Blockchain Scalability Solutions

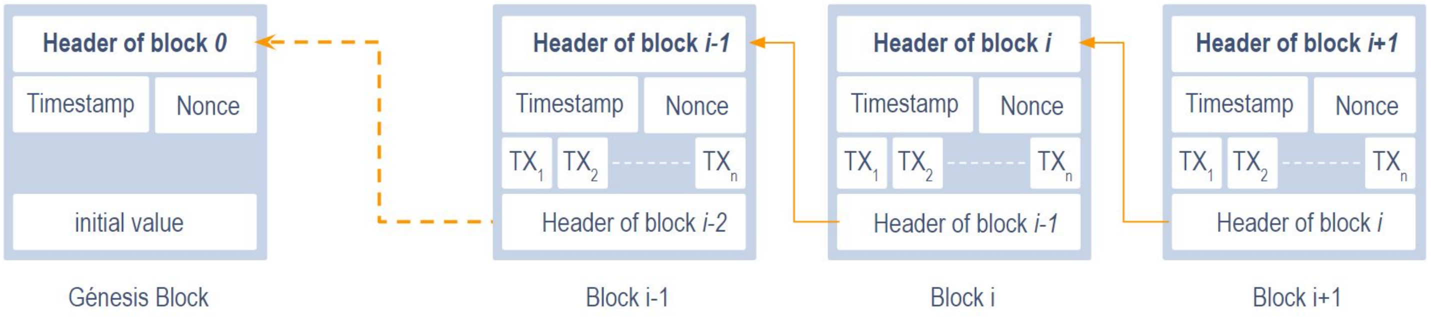
Gimer Cervera - Cartesi

September, 2021

Agenda

- Blockchain
- Ethereum
- Scalability Issues
- Rollups: zk-Rollups & Optimistic
- Arbitrum and Optimism
- Conclusions
- Q&A

Blockchain

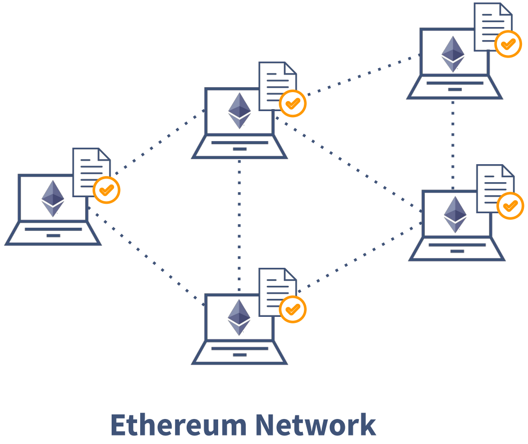


Header of block i = h(Timestamp, Signature, nonce, $M(TX_1, TX_2, \ldots, TX_n)$, header of block_{i-1})

- n = number of transactions.
- TX_i = i-Transaction (e.g., Alice sends $40 to Bob).
- Signature = Creator’s signature.
- Nonce = arbitrary number used only once.
- $h(x_1, x_{2,\ldots,} x_k)$ = hash function.
- $M(TX_1, TX_2, \ldots, TX_n)$ = Merkle tree root.

Ethereum Virtual Machine (EVM)

```
pragma solidity ^0.4.24;

contract Owned {
    address public owner;
    string public message;

    modifier onlyOwner{
        require(owner == msg.sender);
        _;
    }

    constructor (string _msg) public {
        owner = msg.sender;
        message = _msg;
    }

    function changeOwner(address newOwner) public onlyOwner{
        owner = newOwner;
    }

    function getOwner() public view returns(address contractOwner){
        contractOwner = owner;
    }
}
```



Gas cost on Ethereum

Transaction fees on L1

total_cost = gas_price * gas_used

Where:

Gas limit: the maximum amount of gas that the transaction can use.

