Economic Whitepaper: Operating Full Nodes on Decentralized Layer-2 ZK-Rollups

Tal Derei, Xinhao Liu, Steve Ye

May 2022

Abstract

A full node is a program that validates transactions and confirms blocks in a blockchain network. Running full nodes on Layer-1 blockchains has proven to be extremely crypto-economically profitable, for instance proof-of-work mining and staking on the beacon chain in the case of Ethereum. This paper outlines capturing the future potential of similar or greater returns on Layer-2s. Layer-2 (L2) networks represent scaling solutions that derive their security from the underlying Layer-1 base-chain. *ZK-Rollups* specifically are generally centralized, consisting of a single non-censorship resistant node operator. This introduces the risk of denial-of-service attacks if the operator goes offline. The progressive decentralization of ZK-Rollups in the future necessitates independent validators participating in these networks by running full L2 nodes. Incentivizing active participation in the network requires launching a token and rewarding participants in exchange for validating transactions and generating zero-knowledge proofs. This is the direction these rollups are heading towards. We're frontrunning the Layer-2 staking opportunity in the zero-knowledge scaling space by building **Faro**, a liquid-staking protocol for ZK-Rollups. The protocol design will be described in a separate technical whitepaper. We motivate the economic considerations in the following sections below.

1 Introduction

Layer-2 ZK-Rollup ecosystems are growing at an accelerated rate, bringing the "rollup-centric" future envisioned by Vitalik closer to fruition [1]. With that, a larger pool of transaction fees will be paid out to full node operators. For some context, Ethereum miners netted revenues exceeding $21 billion in 2021 [2]. Layer-2s revenues will similarly inflate as they become the primary consumers of Ethereum blockchain demand over time. In contemplating this untapped market, the more lucrative the opportunity presents itself to be.

2 Ecosystem Components

2.1 ZK-Rollups

"General-Purpose ZK-Rollups" that support general EVM computations, like arbitrary smart contract executions, on Layer-2 [3] represent a class of ZK-Rollups called *zkVMs / zkEVMs*. Extracting value from these networks has remained unexplored for a multitude of reasons. These networks currently operate on beta testnets and may lack a native token necessary to facilitate their decentralization. Eventually these zkEVMs will launch on mainnet and decentralize their sequencers. Consequently, zkEVMs will bifurcate into two types of networks: networks *with* an existing token, and networks *without* an existing token.

1. Networks *with* an existing token include the Polygon stack: **Polygon zkEVM, Polygon Hermez, Polygon Zero, Polygon Miden [4].** The compounding network effects brewing in the Polygon ecosystem will cumulatively boost their user-base and rate of adoption. This is one consequence of using the same token to promote seamless interoperability across ZK-Rollups. This incurs a network risk though. Network instability on a single ZK-Rollup risks impacting the MATIC token price and thus adversely affecting all Polygon protocols.
2. Networks *without* an existing token include: **Starknet (Starkware), zkSync 2.0, Aztec, and Scroll**. These networks will eventually launch tokens to facilitate their decentralization, thereby generating hype and increasing the adoption of these rollups, which in-turn will increase rollup provider fees in the short-term. ZK-Rollup networks using different tokens further eliminate the risks, shielding the other networks if something goes wrong on one of them.

Therefore, from a profit-centric perspective, we infer that all these networks will experience higher rates of adoption and growth curves from their current levels. We describe the field of zkEVM networks in further detail below.

Starknet

Starkware is building *Starknet*, a STARK-based L2 ZK-Rollup that supports general-computation over Ethereum [5]. The core backend is Cairo VM, written in a Turing-Complete programming language & framework for producing STARKs called *Cario*. This

supports proofs for general computation instead of crafting complex, application-specific circuits.

