The Proof Supply Chain

by: Trace

Figment Capital

Follow

--

Listen

Share

Zero-knowledge (ZK) cryptography is rapidly improving, academically and commercially. As new ZK applications launch and scale, we'll need new infrastructure to serve them.

However, the mechanism design necessary for robust ZK infrastructure remains underexplored. In this piece, we provide an early look into one important component of ZK infrastructure: the proof supply chain. This is the pipeline from an application's intent to generate a ZK proof to that proof's on-chain submission. We'll show how the proof supply chain is a continuation of Ethereum's trend towards greater fee segmentation. We'll also show how it shares a similar market structure as the transaction supply chain, and as a result, faces many of the same questions and challenges.

Ethereum Fee Markets and MEV Today

Ethereum launched as a low-resource, general-purpose blockchain. The network is computationally flexible via a blockchain-native VM, but it isn't performant and has low throughput and high latency.

Seeking scale, its research community investigated solutions likesharding and plasma, before ultimately settling on a rollup-centric scaling roadmap in 2020. Rollups move execution off-chain, allowing the base layer to focus on DA and settlement.

Multi-Dimensional Fee Markets

To make rollups more affordable, the community proposed a new fee market for data blobs — the data that rollups submit to Ethereum — as part of proto-danksharding (EIP-4844), which is expected to go live in 2024.

The proposal for multidimensional fee markets was one of the earliest examples of block segmentation — the dividing of a block into different components. Block segmentation results in fee segmentation, allowing different types of transactions to have different cost structures.

Before rollups, Ethereum blocks mostly consisted of L1 transactions. Once rollups launched, they needed to occasionally submit their L2 transaction data to Ethereum as calldata. That calldata needs to compete with L1 transactions for the same limited gas per block. Multi-dimensional fee markets change this. After EIP-4844, rollups will be able to submit data blobs through a separate channel from transactions. These data blobs have fees independent from L1 transaction congestion.

In the image above, we show normal transactions above the blobs in the block. However, there's no concept of ordering with respect to transactions and blobs; the fee markets are orthogonal.

The Rise of MEV

Rollups are not the only driver of fee segmentation; we also have segmentation from MEV. In 2019–2020, on-chain activity grew, driven by DeFi. Miners realized that they could extract value from this activity by ordering and including transactions into blocks in certain ways.

In proof-of-work Ethereum, searchers competed for their transactions to be included at the top of the block (TOB) via priority gas auctions (PGAs) to capture the most valuable MEV opportunities like arbitrage. Although this didn't involve technical block segmentation like multi-dimensional fee markets for rollups, it did implicitly introduce a form of fee segmentation: the gas costs needed to be included at the top of the block were different from the rest of the block (ROB). Flashbots later introduced a mechanism for TOB inclusion separate from PGAs in the public mempool, making this segmentation more explicit.

After the merge, motivated by the need to mitigate the centralizing effects of MEV, Flashbots introducedproposer-builder separation through MEV-boost. That push expanded the transaction supply chain.

The transaction supply chain intended to shift the centralizing effects of MEV from the proposer-level to the new builder-level. However, builder centralization is also problematic, particularly for censorship-resistance. Since builders fully construct Ethereum blocks, they have control over what does and does not get submitted on-chain. This concern has motivated additional research into new techniques like censorship-resistance (CR) lists and MEV-boost+ which attempt to return some inclusion power back to the proposer. In the case of MEV-boost+, this is done by allowing the proposer to build the ROB themselves, creating the technical block segmentation between TOB and ROB that is currently implicit.

Recently, builders have also begun to explore bottom-of-block (BOB) MEV opportunities. BOB blockspace has similarities to the next block's TOB since it can react to the transactions executed in the ROB. Overall, the transaction component of the block continues to segment.

Centralization and Vertical Integration

Since its introduction, the transaction supply chain has become more centralized and vertically integrated. Centralization is primarily driven by the orderflow flywheel. To a first approximation, the winning builders are those with the most orderflow.

Integrated-builders with exclusive or self-generated orderflow like BeaverBuild and R-Sync have massivemarket share. Builders are now building relays, which may soon become vertically-integrated. MEV is also driving more sophistication and centralization among proposers, with P2P recently announcing that they will delay their block proposals toaccumulate more MEV rewards.

Centralization and vertical integration will continue as long as the current market structure remains intact.