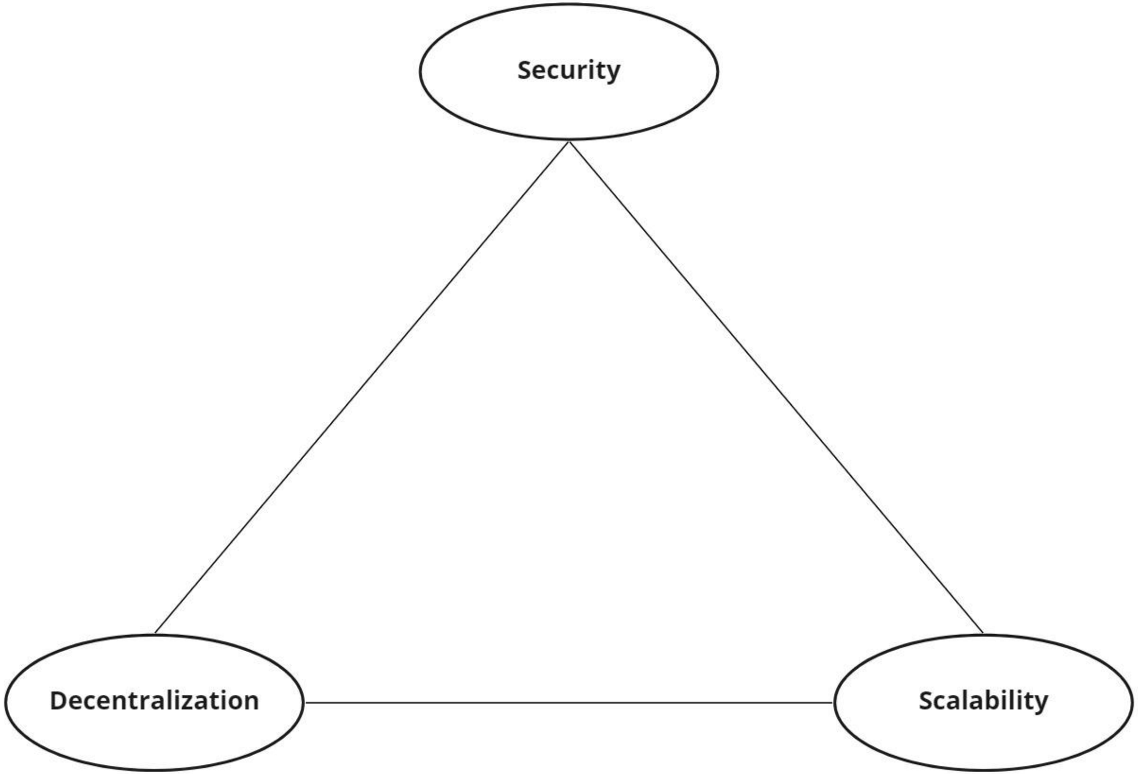
Gas price*: the cost (in wei) of each unit of gas spent and,

gas_used <= gas_limit

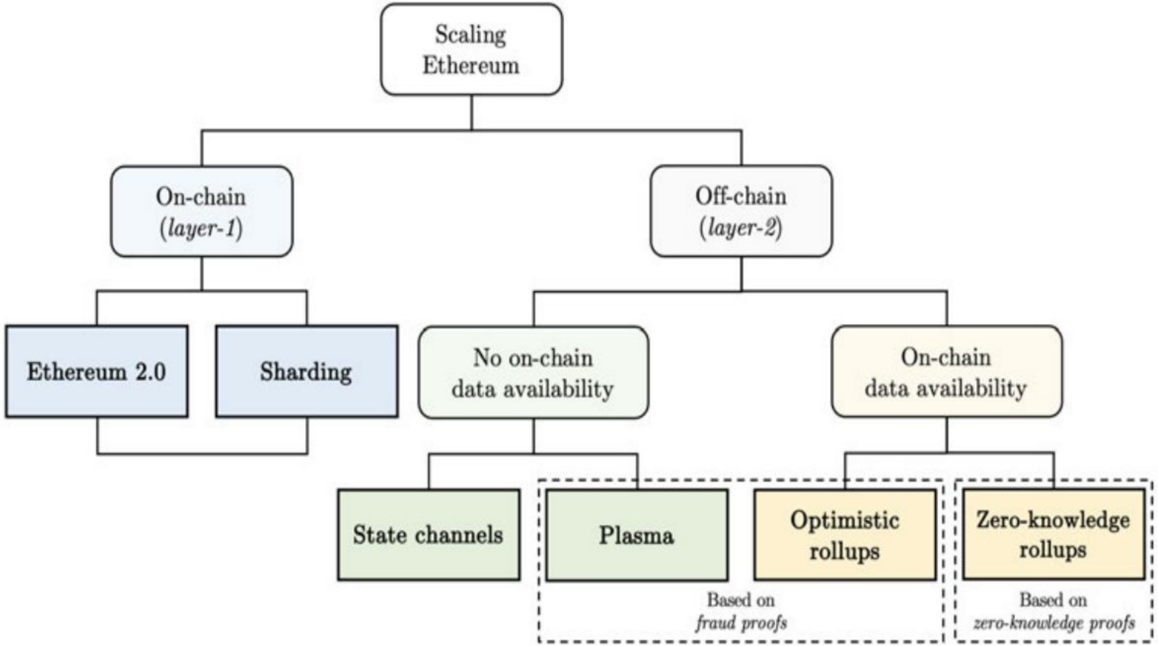
*Users can modify this value to trade off between speed and cost.