The STARK software **stack** incorporates:

1. **Cairo**: General-purpose Turing-complete programming language.
2. **Cairo Compiler**: Transforming Cairo programs into Cairo bytecode.
3. **Cairo VM**: Executes the Cairo bytecode and produces an execution trace of the program.
4. **Prover and Verifier**: Generates and verifies a STARK proof of validity for the execution trace.
5. **Warp Transpiler**: Transforms Solidity to Cairo code.
6. **StarkEx**: Scalability engine (~ compression service) for powering app-specific ZK-Rollups, for example dYdX and ImmutableX.

Starkware's open-source ethSTARK prover [6] performs 20x faster than any other cpu prover. Starknet currently processes 300K transactions in a single STARK proof on mainnet, achieving high gas efficiency with 315 gas/tx. This is orders of magnitude more affordable than transactions on Ethereum's mainnet. The aforementioned presents an enticing case for operating full nodes on Starknet.

zkSync 2.0

MatterLabs is building zkSync 2.0, a zkVM representing a newly designed virtual machine to maintain solidity compatibility on Layer-2 [7].

The zkSync 2.0 software **stack** incorporates:

1. **Redshift***:* zkSNARK Proof System, using PLONK (universal SNARK) with custom gates and lookup tables (TurboPLONK/UltraPLONK) on Ethereum's BN-254 elliptic curve.
2. **Recursion**: Implements Recursive SNARKs for efficiency, by aggregating proofs in a tree-like structure, enabling multiple proofs to be recursively composed into a single SNARK proof.
3. **Zinc**: Framework for developing zero-knowledge smart contracts and SNARK circuits.

4. **ZincVM**: Core backend is zincVM, a SNARK-friendly virtual machine written in Zinc. ZincVM includes a Zinc compiler, with LLVM compiler optimizations.
5. **zkPorter:** Puts data availability, i.e. transaction data required to reconstruct the state, off-chain. This data availability layer reports over 20K TPS and is secured by a Proof-of-Stake (PoS) consensus mechanism.

zkSync has seen incredible growth recently in the ZK-Rollup space. The aforementioned presents an enticing case for operating full nodes on zkSync.

Aztec

Aztec are the inventors of the PLONK cryptographic proving system, a universal SNARK construction with Lagrange-bases over Ethereum's BN-254 elliptic curve [8]. TurboPLONK and UltraPLONK extend this construction by introducing custom gates and lookup tables [9][10].

The Aztec software **stack** incorporates:

- **Zk-zk rollup:** Privacy-preserving ZK-Rollup.
- **zkSNARK Two-Proof Construction System:** Proof system consisting of a 'Privacy' circuit and 'Rollup' circuit. **Privacy Circuit** proves the correctness of a single private transaction, using client hardware. **Rollup Circuit** validates the correctness of a batch of 128 privacy proofs on the server-side.
- **Recursion**: Verifying a SNARK proof (lower level, privacy SNARK) inside another SNARK circuit (upper level, Rollup SNARK).
- **NOIR**: Programming language for private smart contracts. Smart contracts are encoded as SNARK circuits rather than EMV-compatible code.

Aztec has experienced unparalleled growth in the private rollup space with the advent of their zk.money and Aztec Connect product offerings. The aforementioned presents an enticing case for operating full nodes on Aztec.

Scroll

Scroll is an Ethereum Virtual Machine-based Layer 2 ZK-Rollup. The team published PipeZK, an efficient pipelined architecture for accelerating zkSNARKs [11] on ASICs.

The Scroll software **stack** incorporates:

- EVM-equivalent virtual machine, targeting bytecode-level compatibility. Their zkEVM follows the definition and standard outlined in the EVM yellow paper.
- Decentralized prover system that generates multiple blocks in parallel and generates a proof for each block.

Scroll focuses on hardware acceleration, claiming to have built the fastest GPU prover that is 5x-10x faster than Filecoin's implementation **[12]**. Scrolls is leading the cutting-edge research for proof acceleration. The aforementioned presents an enticing case for operating full nodes on Scroll.

Polygon zkEVM

Polygon zkEVM is a fully open-source EVM-equivalent ZK L2 **[13]**.