Summary

In contrast to Solana's state localized fee markets, Ethereum is segmenting fees by transaction type. Once proto-danksharding goes live, Ethereum will have a separate fee market for rollups' data blobs. Ethereum blocks are further segmenting due to MEV.

Additionally, the transaction supply chain is actively centralizing and vertically integrating.

So how do ZK proofs fit into this picture?

The Emerging Proof Supply Chain

ZK rollups were one of the earliest use cases for ZK. These rollups submit their state root and a proof of their state transitions to Ethereum for settlement for which they must pay gas costs.

Although ZK rollups have been discussed for years, they have only recently launched. Scroll, ZKSync, and Polygon zkEVM all went live in 2023. ZK rollups have been followed by additional ZK applications including coprocessors, zkBridges, zkOracles, zkML, and zkDID. Many of these applications will also be on mainnet in the next couple of years.

Each one of these applications generates ZK proofs that must be submitted on-chain. That means they must compete with transactions and other proofs for limited blockspace.

Proof Aggregation

Submitting proofs on-chain is expensive. Fortunately, there's a solution: proof aggregation.Proof aggregation is a technique to combine multiple proofs together into a single proof. Just like ZK can be used to compress many transactions into a single proof, it can also be used to compress many proofs into one. This is done by recursively proving the verification of multiple proofs, often in a tree-like structure.

The final aggregated proof can then be submitted on-chain, where it is verified by the network, implicitly verifying all the input proofs. Proof aggregation allows the on-chain submission and verification gas costs to be amortized across all of the proofs; the cost of verifying an aggregated proof is roughly the same as verifying a single regular proof.

Proof aggregators face 2 questions:

1. Inclusion

— which proofs should be included in the aggregated proof?

1. Ordering

— what order should the included proofs be in?

Applications want their proofs submitted on-chain quickly, but proof aggregation is computationally intensive. An aggregator cannot combine an unlimited number of proofs within a single block time. Therefore, it needs to decide which subset of proofs should be included and which should not for every block.

Ordering and proof height may also matter. For example, applications whose proofs are closer to the top of the aggregation tree have shorter merkle paths, providing them with cheaper merkle inclusion proofs.

Decision making over proof ordering and inclusion makes proof aggregators similar to transaction sequencers. And just like sequencers and builders have opportunities to extract MEV from transactions, aggregators may be able to extract value from their ability to order and include proofs.

Moreover, aggregators benefit from a similar flywheel effect as builders. The more proof flow sent to the aggregator, the more on-chain gas costs can be amortized, resulting in more proof flow.

Proof aggregators drive further fee segmentation by introducing a new fee market: the cost to be included in a proof aggregation. This proof gas market is still weakly impacted by L1 transaction gas prices (since the aggregated proof must still compete for that blockspace), but is largely independent.

Proof Generation

The other major component of the proof supply chain is proof generation. ZK applications have a problem: maintaining a decentralized prover set is expensive and complex. It requires running specialized hardware and involves complex mechanism design. Applications already have plenty of engineering and business development challenges. Most teams will want to outsource proof generation to a third-party instead of handling it themselves. Proof markets — networks that provide proofs-as-a-service — are that third-party.

Proof markets are a continuation of the modular blockchain thesis, which argues that each task should be performed by a different piece of specialized infrastructure. Infrastructure specialization allows teams to outsource complexity and inherit shared security, while the specialized layer can benefit from economies of scale.

At a high level, a proof market has just 3 components:

1. Request pool

— a mempool for requests for proofs

1. Prover set

— a set of provers

1. Matching algorithm

— an algorithm for matching proof requests to a prover

Applications send proof requests to the request pool. The proof market then uses a matching algorithm, such as an orderbook or auction, to match that request to a prover. That prover then generates the proof and sends it to some destination, likely an L1 or L2, which may be specified in the request's metadata.

Proof markets have flywheel effects too. More request flow drives more competition among provers, resulting in lower costs for proof generation. It also results in higher hardware utilization, creating economies of scale; the prover set's fixed costs for running the infrastructure can be amortized with more volume, lowering marginal costs.

The Proof Supply Chain

Together, proof aggregators and proof markets construct the proof supply chain.

