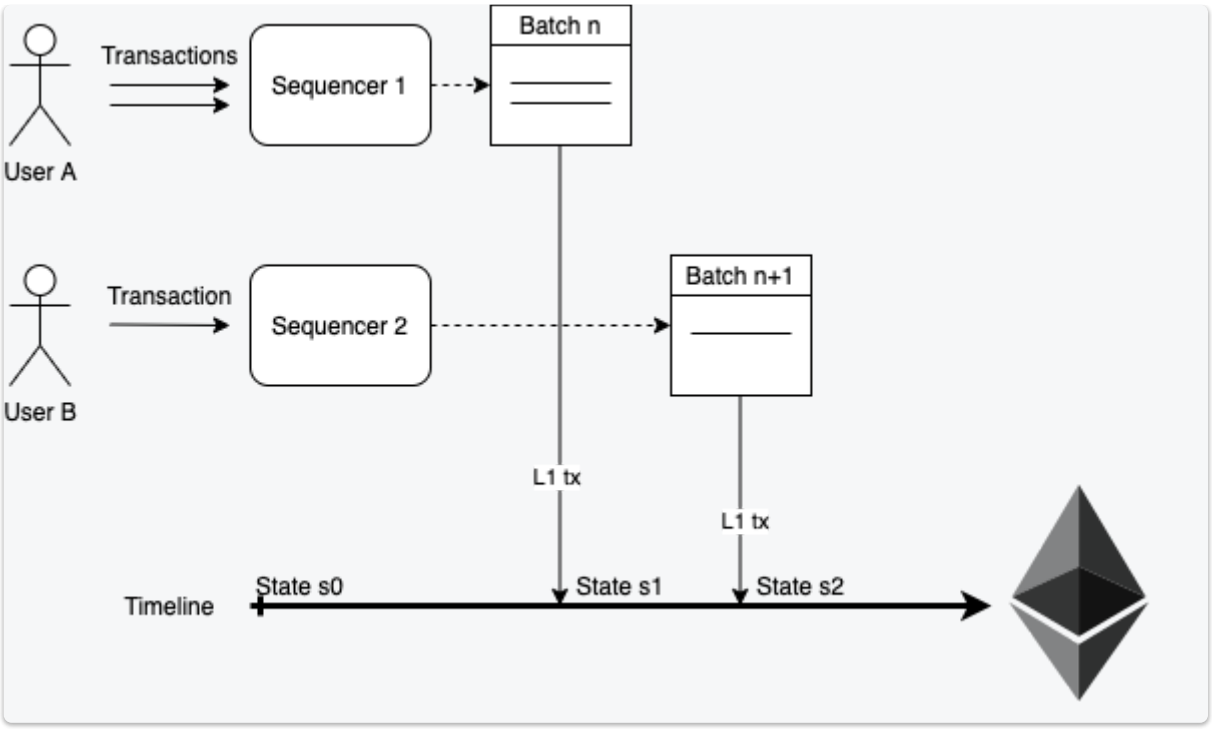
Do rollups increase centralisation ?

Polygon (Hermez) team are working on a protocol Proof of Efficiency
This involves 2 permissionless roles :

