Decentralized Proving, Proof Markets, and ZK Infrastructure

By: Trace

Figment Capital

Follow

--

Listen

Share

Acknowledgements: special thanks to

[Velvet

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

[Dougie DeLuca

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

[Ole Spjeldnæs

](https://twitter.com/spjoleh)(Repyh/Delendum),

[Weikeng Chen

](https://twitter.com/weikengchen)(DZK),

[Siddhartha Dutta

](https://twitter.com/nihilistsid?lang=en)and

[Roshan Raghupathy

](https://in.linkedin.com/in/roshanrags)(Marlin),

[Uma Roy

](https://twitter.com/pumatheuma?lang=en)(Succinct),

[Mikhail Komarov

](https://twitter.com/nemothenoone?lang=en)(=nil; Foundation),

[Kacper Koziol

](https://twitter.com/kacperkozi)(Herodotus),

[Cem Özer

](https://twitter.com/cemozer_?lang=en)(Sovereign Labs),

[Tracy Livengood

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

[Tiancheng Xie

](https://niconiconi.github.io/en_index.html)(Polyhedra Network),

[Chris Goes

](https://twitter.com/cwgoes)(Anoma), and

[Toghrul Maharramov

](https://twitter.com/toghrulmaharram?lang=en)(Scroll) for their insights and feedback which contributed to this piece.

Summary

- Improvements to ZK cryptography, tooling, and hardware are enabling new blockchain applications that require computationally-intensive proof generation.
- These applications will seek to decentralize their proving infrastructure to improve their liveness and censorship-resistance.
- We explore the leading approaches to decentralizing proof networks and conclude that most applications will pursue a stake-based mechanism.
- Third-party proof networks will be built that allow applications to outsource their proving, thereby lowering the overhead for building and running ZK infrastructure. We discuss the main design considerations for proof markets.
- Finally, we discuss the requirements to operate ZK infrastructure and conclude that staking providers and new ZK-native teams are best positioned to serve the emerging proving market.

Introduction

Zero-knowledge (ZK) technology is rapidly improving. As it advances, more ZK applications will emerge, driving higher demand for zero-knowledge proof (ZKP) generation.

Most ZK applications today are privacy protocols. Proofs for privacy applications likeZCash and TornadoCash are generated locally by the user, since knowledge of the secret input is required to generate the ZKP. Such computations are fairly small, allowing them to be generated on consumer hardware. We refer to user-generated ZK as client-side proving

.

While some proof generation can be relatively lightweight, others require more intensive computation. Validity rollups (i.e., zkRollups), for example, may require proving thousands of transactions in a ZK virtual machine (zkVM), requiring more computation and, as a result, longer proving times. Proving these large computations requires powerful machines. Fortunately, since these proofs rely only on succinctness and not zero-knowledge (there are no secret inputs), proof generation can be safely outsourced to an external party. We refer to outsourced proof generation as server-side proving

.

Server-Side Proving

Server-side proving is found in a number of blockchain applications. These include:

1. Scaling

: Validity rollups like Starknet, zkSync, and Scroll scale Ethereum by moving computation off-chain.

1. Cross-chain interoperability

: Proofs can be leveraged to facilitate trust-minimized communication between different blockchains, enabling secure data and asset transfer. Teams include Polymer, Polyhedra, Herodotus, and Succinct.

1. Trustless Middleware

: Middleware projects like RiscZero and HyperOracle provide access to trustless off-chain computation and data using ZKPs.

1. Succinct L1s

: Succinct Blockchains like Mina and Repyh use recursive SNARKs, allowing even computationally weak users to independently verify the state.

Now that much of the prerequisite cryptography, tooling, and hardware has been developed, applications utilizing server-side proving are finally coming to market. Server-side proving will grow exponentially in the coming years, requiring the development of new infrastructure and operators who can effectively generate these computationally-intensive proofs.

Though centralized to start, most applications leveraging server-side proving have long-term aspirations of decentralizing the prover role. As with other pieces of the infrastructure stack, like validators and sequencers, effectively decentralizing the prover role will require careful protocol and incentive design.

In this piece, we explore designs for prover networks. We begin by distinguishingproof networks

from proof markets

. Proof networks are prover sets that service a single application, like a validity rollup. Proof markets are open markets where multiple applications can submit requests for verifiable computations. Next, we provide an overview of current decentralized proof network models before sharing some early scoping on proof market design, an area that remains underexplored. Finally, we discuss the challenges of operating ZK infrastructure, concluding that staking providers and specialized ZK teams are best positioned to cater to the emerging proving market, rather than PoW miners.