txstreet.com

Vitalik Buterin’s Scalability trilemma



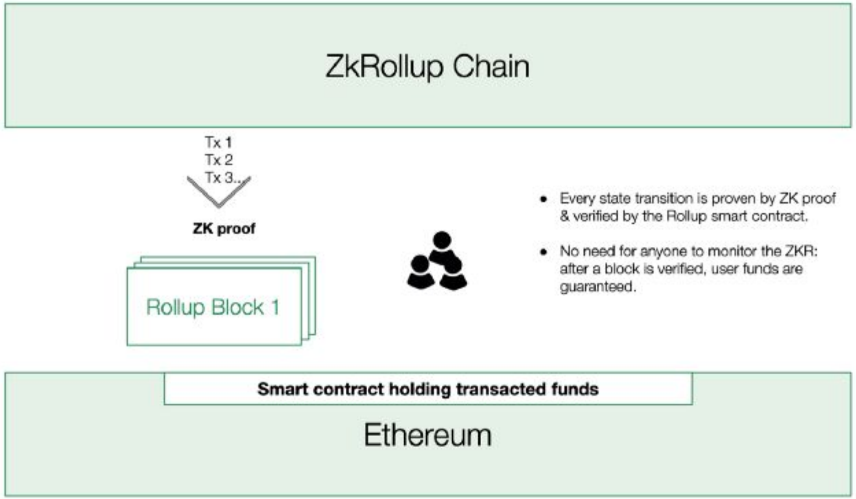
Scaling Ethereum



Source: “Scaling Public Blockchains: A comprehensive analysis of optimistic and zero-knowledge rollups”. Tobias Schaffner, January 2021.

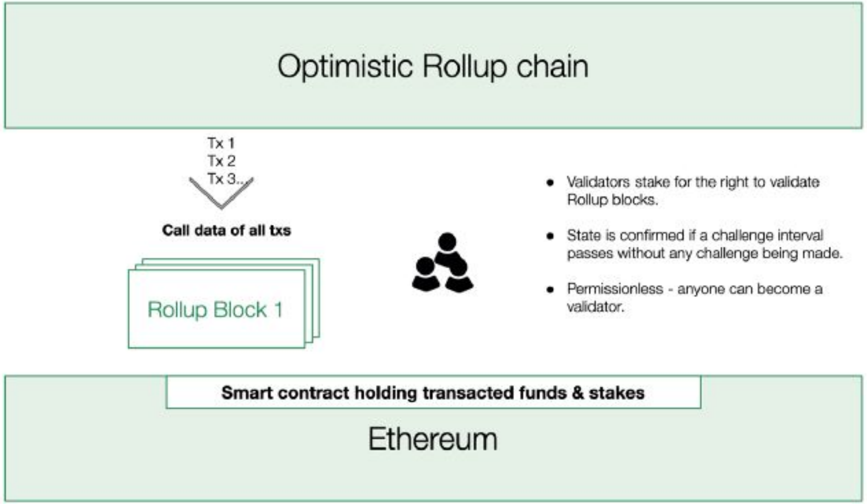
zk-Rollups

- Rely on cryptographic zero-knowledge proofs, which are instantly validated by smart contracts.
- Every state transition is proven by Zk proof & verified by the rollup smart contract.
- No need for anyone to monitor th zkR; after a block is verified user funds are guaranteed.



Optimistic rollups

- Depend on a dispute game run by active validators.
- Validators stake for the right to validate rollup blocks.
- State is confirmed if a challenge interval passes without any challenge being made.
- Permissionless - anyone can become a validator.



Source: Medium - “A Primer on Ethereum L2 Scaling Techniques”, retrieved September, 2021.