The Polygon zkEVM software **stack** incorporates:

- **Proof of Efficiency** consensus mechanism.
- **zkEVM Node:** Go implementation of a sequencer node that operates the network.
- **zkProver**: C++ implementation of recursive STARKs and using a SNARK proof to prove the correctness of the STARK proofs.
- **Code-level optimizations**: Goldilocks field for efficient 256-bit operations, keccak circuits computed in parallel, poseidon-hash Merkle tree for storage.

Polygon zkEVM's permissionless and open-source nature, in association with the technical contributions from the Polygon Hermez and Polygon Zero teams, makes it a promising choice. The aforementioned presents an enticing case for operating full nodes on Polygon zkEVM.

Polygon Hermez

Polygon zkEVM is a ZK-Rollup providing complete EVM opcode equivalence **[14]**.

The Polygon Hermez software **stack** incorporates:

- **zkEVM**: Based on bottom-up construction of rebuilding all opcodes for layer-2.

- **Proof-of-Donation (Auction-Model):** Consensus mechanism, which is replaced by Proof-of-Efficiency.
- **Bytecode-level Compatible**: New set of assembly codes to express each opcode and then prove the execution of the new assembly over their own defined state machine. Recursive STARKs are then used to prove the execution trace of the state machines.

Polygon Hermez includes world class researchers and technical builders. The aforementioned presents an enticing case for operating full nodes on Polygon Hermez.

Polygon Zero

Polygon Zero is a recursive SNARK-based zkEVM **[15]**.

The Polygon Zero software **stack** incorporates:

- **Proof System (Plonky2)**: Recursive SNARKs w/ PLONK and FRI Polynomial Commitments.
- **Goldilocks Field**: Field represents nice mathematical structures for doing field arithmetic efficiently on modern CPUs.
- **Optimizations**: FRI has a parameter called the blowup factor that describes a tradeoff between proof size and proof time. They recursively aggregate proofs in a tree-like structure, and then compress it further using the blowup factor.

Polygon Zero is skilled at implementing advanced low-level optimizations, achieving 100x recursion performance for proof generation, going from 10s to 170ms with 2^12 gates on commodity hardware. The aforementioned presents an enticing case for operating full nodes on Polygon Zero.

Polygon Miden

Polygon Miden is the first open-source general-purpose, STARK-based L2 zk-Rollup **[16]**.

The Polygon Miden software **stack** incorporates:

- **Miden VM:** Core backend that improves on Distaff VM (zero-knowledge virtual machine written in Rust) by replacing the underlying proving system with Winterfell (STARK prover).

- **Miden Assembly:** Compile solidity-based smart contracts directly into Miden Assembly - the native language of Miden VM.
- **AIR Design:** Generate proofs for any arbitrary computation without requiring specific AIRs for each computation. For any program executed on Polygon Miden, a STARK-based proof of execution is automatically generated.

Polygon Miden has demonstrated experience in building advanced compiler designs and creating an efficient STARK-based VM. The aforementioned presents an enticing case for operating full nodes on Polygon Miden.

2.2 Full-Node Types and Fee Structures

ZK-Rollups will feature two types of nodes:

- **Sequencers** are equivalent to full nodes on Ethereum: ordering transactions, executing transactions, and batching multiple transaction executions into blocks. Centralized sequencers can in theory front-run transactions and extract miner extractable value (MEV) on Layer-2.
- **Builders / Provers** generate the computationally heavy zero-knowledge proofs that are posted on Layer-1, alongside the call-data for data availability. Optimized precompiled smart contracts on Ethereum perform the lightweight proof verification on-chain.

Running a sequencer will demand less computational resources, yielding lower rollup fees. We suspect most of the profit will be generated by builders / provers. The specific fee structures are yet to be determined until the open-source sequencer and prover software is released.