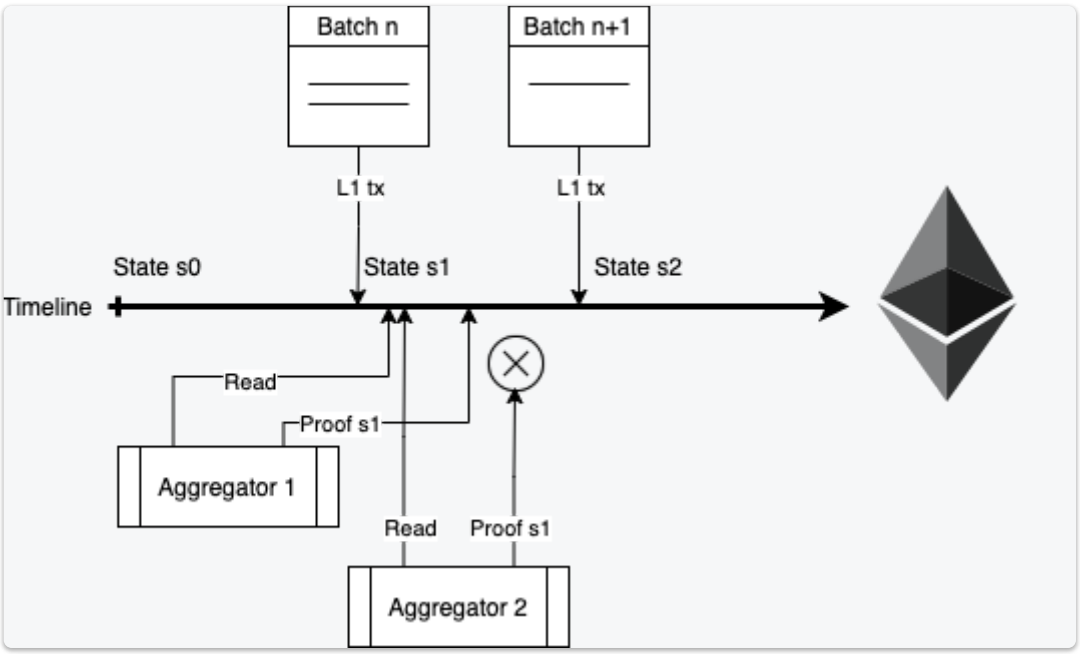
- Sequencer
- Aggregator

From article

Sequencers collect transactions from users on the rollup, then select and pre-process new batches of this Layer 2 data. Finally, they send transactions to Layer 1 to be recorded. Sequencers also deposit a fee in $MATIC token as an incentive for Aggregators to include the batch in a zero knowledge proof.



Aggregators are the parties that participate in the consensus protocol of PoE. The right to create the validity proof of a new state of the L2 is earned simply by being the first Aggregator to do it.



Data Availability problem

Data maybe kept on chain (L1), this gives the data the same security guarantees as native assets. Data availability depends on nodes holding the data and making it available if needed to other nodes.

If we store data off chain, we introduce a (further) data availability problem, users interacting with the L2 will need access to that data, so the holders of the data will need to be incentivised to provide it.

For StarkEx, 3 modes are available :

- In ZK-Rollup mode data is published on-chain.
- In Validium mode data is stored off-chain.
- Volition is a hybrid data availability mode, where the user can choose whether to place data on-chain or off-chain.

There are on chain and off chain components

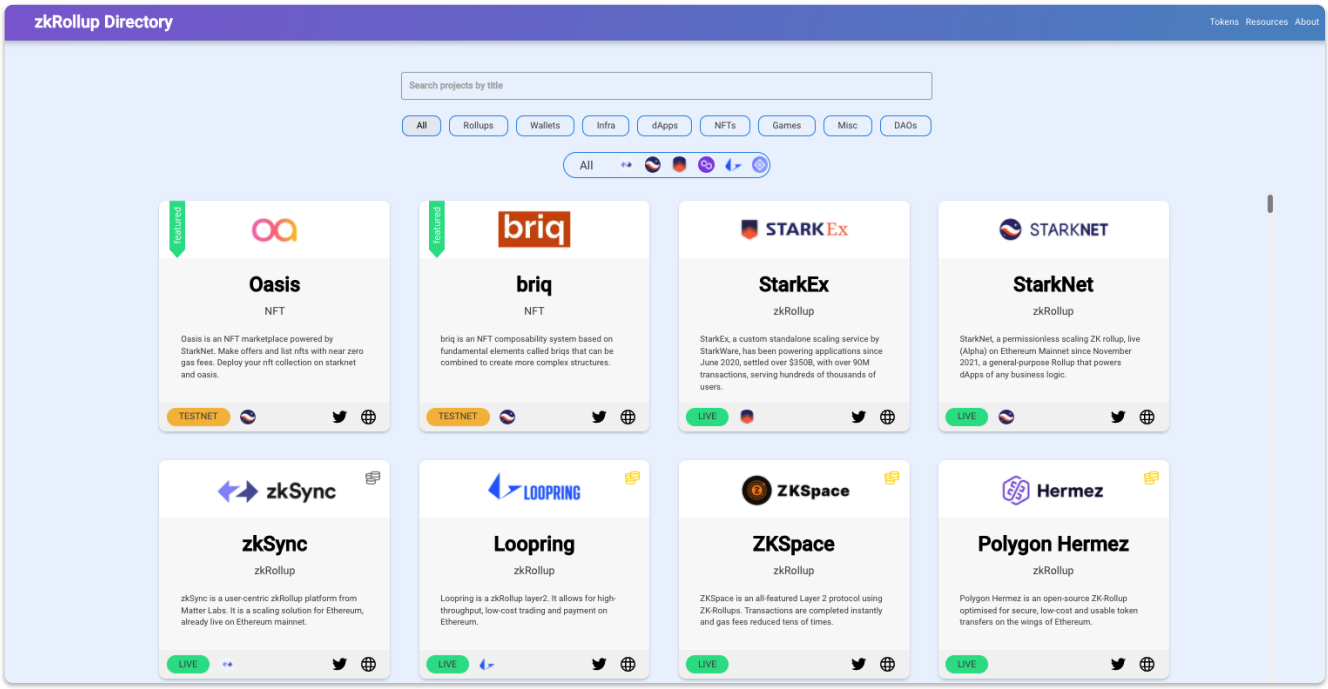
Other approaches using zero knowledge

ZKEVM

zkEVM is a virtual machine that executes smart contracts in a way that is compatible with zero-knowledge-proof computation.

Survey of zk Rollup projects

See zkRollups.xyz



Economics behind rollups

A breakdown of costs involved in rollups
See article and analytics

L2 Operator costs

Computational resources are spent by the operators on L2 in batching transactions, creating proofs etc.