Proof Networks and Proof Markets

ZK applications require provers to generate their proofs. Though centralized today, most ZK applications will decentralize their proof generation. Provers do not need to be trusted to produce a correct output since proofs can be easily verified. Nevertheless, applications will pursue decentralized proving for a few reasons:

1. Liveness

: Multiple provers ensure that the protocol operates reliably and doesn't face downtime if some provers are temporarily unavailable.

1. Censorship Resistance

: Having more provers improves censorship resistance. A small prover set could refuse to prove certain types of transactions.

1. Competition

: A larger prover set can strengthen market pressures for operators to create faster and cheaper proofs.

This leaves applications with a design decision: should they bootstrap their own proof network or should they outsource the responsibility to a proof market?

Outsourcing proof generation to a proof market like those under development by=nil;, RiscZero, and Marlin provides out-of-the-box decentralized proving and allows the application's developers to focus on other components of their stack. Indeed, these markets are a natural extension of the modularity thesis. Similar to shared sequencers, proof markets are de facto shared prover networks.[1] By sharing provers across applications, they also maximize hardware utilization; provers can be repurposed whenever one application does not immediately need a proof to be generated.

Proof markets also have downsides. Internalizing the prover roleimproves native token utility by allowing a protocol to leverage its own token for staking and prover incentivization. It also provides applications with greater sovereignty, rather than creating an external point of failure.

An important difference between proof networks and proof markets is that in proof networks, there is often only a single proof request which must be satisfied by the prover set at a time. For example, in validity rollups, the network takes a number of transactions, computes a validity proof to demonstrate they were executed correctly, and sends the proof down to the L1. The single validity proof is generated by a prover chosen from a decentralized set.[2]

Decentralizing Proof Networks

As ZK protocols harden, many teams will progressively decentralize their infrastructure to improve network liveness and censorship-resistance. Introducing multiple provers to a protocol creates additional complexity for the network. In particular, the protocol must now decide which prover is assigned to a given computation. There are 3 main approaches so far:

1. Stake-based prover selection

— Provers stake assets to participate in the network. At each proving slot, a prover is selected at random, weighed by their value of staked tokens, and computes the output. Provers are compensated for producing a proof when chosen. Specific slashing conditions and leader selection can be different for each protocol. This model is similar to PoS.

1. Proof mining

— Provers are tasked with repeatedly generating ZKPs until they generate a proof with a sufficiently rare hash. Doing so earns them the right to prove at the next slot and earn the slot reward. Provers that can generate more ZKPs are more likely to win the slot. This type of proving closely mirrors PoW mining — it is energy and hardware intensive. A key difference with traditional mining is that in PoW, hashing is merely a means to an end. Being able to produce SHA-256 hashes in Bitcoin has no value beyond increasing the network's security. In proof mining however, the network provides incentives to miners to accelerate ZKP generation, which ultimately benefits the network. Proof mining has been pioneered byAleo.

1. Proof racing

— At each slot, provers compete to produce a proof as quickly as possible. Whoever generates the proof first receives the slot reward. This approach is vulnerable to winner-takes-all dynamics. If a single operator is able to generate proofs faster than anyone else, they should win every slot. Centralization can be reduced by splitting up the proof reward across the first n

operators who generate a valid proof or introducing some randomness into which proof is accepted. Yet even in this case, the fastest operator can simply run multiple machines to capture the other revenue.

Another technique is distributed proving. Here, instead of a single party winning the right to prove at a given slot, proof generation is distributed between multiple parties who work together to generate a single output. One example is a federated proving network which splits a proof into many smaller statements that can be proved separately, then recursively proved up to a single statement in a tree-like structure. Another is zkBridge, which proposed a new ZKP protocol called deVirgo that enables proofs to be easily distributed across machines, and has already been deployed by Polyhedra. Distributed proving is inherently easier to decentralize and can significantly improve the proof generation speed. Each group of participants can form a computing cluster and join proof mining or racing. Rewards can be divided evenly according to their contribution to the cluster. Distributed proving is compatible with any prover selection model.

Stake-based prover selection, proof mining, and proof racing tradeoff on 3 axes: capital requirements, hardware accumulation requirements, and prover optimization.