2.3 Hardware and Cloud Infrastructure Costs

Operating full nodes requires a substantial amount of startup capital to purchase the hardware and compute resources necessary to be competitive in this landscape. With the rise in efficiency and lower amortized costs of cloud computing services, we must consider the feasibility of operating on Amazon Web Services (AWS) as a viable business model. Sourcing native hardware infrastructure serves as another model. The trade-offs between employing cloud-based and native hardware infrastructures and their cost-structures are explored in-

depth below. The high-level profitability models are calculated by the following mathematical equations, respectively:

Profit Margin = Rollup Transaction Fees – Cloud Server Costs – L1 Relayer Gas Costs

vs.

Profit Margin = Rollup Transaction Fees – Hardware Costs – L1 Relayer Gas Costs

The profit margin is a function of several dependent variables: *rollup transaction fees* represent network transaction fees paid out to rollup operators for operating full nodes. *Cloud-based server / hardware* costs represent the startup capital costs for purchasing computing resources. *L1 relayer gas* costs represent the cost of sequencers submitting batches of calldata on the layer-1 chain. To summarize, a net positive profit margin relies on rollup transaction fees exceeding these costs.

Estimating the computation costs is challenging, but we can make inferences based on existing application-specific ZK-Rollups like Loopring. Loopring boasts over $5B in total L2 trading volume, spending ~$2.8M ($0.57 onchain cost / tx, 4.8 million txs) in on-chain costs for submitting batches **[17]** with ~$9M in revenues **[18].** This makes their rollup profitable. With regards to hardware costs, Loopring developed production-grade circuits with constraint sizes in the millions. Loopring's largest circuit consists of 64M (2^26) constraints for 1024 trades, requiring 104 seconds to prove using just 55GB of memory on a 16-core CPU **[19]**. At current consumer hardware prices, a 16-core CPU + 64 GB of RAM costs around $1000. As GPU-based provers with presumably higher performance become mainstream, these costs will increase to around $2000 for high-end 3000-series Nvidia GPUs.

2.4 Circuit Costs

As these application-specific ZK-Rollups transition to general-purpose SNARK/STARK-based zkEVMs, the circuit costs will proportionally change. The computational costs associated with proving a circuit is log-linear, proportional to the number of **n** constraints the circuit is bounded by. We must watch for whether porting an application specific rollup to a zkEVM program increases or decreases this number to make a fair judgment.

This introduces the following potentialities: (1) number of constraints may increase when moving to zkEVMs since these application specific rollups are specialized (which means

they're more likely fine-tuned and optimized to run their specific workload) like an ASIC, or (2) the number of constraints may decrease since the compilers and execution engines/VM are more efficient in transforming programs into ZK optimized representations. In the case (1), the main question from a profit-margin perspective is whether the spread-difference is large enough to counteract the increase in the number of constraints. As more application specific programs port to zkEVM programs, the network effects will multiply and the amortized costs per transaction will decrease. This means the "spread" between the transaction fees paid to the rollup providers and the cost to post the proof on-chain increases. The spread needs to be large enough to account for an increase in the number of constraints, which makes proof generation more expensive. In the case of (2), it's cheaper to generate and relay a proof on L1, and the same network effects still apply.

2.5 GPUs and RDMA

ZK proof generation is notoriously memory intensive, possibly exceeding the host memory of a given machine. RDMA allows a machine to access the main memory from a remote host without involving either machine's CPU. RDMA would allow the proving system to access a remote machine's memory as if it were local memory. Exploiting optimizations enabling ZKP to interface with RDMA on GPUs to become a low-cost zk provider serves as another business model in the future [20].

References

[1] Vbuterin, et al. “A Rollup-Centric Ethereum Roadmap.” *Fellowship of Ethereum Magicians*, 2 Oct. 2020. Available: https://ethereum-magicians.org/t/a-rollup-centric-ethereum-roadmap/4698.

[2] “Ethereum Archives.” Available: *The Block*,https://www.theblockcrypto.com/data/on-chain-metrics/ethereum.