L1 Data publication

Transaction data is compressed and submitted as calldata, which is cheaper than memory, but still a significant cost.

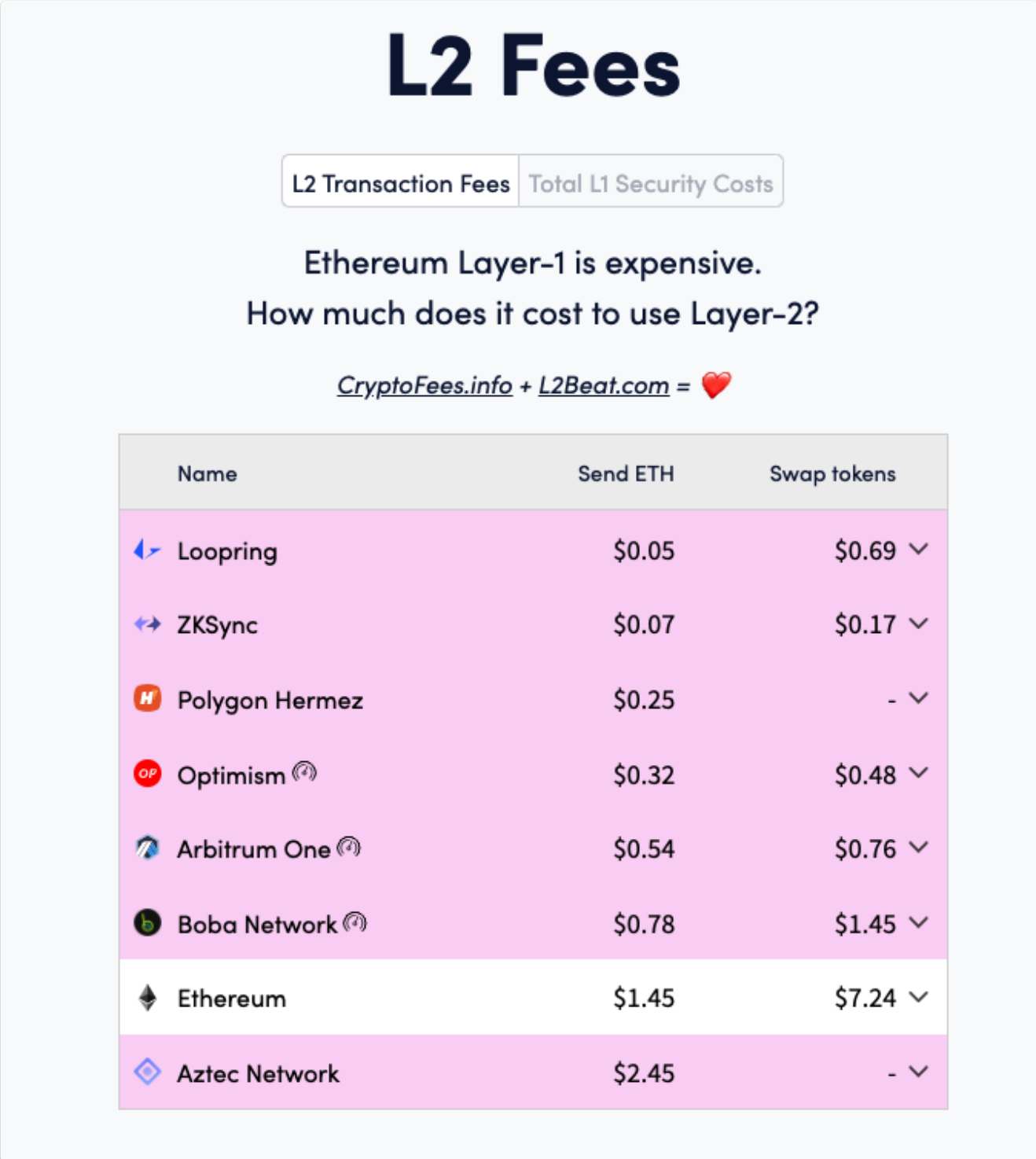
L1 Congestion costs

There is a cost in time, or lost opportunity caused by network congestion.

Revenues

The rollup project (or operators) will charge the user to cover the costs, one way that is used is to use the L1 base fee and add a hedge to this against the variable costs of data publication. Barnabé Monnot,see article suggests that derivatives could be used (such as future contracts) to ensure profit for the operator.

For an overview of fees see L2Fees



L2 Fees

L2 Transaction Fees | Total L1 Security Costs

How much are rollups paying for Ethereum's security?

CryptoFees.info + *L2Beat.com* = ❤️‍💔

Name	One day security costs
Arbitrum One	$42,124.69
Optimism	$25,131.93
dYdX	$20,035.07
Metis	$15,251.59
Starkware Shared Prover Applications	$5,383.94
Aztec Protocol	$1,443.44
Loopring	$1,244.63
Boba	$1,107.93
Polygon Hermez	$135.79
Total	$111,859.02