Stake-based prover models

require provers to lock-up capital but place less importance on accelerating proof generation since provers are not selected based on their proving speeds (although faster provers may have more success in attracting delegations). Proof mining

is more balanced. It requires some capital to accumulate machines and cover energy costs to produce more proofs. It also places incentives on ZKP acceleration, just as Bitcoin mining creates incentives for accelerating SHA-256 hashing. Proof races

require the least capital and infrastructure. Operators can run a single hyper-optimized machine to compete at each slot. Despite being the most lightweight, we believe proof races risk the highest centralization due to their winner-takes-all dynamics. Proof races (like mining) also result in redundant computation, but provide better liveness assurances since you do not need to worry about a prover missing the slot they were selected for.

Another benefit of the stake-based model is that placing less pressure on provers to compete on performance creates space for collaboration between operators. Collaboration typically includes knowledge sharing, like the dissemination of new techniques to accelerate proof generation, or guidance for new operators on how to start proving. By contrast, proof racing is more analogous to MEV searching, where entities are more secretive and adversarial to maintain their competitive edge.

Among these 3 factors, we believe that the demand for speed will be the primary variable impacting the network's ability to decentralize its prover set. Capital and access to hardware will be plentiful. However, the more that provers must compete on speed, the less decentralized the network's proving becomes. On the other hand, the more that speed is incentivized, the more performant the network will be, all else equal. Although the implications differ, proof networks face the same performance vs. decentralization tradeoff as layer 1 blockchains.

Which Prover Selection Model Will Win?

We anticipate that the majority of proof networks will use stake-based models. Such models provide the best balance between incentivizing performance and maintaining decentralization.

Distributed proving probably won't be used for most validity rollups. Models where provers each prove a small number of transactions and then recursively aggregate them together face network bandwidth limitations. The sequential nature of rollup transactions also makes it difficult to parallelize — proofs for previous transactions must be included before proofs for subsequent transactions. If a prover does not make its proof available, the final proof cannot be constructed.

Outside of Aleo and Ironfish, ZK mining will be an unpopular choice among ZK applications. It is energy-intensive and unnecessary for most applications. Proof races will also be unpopular because of their centralization effects. The more a protocol prioritizes performance relative to decentralization, the more appealing race-based models become. However, accessible ZK hardware and software acceleration already provides a large speedup. We expect that the marginal improvement to proof generation times from adopting proof races will not be worth the marginal costs to decentralization for most applications.

Designing Proof Markets

As more applications turn to ZK, many are realizing that they would rather outsource the ZK infrastructure to proof markets than handle it in-house. Unlike proof networks, proof markets service multiple applications, each with different proving needs. These markets will aim to be performant, decentralized, and flexible.

1. Performant —

The market will have heterogeneity in proving demand. For example, some proofs require more computation than others. Proofs with long generation times will need ZKP acceleration through specialized hardware and other optimizations. The market also needs to service fast proof generation for applications and users willing to pay for it.

1. Decentralized —

Similar to proof networks, proof markets and their applications will want the market to be decentralized. Decentralized proving increases liveness, censorship-resistance, and market efficiency.

1. Flexible —

All else equal, proof markets want to be as flexible as possible to satisfy different applications' needs. A zkBridge connected to Ethereum may need a final proof like Groth16 that provides cheap on-chain proof verification. By contrast, a zkML model may prefer a Nova-based proving scheme that optimizes for recursive proving. Flexibility can also come in terms of the integration process. The market can provide a zkVM for verifiable computation of programs written in higher level languages like Rust, providing a simpler integration for developers.

Designing proof markets that are performant, decentralized, and flexible enough to power an array of ZK applications is a difficult, underexplored area of research. Solving it requires careful incentive and technical design. Below, we share some early scoping on proof market design considerations and trade-offs including:

- Incentives and Disincentives
- Matchmaking
- Custom circuits vs zkVMs
- Continuous vs aggregated proving
- Hardware heterogeneity
- Operator diversity
- Discount, derivatives, and order types
- Privacy
- Progressive and persistent decentralization

Incentives and Disincentives

Provers must have incentives and disincentives to maintain market integrity and performance. The easiest way to introduce incentives is with staking and slashing dynamics. Operators can be incentivized with proof request bids, and potentially with token inflation rewards.

Staking minimums could be imposed for joining the network to prevent sybil attacks. Provers that submit false proofs could be slashed. Provers that take too long to generate proofs, or fail to produce them at all, could also be punished. Such punishment could be proportional to the proof bid — the higher the bid that gets delayed (and therefore the more economically significant), the larger the punishment.