zk-Rollups

zk-SNARK, zero-knowledge Succinct Non-interactive ARgument of Knowledge

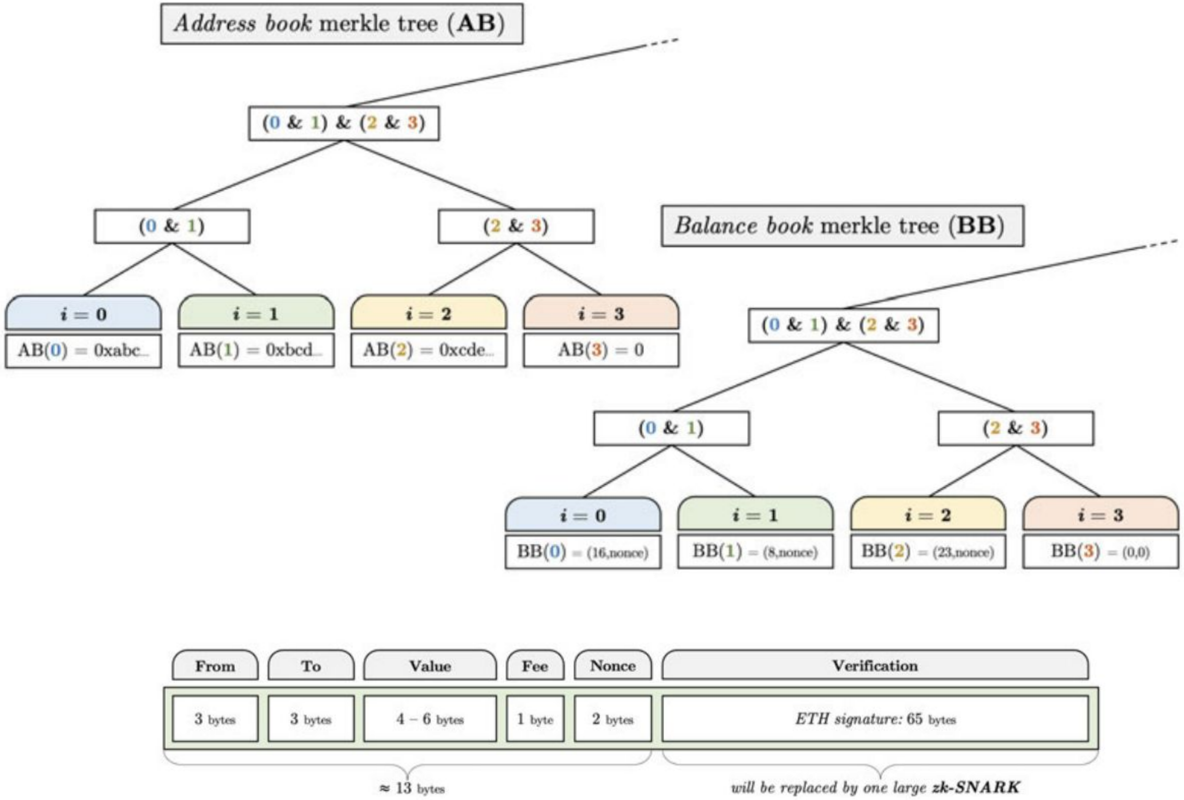
- ***transactors*** who want to transfer tokens.
- ***relayers*** who collect and process a set of transactions off-chain.

The zk-SNARKs are published on the Ethereum mainchain, along with the highly compressed transaction data via ***calldata***.

- zkSync is live with Curve Finance as the first resident dapp (Rinkeby Tesnet).

Drawback: The generation of such proofs is computationally intensive and therefore relatively time-consuming.

Compression



Source: “Scaling Public Blockchains: A comprehensive analysis of optimistic and zero-knowledge rollups”. Tobias Schaffner, January 2021.

One zk-Rollup transaction

From	To	Value	Fee	Nonce	Verification
174590	3420	40’000	110	XYZ	*prev state:* 0x73e...
97590	8030202	15’000	90	XYZ	*new state:* 0x3f5...
6289	489304	3’200	90	XYZ	*zk-SNARK*: 300 bytes
⋮	⋮	⋮	⋮	⋮	
523093	592	55’000	80	XYZ	

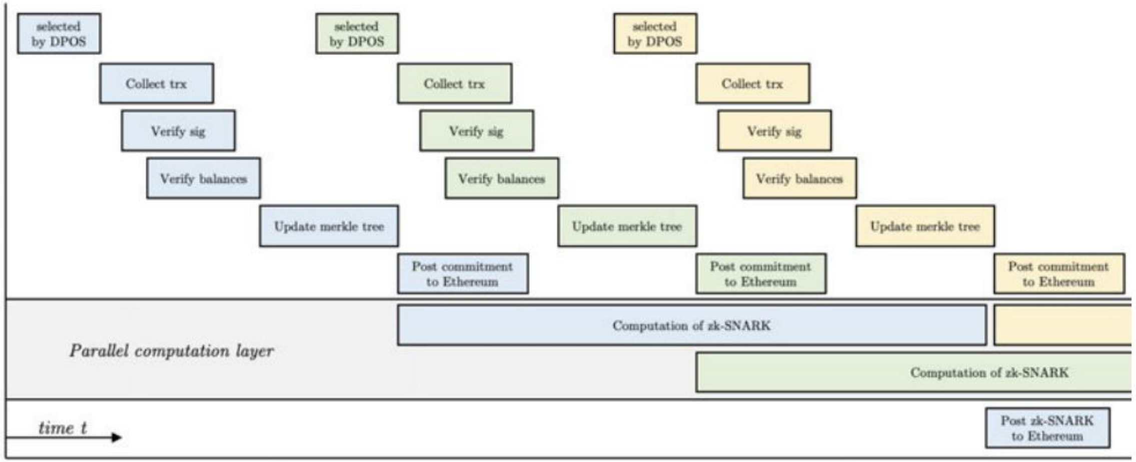
		gas limit		zk-SNARK		overhead		trx cost		new block	
Ethereum trx (PRE)	(	8’000’000 gas	−	0	−	0	) /	21’000 gas*	/	15 sec	≈ 25 trx/sec
Ethereum trx (POST)	(	12’500’000 gas	−	0	−	0	) /	21’000 gas*	/	15 sec	≈ 40 trx/sec
zk-rollups trx (PRE)	(	8’000’000 gas	−	600’000 gas	−	50’000 gas	) /	884 gas**	/	15 sec	≈ 550 trx/sec
zk-rollups trx (POST)	(	12’500’000 gas	−	600’000 gas	−	50’000 gas	) /	208 gas***	/	15 sec	≈ 3’800 trx/sec