[3] Team, DeGate. “An Article to Understand zkEVM, the Key to Ethereum Scaling.” *Medium*, DeGate, 23 Sept. 2021. Available: https://medium.com/degate/an-article-to-understand-zkevm-the-key-to-ethereum-scaling-ff0d83c417cc.

[4] “Polygon Takes a Major Lead in ZK Rollups; Welcomes Mir, a Groundbreaking Zk Startup in a $400m Deal - Polygon: Blog.” *Polygon*, 18 May 2022. Available: https://blog.polygon.technology/polygon-takes-a-major-lead-in-zk-rollups-welcomes-mir-a-groundbreaking-zk-startup-in-a-400m-deal/.

[5] StarkWare. “On the Road to StarkNet: A Permissionless Stark-Powered L2 Zk-Rollup.” *Medium*, StarkWare, 12 Dec. 2021. Available: https://medium.com/starkware/on-the-road-to-starknet-a-permissionless-stark-powered-l2-zk-rollup-83bE53640880.

[6] Starkware-Libs. “Starkware-Libs/Ethstark.” *GitHub*. Available: https://github.com/starkware-libs/ethSTARK.

[7] Matter Labs. “ZkSync 2.0: Hello Ethereum!” *Medium*, Matter Labs, 16 Oct. 2021. Available: https://blog.matter-labs.io/zksync-2-0-hello-ethereum-ca48588de179.

[8] “Understanding Plonk.” *Vitalik Buterin's Website*. Available: https://vitalik.ca/general/2019/09/22/plonk.html.

[9] *Proposal: The Turbo-PLONK Program Syntax for Specifying SNARK Programs*. Available: https://docs.zkproof.org/pages/standards/accepted-workshop3/proposal-turbo_plonk.pdf.

[10] *PLOOKUP: A Simplified Polynomial Protocol for Lookup Tables*. Available: https://eprint.iacr.org/2020/315.pdf.

[11] Y. Zhang, S. Wang, X. Zhang, et al., “Pipezk: Accelerating zero-knowledge proof with a pipelined architecture,” in 48th ACM/IEEE Annual Inter- national Symposium on Computer Architecture, ISCA 2021, Valencia, Spain, June 14-18, 2021, IEEE, 2021. Available: https://doi.org/10. 1109/ISCA52012.2021.00040.

[12] *Scroll*, Available: https://scroll.io/blog/zkEVM.

[13] Team, Polygon. “The Future Is Now for Ethereum Scaling Introducing Polygon Zkevm.” Polygon, 20 July 2022, https://blog.polygon.technology/the-future-is-now-for-ethereum-scaling-introducing-polygon-zkevm/.

[14] Hermez, Polygon. “Introducing Hermez ZKEVM.” Blog, Polygon Hermez | Blog, 19 Aug. 2021, https://blog.hermez.io/introducing-hermez-zkevm/.

[15] Farmer, Brendan. “Introducing Plonky2 - Polygon: Blog.” Polygon, 18 May 2022, https://blog.polygon.technology/introducing-plonky2/.

[16] Bjelic, Mihailo. “Polygon Announces Polygon Miden - a Stark-Based, Ethereum-Compatible Rollup - Polygon: Blog.” Polygon, 13 Apr. 2022, https://blog.polygon.technology/polygon-announces-polygon-miden-a-stark-based-ethereum-compatible-rollup/.

[17] “Dune Analytics.” *Dashboards*. Available: https://dune.com/Brecht/loopring.

[18] “Loopring (LRC): Dashboard.” *Token Terminal*, Available: https://tokenterminal.com/
terminal/projects/loopring.

[19] Devos, Brecht. “ZKSNARK Prover Optimizations.” *Medium*, Loopring Protocol, 2 Mar. 2020. Available: https://medium.loopring.io/zksnark-prover-optimizations-3e9a3e5578c0.

[20] M. Vezenov. Accelerating zkSNARKs on Modern Architectures. Masters Thesis, Lehigh University, 2022.