In cases where slashing is overkill, a reputation system could be used instead.=nil; currently uses a reputation-based system to hold provers accountable. Provers with a history of dishonesty or poor performance are less likely to be matched with bids by the matching engine.

Matchmaking

Matchmaking is the problem of connecting supply and demand in the market. Designing the matchmaking engine — that is, defining the rules by which provers are matched with proof requests — will be one of the most difficult and consequential tasks of the marketplace. Matchmaking can be done through auctions or orderbooks.

- Auctions

: Auctions involve provers bidding on proof requests to determine which prover wins the right to generate the proof. A challenge with auctions is that if the winning bid fails to return the proof, the auction must be re-run (you can't immediately enlist the second highest bidder to run the proof).

- Orderbooks

: Orderbooks require applications to submit bids for buying proofs to an open database; provers must submit asks

for selling proofs. Bids and asks could be matched if they meet 2 requirements: 1) the protocol's bid computation price is higher than the prover's ask price, and 2) the prover's turnaround time is lower than the bid's requested time. In other words, applications submit a computation to the orderbook and define a maximum reward they're willing to pay and a maximum time they're willing to wait to receive the proof. If a prover submits an ask at a price and time below that requirement, they're eligible to be matched. Order books are better suited for low-latency use-cases, since orderbook bids can be filled immediately.

Proof markets are multi-dimensional; applications must request a computation within a certain price and timescale. Applications may have dynamic preferences over proof latency, where the price they're willing to pay for proof generation decreases over time. Though efficient, order books have shortcomings in reflecting the complexity of users' preferences.

Additional matchmaking models can be drawn from other decentralized marketplaces like Filecoin's decentralized storage market which uses off-chain negotiation and Akash's decentralized cloud computing market which uses a reverse auction. In Akash's marketplace, developers (called "tenants") submit computing jobs to the network, and cloud providers bid on the workload. The tenant can then choose which bid to accept. Reverse auctions are well-suited for Akash, since workload latency isn't as important, and the tenant can manually select which bid they want. By contrast, proof markets need to operate quickly and autonomously, making reverse auctions a suboptimal matchmaking system for proof generation.

The protocol could place restrictions on the types of bids certain provers can accept. For example, a prover with an insufficient reputation score could be barred from matching with large bids.

The protocol must protect against attack vectors that arise from permissionless proving. In some scenarios, provers can commit proof delay attacks: by delaying or failing to return a proof, provers can expose a protocol or its users to certain economic attacks. Token slashing or reputation score penalties may be insufficient to deter malicious provers if the attack profits are large. In the event of a proof delay, rotating proof generation rights to a new prover can minimize downtime.

Custom Circuits vs zkVM

Proof markets can provide custom circuits for each application, or they can provide a single general-purpose zkVM. Custom circuits have higher integration and financial overhead but result in better performance for the application. The custom circuits can be built by the proof market, the application, or third-party developers who earn a share of network revenue in exchange for their services, as is the case of =nil;.

Though slower, zkVMs like RiscZero's STARK-based RISC-V zkVM allow application developers to write verifiable programs in higher-level languages Rust or C++. The zkVM can support accelerators for common zk-unfriendly operations like hashing and elliptic curve addition to improve performance. While proof markets with custom circuits may require separate orderbooks, leading to prover fragmentation and specialization, zkVMs can use a single orderbook to facilitate and prioritize computation on the zkVM.

Individual vs. Aggregated Proving

Once proofs are generated, they must be relayed back to applications. For on-chain applications, this requires costly on-chain verification. Proof markets can relay individual proofs back to developers, or they can use proof aggregation to convert many proofs into one before returning them, amortizing gas costs across them.

Proof aggregation introduces additional latency. The proofs need to be aggregated together, requiring more computation, and multiple proofs must be completed before they can be aggregated, potentially delaying the aggregation process.

Proving markets must decide how they handle the latency versus cost tradeoff. Proofs can be returned quickly ad hoc at a higher cost or aggregated at a lower cost. We expect proving markets will require proof aggregation, but can shorten their aggregation cadence with scale.

Hardware Heterogeneity

Large computations are slow to prove. What happens when an application wants a computationally-intensive proof to be generated quickly? Provers could use more powerful hardware like FPGAs and ASICs to accelerate proof generation. Though great for performance, specialized hardware can restrict the set of possible operators, hindering decentralization. Proof markets need to decide what hardware its operators run.