PRE *Istanbul hard fork*
POST *Istanbul hard fork*

* base gas fee for a simple transaction
** $(3 + 3 + 4 + 1 + 2) \cdot$ gas cost per byte *for calldata* $=$ 13 bytes $\cdot$ 68 $=$ 884 gas
*** $(3 + 3 + 4 + 1 + 2) \cdot$ gas cost per byte *for calldata* $=$ 13 bytes $\cdot$ 16 $=$ 208 gas

Source: “Scaling Public Blockchains: A comprehensive analysis of optimistic and zero-knowledge rollups”. Tobias Schaffner, January 2021.

A multiple-relayer model for zk-rollups



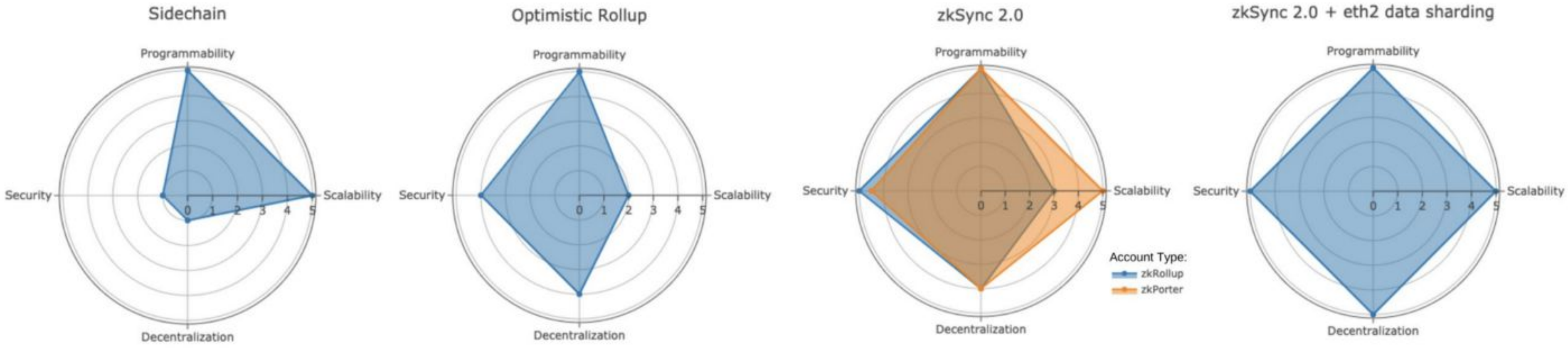
Matter Labs's *commit - verify* approach

Source: "Scaling Public Blockchains: A comprehensive analysis of optimistic and zero-knowledge rollups". Tobias Schaffner, January 2021.

zkSync 2.0: Hello Ethereum!

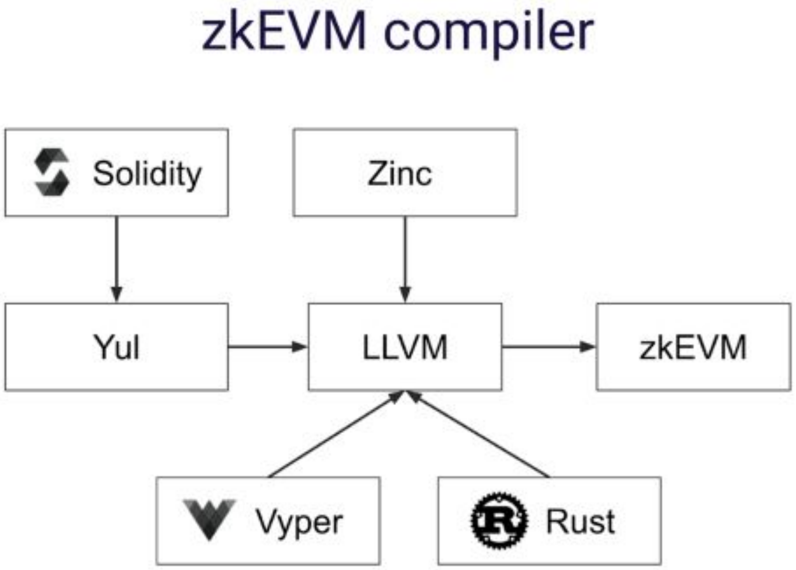
zkEVM: The engine powering our EVM-compatible zkRollup, the only solution with L1 security and solidity smart contract support.

zkPorter: An off-chain data availability system with 2 orders of magnitude more scalability than rollups.