The entire proof supply chain is illustrated above. Walking through it step-by-step:

1. Request submission —

applications submit proof requests to the request pool

1. Request matching

— requests are matched to provers

1. Proof generation

— provers generate proofs for their matched requests

1. Proof submission

— proofs are submitted to the proof mempool

1. Proof selection

— a subset of proofs in the proof mempool are selected and ordered to be included in the aggregated proof

1. Proof aggregation

— the aggregator generates an aggregated proof from the selected proofs

1. Aggregation submission

— the aggregated proof is submitted to its destination

Steps 1–4 make up the proof market, while steps 5–7 are performed by the proof aggregator.

Comparison to Transaction Supply Chain

Viewing the proof supply chain in its entirety reveals its similarities to the transaction supply chain. As with normal transactions, the proof supply chain begins with applications or users who submit intents, or in this case requests, to a platform.

Proof markets have a similar structure to orderflow auctions. In orderflow auctions, searchers bid for exclusive rights to a transaction or intent. Searchers are highly fungible, with most value in a competitive market flowing back to the intent originator. In proof markets, provers fight for the right to satisfy proof requests in a similarly competitive market.

As mentioned previously, aggregators have similar flywheel effects to builders. We expect the market to centralize around a small number of request pools and proof aggregators. Meanwhile, prover sets and destination chains will remain relatively decentralized and competitive among operators.

Vertical Integration

The transaction supply chain is vertically integrating. We expect the proof supply chain to be vertically integrated as well.

Proof markets and proof aggregators each have flywheel effects. If instead of splitting these tasks into 2 separate roles, a single entity performed both, they would benefit from both flywheel effects.

Crucially, a vertically integrated proof supply chain allows the proof market to direct its proof flow exclusively toward its own proof aggregator. That proof aggregator then has higher volume, providing cheaper on-chain verification costs. These cheaper costs then incentivize more request flow to the proof market.

Third-party provers, like client-side applications or lower-volume proof markets, are incentivized to submit their proofs to the dominant aggregator (assuming they service third-party proofs). Of course, third-party proofs could be submitted directly to the destination chain or a smaller aggregator; it would just be more expensive. This external proof flow may drive further centralization. Since the aggregator has leverage over proof inclusion, this market structure introduces censorship-resistance concerns, similar to blockbuilding.

These developments will take years to unfold. Today, there are only a handful of projects building components of the proof supply chain. These include Nebra (proof aggregator), Gevulot (proof market), Marlin (proof market), Pluto (vertically-integrated), Bonsai by RiscZero (vertically-integrated), Succinct (vertically-integrated), and =nil; (vertically-integrated).

Proof Bundles

Aggregated proofs contain many individual proofs. Nebra, for example, will initially be able to aggregate 32 proofs per batch. This compression makes aggregated proofs consequential transactions. Depending on what ZK applications arise and what their proofs involve, there may be incentives to front-run or back-run aggregated proofs. A close analogy is oracle updates, which can be profitably backrun.

As ZK applications mature, we expect proof bundles to become an important part of each Ethereum block.

Conclusion

The commercialization of ZK applications will scale proof generation, spawning a new supply chain for ZK proofs. This new supply chain has similar market structure, centralization vectors, and censorship concerns as the transaction supply chain.

The proof supply chain will continue Ethereum's trend towards more granular fee segmentation; proofs will have a fee market semi-independent from normal transactions.

This supply chain will become an important part of Ethereum in the years ahead. We look forward to working with the rest of the community to investigate the opportunities and challenges it creates.

Open Research Questions

We’re interested in research into the following questions, among others:

1. How can we build censorship-resistance into proof supply chains, given their centralization vectors?
2. What forms of aggregator extractable value exist, and how should we design the proof supply chain around them?
3. How will the emerging proof supply chain impact the existing transaction supply chain?

If you’re a founder or researcher interested in this space,we’d love to chat.

Acknowledgements: Special thanks to

[Velvet

](https://twitter.com/VelvetMilkman),

[Dougie

](https://twitter.com/DougieDeLuca),

[Mike

](https://twitter.com/mikeneuder),

[Kubi

](https://twitter.com/kubimensah),

[Dmarz

](https://twitter.com/_danielmarzec),

[Teemu

](https://twitter.com/teemupai),

[Tracy

](https://twitter.com/devloper_xyz),

[Uma

](https://twitter.com/pumatheuma),

[Sidd

](https://twitter.com/MarlinProtocol), and

[Misha

](https://twitter.com/nemothenoone)for their feedback on an earlier version of this piece.