# Polygon zkEVM

===============

Polygon zkEVM is a decentralized Ethereum Layer 2 scalability solution that uses cryptographic zero-knowledge proofs to offer validity and quick finality to off-chain transaction computation, also known as a **ZK-Rollup**.

Polygon zkEVM

1.handles state transitions caused by Ethereum Layer 2 transaction executions

2.creates validity proofs that attest to the accuracy of these off-chain state change calculations by utilising zero-knowledge features.

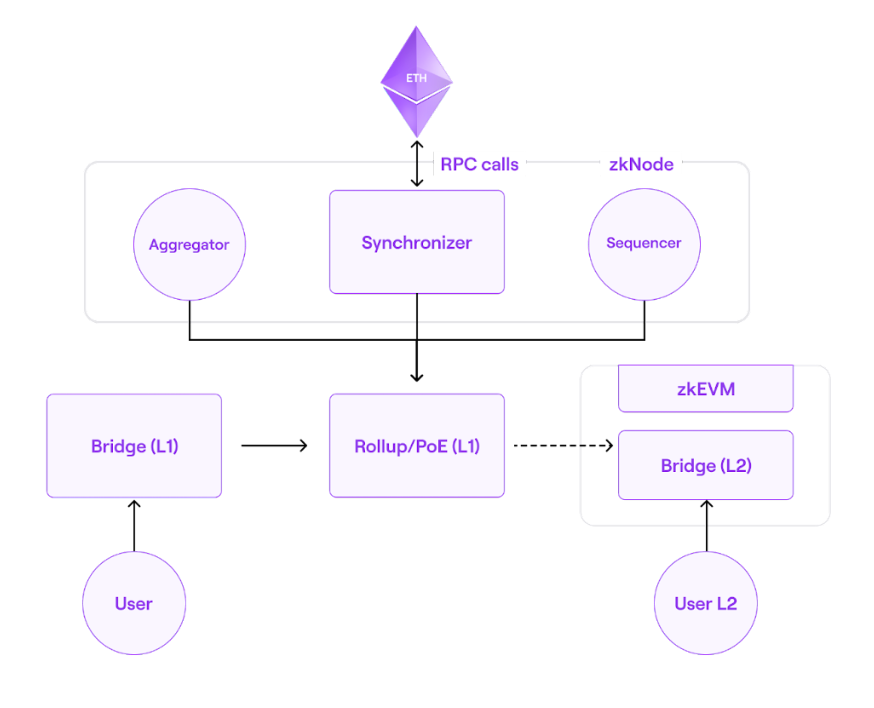
EVM equivalence

=============

EVM-equivalence means that you can deploy your existing solidity code without going through any extra steps to compile your code to get it to work on this network.

# Polygon zkEVM Architecture

===========================



The major components of zkEVM are:

1.Proof of Efficiency (PoE) Consensus Mechanism

2. zkNode

3.zkProver

4.LX-to-LY Bridge

1.Proof of Efficiency (PoE) Consensus Mechanism

=======================================

The earlier version, **Polygon Hermez 1.0**, was based on the **Proof of Donation (PoD)** consensus mechanism.**Proof of Efficiency (PoE)** consensus mechanism leverages the experience of the existing **PoD** in v1.0 and add support for the permissionless participation of multiple coordinators to produce batches in L2.

2. zkNode

========

zkNode is the software needed to run any zkEVM node. It is a client that the network requires to implement the Synchronization and govern the roles of the participants (Sequencers or Aggregators). Polygon zkEVM participants will choose how they participate:

* As a node to know the state of the network, or
* As a participant in the process of batch production in any of the two roles: **Sequencer** or **Aggregator**

The two permissionless participants of the zkEVM network are: **Sequencers** and **Aggregators**.

**Sequencer**

* Collect transactions and publish them in a batch
* Receive fees from the published transactions
* Pay L1 transaction fees + MATIC (depends on pending batches)