Zinc

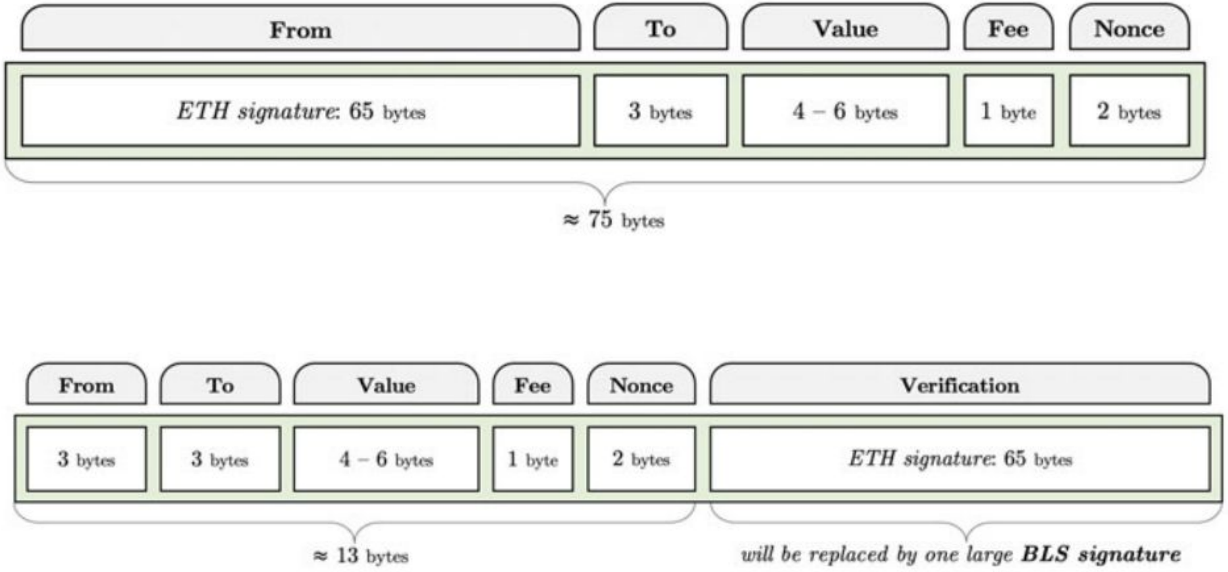
- Zinc is a programming language which can be used to develop smart contracts.



Optimistic Rollups

- Combines on-chain data availability plus a fraud proof mechanism.
- Tackle the main drawbacks of zk-rollups:
  - The very long generation process of zk-SNARKs
  - Lack of support for commonly used smart contract standards.
- The aggregators publish the transaction data and the state root onto the Ethereum blockchain through calldata
- Drawbacks
  - Trade-off some degree of scalability
  - Long withdrawal period

Optimistic rollup transaction data



Optimistic rollup transaction data using BLS signature

Optimistic rollups - Theoretical throughput

		gas limit		state root		signature		trx cost		new block	
Ethereum trx	(	8'000'000 gas	−	0	−	0	) /	21'000 gas*	/	15 sec	≈ 25 trx/sec
Ethereum trx	(	12'500'000 gas	−	0	−	0	) /	21'000 gas*	/	15 sec	≈ 40 trx/sec
OR's *with sig*	(	8'000'000 gas	−	2176 gas	−	0	) /	5'100 gas**	/	15 sec	≈ 100 trx/sec
OR's *with sig*	(	12'500'000 gas	−	512 gas	−	0	) /	1'200 gas***	/	15 sec	≈ 700 trx/sec
OR's *with BLS*	(	8'000'000 gas	−	2176 gas	−	113'000 gas	) /	884 gas****	/	15 sec	≈ 600 trx/sec
OR's *with BLS*	(	12'500'000 gas	−	512 gas	−	113'000 gas	) /	208 gas*****	/	15 sec	≈ 3'500 trx/sec

PRE *Istanbul hard fork*
POST *Istanbul hard fork*

*	base gas fee for a simple transaction				
**	$(65 + 3 + 4 + 1 + 2)$ · gas cost per byte *for calldata*	=	75 bytes · 68	=	5'100 gas
***	$(65 + 3 + 4 + 1 + 2)$ · gas cost per byte *for calldata*	=	75 bytes · 16	=	1'200 gas
****	$(3 + 3 + 4 + 1 + 2)$ · gas cost per byte *for calldata*	=	13 bytes · 68	=	884 gas
*****	$(3 + 3 + 4 + 1 + 2)$ · gas cost per byte *for calldata*	=	13 bytes · 16	=	208 gas