There's also the question of prover homogeneity: proof markets must decide whether all provers will have the same hardware or whether it will support different setups. Markets may be easier to maintain decentralization if all provers play on a level playing field with accessible hardware. Given the nascency of ZK hardware and the need for market performance, we anticipate proof markets will remain hardware agnostic, allowing operators to run whatever infrastructure they want. However, more work needs to be done on the implications of prover hardware diversity on prover centralization.

Operator Diversity

Developers must define the requirements for operators to enter and remain active market participants. These requirements will influence operator diversity, including their size and geographical distribution. Some protocol-level considerations include:

- Will provers need to be whitelisted or will admission be permissionless?
- Will there be a cap on the number of provers who can participate?
- Will provers be required to stake tokens to enter the network?
- Will there be any hardware or performance minimums?
- Will there be limits to how much market share an operator can hold? If so, how is that limit enforced?

Markets exclusively looking for institutional-grade operators may have different market entrance requirements than those seeking retail participation. Proof markets should define their vision for what a healthy operator set looks like, and work backward from there.

Discounts, Derivatives, and Order Types

Proof market prices may be subject to price swings during times of higher or lower demand. Price volatility causes uncertainty. Applications need to predict future proving market prices in order to pass those fees on to the end user — a protocol doesn't want to charge their user $.01 for a transaction, only to later find that it costs $.10 to prove the transaction. This is the same problem faced by layer 2s that must pass future calldata prices onto users. Some have proposed that L2s use blockspace futures to solve this problem: an L2 can buy blockspace up front for a fixed price while also providing more stable prices to users.

The same demand is present in proving markets. Protocols like validity rollups may generate proofs at a recurring cadence. If a rollup needs to generate a proof every hour for a year straight, could it submit that bid once instead of needing to submit a new bid ad hoc, and potentially becoming vulnerable to price surges? Ideally, they could pre-order proving capacity. If available, should proving futures be provided within the protocol, or should the protocol allow another protocol or centralized provider to create the service on top of it?

And what about discounts for bulk or predictable orders? If a protocol brings a lot of demand to the market, should it get a discount, or must it pay the open market price?

Privacy

Proof markets could provide private computation, though outsourced proof generation is difficult to perform privately. A secure channel is needed to send private inputs from the application to an untrusted prover. Once received, the prover needs a secure computational sandbox to generate proofs without leaking private inputs; secure enclaves are one promising direction. Indeed, Marlin is already experimenting with private computation via secure enclaves on Azure with Nvidia's A100 GPUs.

Progressive and Persistent Decentralization

Proof markets will need to figure out the best way to progressively decentralize. How should the first third-party provers enter the market? What are the tangible steps toward decentralizing?

Related is the problem of maintaining decentralization. One challenge with proof markets is prover undercutting. A well-funded prover could choose to operate at a loss by bidding below market rates to price other operators out of the market before growing and increasing prices. Another form of undercutting is operating an excessive number of nodes while bidding market rates, allowing random selection to match the operator with a disproportionate amount of proof requests.

Conclusion

Beyond the considerations above, additional decisions include how bids are submitted and whether proof generation can be distributed across multiple provers. In conclusion, proving markets have a massive design space that must be carefully explored to build a performant and decentralized market. We look forward to collaborating with the leading teams in this area to identify the most promising approaches.

Operating ZK Infrastructure

So far, we've looked at design considerations for building decentralized proof networks and proof markets. In this section, we evaluate which operators will be best suited to participate in proof networks, and share some thoughts on the supply side of ZKP generation.

Miners and Validators

There are 2 main types of blockchain infrastructure providers today: miners and validators.[3] Miners run nodes on PoW networks like Bitcoin. These miners compete to produce sufficiently rare hashes. The more powerful their computers and the more computers they have, the more likely they are to find the rare hash and earn the block reward. Early Bitcoin miners started with CPUs on household computers but professionalized as the network grew and as the block reward became more valuable. Nodes were pooled together to achieve economies of scale and hardware setups specialized over time. Today, miners run almost exclusively with Bitcoin ASICs in data centers built near sources of cheap energy.

The rise of proof-of-stake required a new type of node operator: validators. Validators perform a similar role to miners. They propose blocks, execute state transitions, and participate in consensus. However, instead of producing as many hashes as possible to increase their odds of proposing a block, validators are randomly selected to propose a block based on the value of assets staked to them. This change eliminated the need for energy-intensive setups and specialized hardware in PoS, allowing for a more widely distributed set of node operators. Validators can even be run in the cloud.