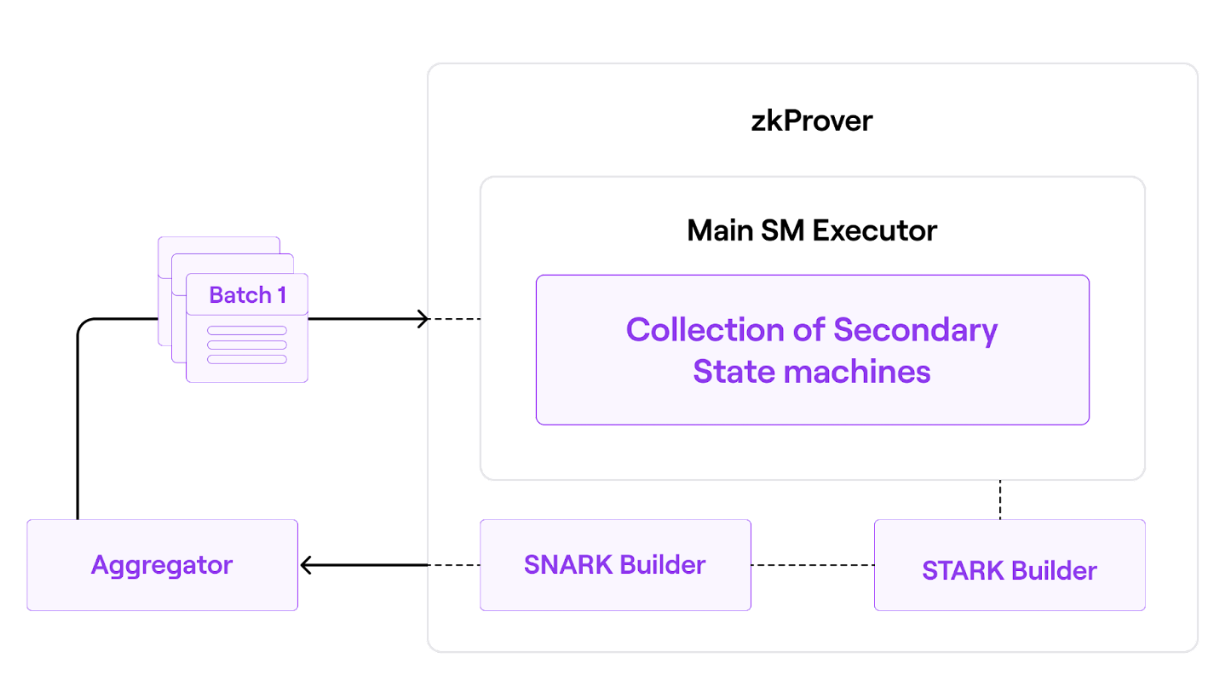
**Aggregators**

* Process transactions published by Sequencers
* Build zkProof
* Receive MATIC from Sequencer

3. zkProver

============

zkEVM employs advanced zero-knowledge technology to create validity proofs. It uses a **zero-knowledge prover (zkProver)**.Every **Aggregator** will use this zkProver to validate batches and provide Validity Proofs.



4.LX-to-LY Bridge

==========

An **LX-LY bridge** is a Smart Contract that lets users transfer their assets between two layers, LX and LY. It is a combination of two smart contracts, one deployed on one chain and the second on the other.

## 

## 

## Verifier

=========

Verifier is a Smart Contract which is able to verify any ZK-SNARK cryptographic proof. This SNARK Verifier proves the validity of every transaction in the batch. It is the key entity in any zk-Rollup architecture for the reason that it verifies the correctness of a Proof ensuring a valid state transition.

## Transaction Process of Polygon zkEVM

=======================================

Before getting into a transaction flow in L2, users need some funds to perform any L2 transaction. In order to do so, users need to transfer some ether from L1 to L2 through the zkEVM Bridge dApp.

* Bridge

======

* Deposit ether
* Perform claim on L2 and receive the funds
* L2 Transactions

=============

* User initiates tx in a Wallet (e.g. Metamask) and sends it to a Sequencer
* It gets finalized on L2 once Sequencer commits to add his transaction
* Transaction has finalized on L2, but not on L1 (simply put, L2 state is not yet on L1). Also known as **Trusted State**
* Sequencer sends the batch data to L1 smart contract, enabling any node to synchronize from L1 (known as **Virtual State**)
* Aggregator will take pending transactions to be verified and build a Proof.
* Once the Proof is validated, user's transactions will attain L1 finality (important for withdrawals). This is called the **consolidated state**.

zkEVM characteristics

=================

**Permissionless-ness** **-** The aim is to allow anyone with the zkEVM software to participate in the network.The consensus algorithm will give everyone the opportunity to be a Sequencer or an Aggregator.

**Decentralization-** The aim is to ensure that there is no censorship and that no one party can control the network.

**Security -**Since this is an L2 solution, most of the security is inherited from Ethereum.Smart contracts will ensure that everyone who executes state changes does so appropriately, creates a proof that attests to the validity of a state change, and makes validity proofs available on-chain for verification.

**zkEVM implementation strategies**

**=================================**

* PoE, incentivizes the most efficient aggregators to participate in the proof generation process.
* all computations done off-chain while keeping only the necessary data and zk-proofs on-chain.
* Utilisation of specialised cryptographic primitives to speed up computations and minimise proof sizes such as

1. Using zero-knowledge tools such as **zk-STARKs** for proving purposes; these proofs are very fast though they are bigger in size.
2. Since zk-STARKS are bigger in size **zk-SNARK** is used to attest to the correctness of the zk-STARK proofs. These zk-SNARKs are, in turn, published as the validity proofs to state changes. This helps in reducing the gas costs from 5M to 350K.