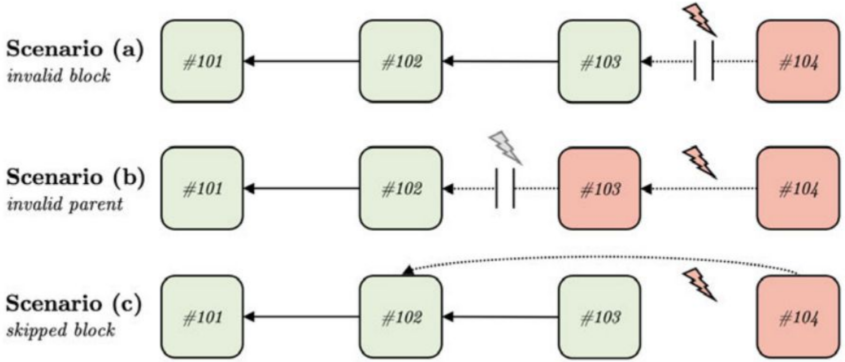
Optimistic Virtual Machine (OVM)

- Plasma Group (2019) → Ethereum Optimism
- Allows an implementation of arbitrary smart contract logic natively on layer 2.
- Arbitration court that checks the validity of fraud proofs
- Components:
    - Transpiler algorithm
    - Safety checker
    - Execution Manager

Aggregators (or Sequencer) in Optimism

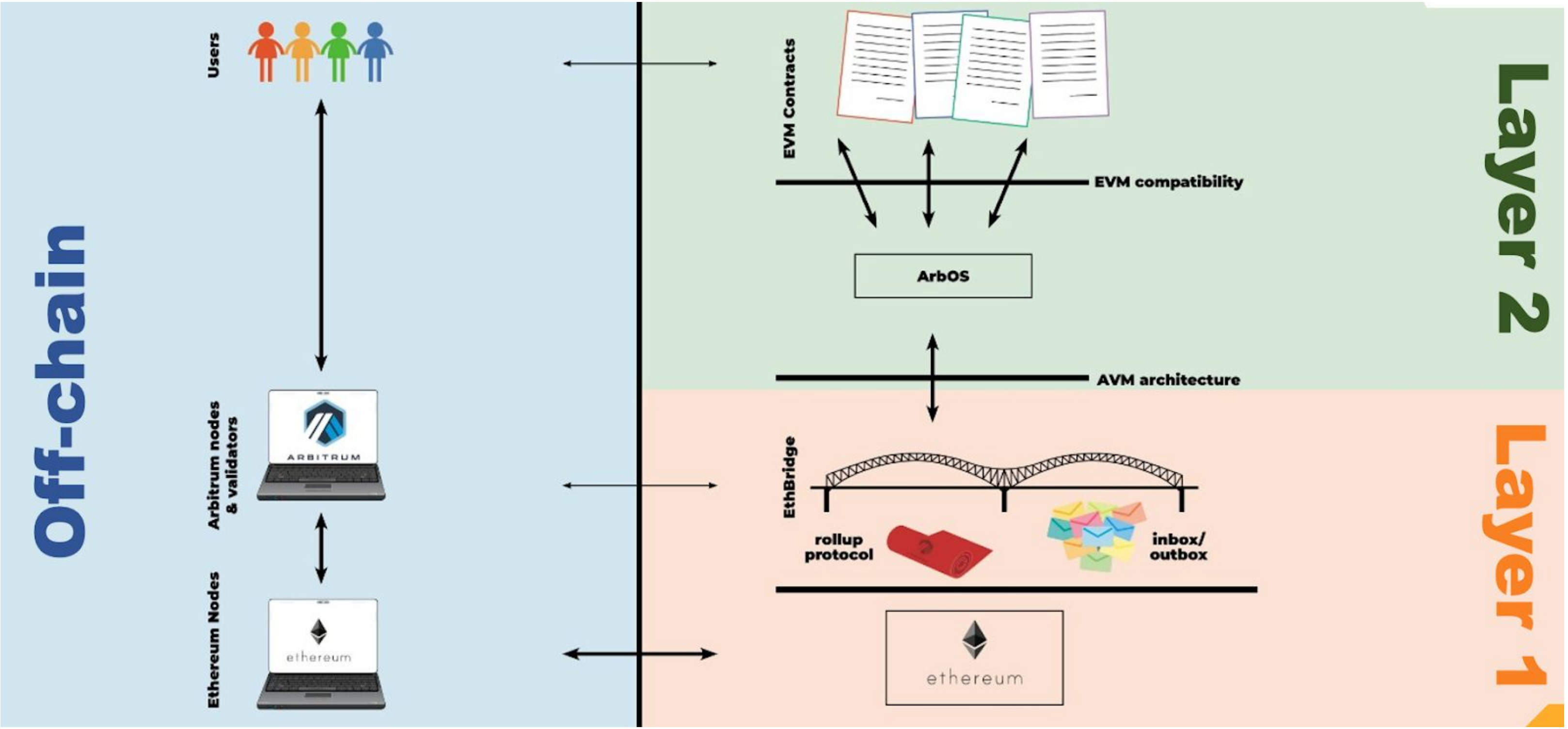
The *head state* has to be ***valid***, ***live*** and ***available***.

- Aggregators are incentivized to verify the **validity** of every block.
- The head state of the optimistic rollup chain is always live, as long as there is at least one ***non-censoring*** aggregator.



Source: "Scaling Public Blockchains: A comprehensive analysis of optimistic and zero-knowledge rollups". Tobias Schaffner, January 2021.

Arbitrum Architecture



Source: https://developer.offchainlabs.com/docs/inside_arbitrum

Arbitrum Full Nodes

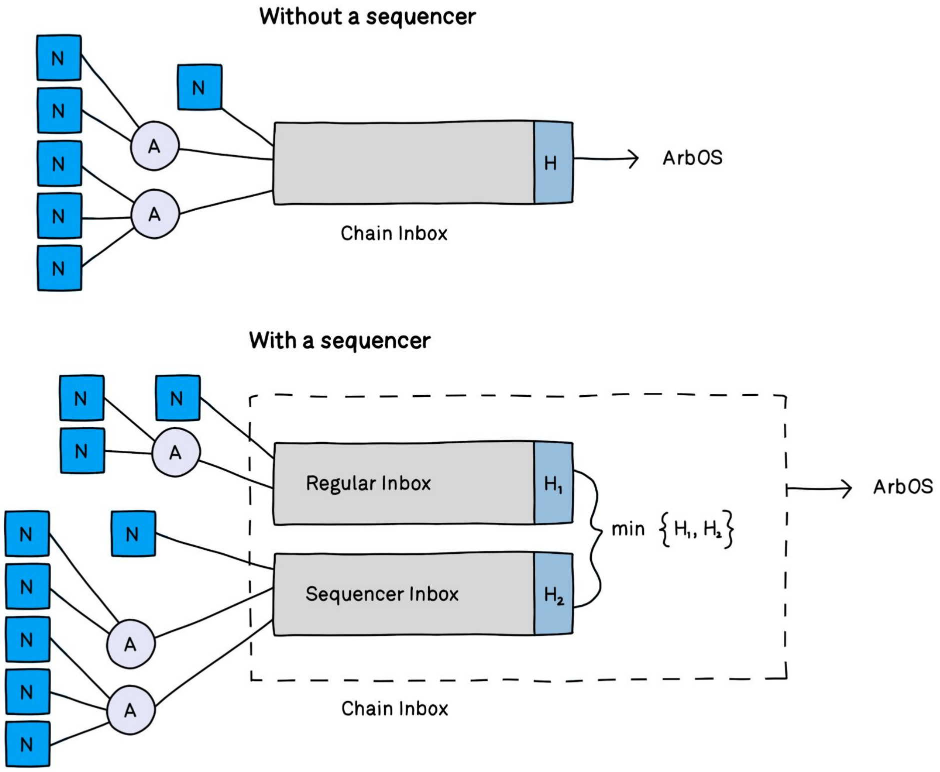
- Track the state of the chain and allow others to interact with the chain
- Can serve as an ***aggregator***.
- Full nodes compress transactions to minimize L1 calldata cost.
- A full node typically incorporates both **compression** and **aggregation**.
- It will submit a batch of compressed transactions to the chain’s inbox
- Sequencer mode is optional.

Arbitrum - Aggregators

- Responsible for packaging together multiple client transactions into a single message to be submitted to the chain’s inbox as a single unit.
- There is no limit on how many aggregators can exist, nor on who can be an aggregator.
- Submitting a batch is permissionless, so any user can, submit a single transaction "batch" if necessary.
- Arbitrum allows to collect fees from users, to reimburse the aggregator.

Source: https://tracer.finance/radar/arbitrum-in-under-10/

Sequencer Mode



Sequencer Transaction Fees (Optimism)

Fees for sequencer transactions considering: Data cost & Execution cost.

total_cost = ((tx_size * d_price) + (exec_gas * exec_price))

- ***tx_size*** is the size (in bytes) of the serialized transaction that will be published to Layer 1.
- **d_price** current cost of publishing data to Layer 1.
- ***exec_gas*** is the amount of gas that the transaction can use.
- ***exec_price*** is the cost (in wei) per unit gas allotted (much like gas_price on L1).

Source: https://community.optimism.io/docs/developers/fees.html

Encoding Sequencer transaction costs

In Optimism:

gas_price is fixed to a value of 0.015 gwei.

Next, when you call eth_estimateGas, the L2 node computes:

gas_limit = (tx_size * d_price) + (exec_gas * exec_price) / gas_price

Source: https://community.optimism.io/docs/developers/fees.html

Verifier’s Dilemma

1. If the system’s incentives work as intended, nobody will cheat.
2. If nobody cheats, then there’s no point in running a verifier because you make no money from operating it.
3. Since nobody runs a verifier, there’s eventually an opportunity for a ***sequencer*** to cheat
4. The sequencer cheats, the system no longer functions as intended.

Questions?