A more subtle change that PoS introduced was it made the blockchain infrastructure business a service business. In PoW, miners operate on the backend and are largely invisible to users and investors (how many Bitcoin miners can you name?). They have a single customer, which is the network itself. In PoS, validators “sell” their staked security to the network, but they also have another customer: the staker. Tokenholders sought out operators they could trust to safely and reliably run infrastructure on their behalf to earn staking rewards. Since validator revenues grow relative to the assets they can attract, they operate like a service company. Validators branded themselves, hired sales teams, and built relationships with individuals and institutions who could stake their tokens to the validator. This makes the staking business very different from mining. The important differences between the 2 businesses is one reason why the largest PoW and PoS infrastructure providers are entirely different companies.

ZK Infrastructure Companies

A number of companies have emerged in the past year that specialize in ZKP hardware acceleration. Some of these companies produce hardware to sell to operators; other organizations run the hardware themselves, making them a new type of infrastructure provider. The most well-known ZK hardware companies today include Cysic, Ulvetanna, and Ingonyama. Cysic plans to build ASICs that can accelerate common ZKP operations while keeping the chip flexible for future software innovations. Ulvetanna is building FPGA clusters to service applications that require especially intensive proving. Ingonyama is researching algorithmic improvements and building a CUDA library for ZK acceleration with plans to eventually design an ASIC.

Who Will Run ZK Infrastructure?

We believe that the types of companies which excel at operating ZK infrastructure will primarily depend on the prover incentive model and performance requirements. The market will be split between staking companies and new ZK-native teams. Applications that require the highest performance provers or extremely low-latency will be dominated by ZK-native teams capable of winning proof races. We expect such extreme requirements to be the exception rather than the norm. The rest of the market will be led by staking businesses.

Why aren’t miners well-positioned to run ZK infrastructure? After all, ZK proving, especially for proofs over large circuits, has many similarities with mining. It’s energy and compute intensive and may require specialized hardware. However, we do not believe miners will be early leaders in the proving space.

First, PoW hardware cannot be effectively repurposed for proving. Bitcoin ASICs by definition cannot be repurposed. The GPUs commonly used for mining Ethereum pre-merge like the Nvidia Cmp Hx were specifically designed for mining, making them ineffective at ZK workloads. Specifically, their weak data bandwidth eliminates any real gains from the parallelization provided by the GPU. Miners wanting to enter the proving business would have to accumulate ZK-friendly hardware from scratch.

Moreover, mining companies’ lack of brand recognition puts them at a disadvantage for stake-based proving. Miners’ biggest strength is their access to cheap energy, which could enable them to charge lower fees or participate in proving markets more profitably, but is unlikely to outweigh their challenges.

Lastly, miners are accustomed to static requirements. Bitcoin and Ethereum mining have not required frequent or significant changes to their hash functions, nor have these operators been tasked with other modifications to the protocol that impact their mining setup (excluding The Merge). By contrast, ZK proving requires staying vigilant to changes in proving technology that may affect hardware setups and optimizations.

The stake-based prover model is a natural fit for validator companies. Retail and institutional investors in ZK applications will delegate their tokens to infrastructure providers to generate rewards. Staking businesses have the existing teams, experience, and relationships to attract large delegations to their infrastructure. This applies even to protocols that do not support delegated PoS, as many validator companies offer whitelisted validator services to run infrastructure on another party’s behalf, a common practice on Ethereum.

Validators do not have the same access to affordable electricity that miners do, making them a poor fit for the most energy-intensive tasks. Running provers for a validity rollup would require a more intensive hardware setup than a normal validator, but can likely fit within a validator's current cloud or bare metal infrastructure. But like miners, these companies do not have in-house ZK expertise that could stay competitive in proof races. Outside of stake-based proving, running ZK infrastructure is a different business model from running validators and doesn't have strong flywheel effects with the staking business. We expect ZK-native infrastructure providers to dominate non-stake based and high performance proving tasks.

Conclusion

Most provers today are run by the teams building the applications that require them. As more ZK networks launch and decentralize, new operators will enter the market to meet the proving demand. Who those operators are is contingent on the prover selection model and on the prover requirements that specific protocols impose.

Staking infrastructure companies and ZK-native infrastructure operators are best positioned to dominate this new market.[4]

Decentralized proving is an exciting new category of blockchain infrastructure. If you're an application developer or infrastructure provider building in ZK, we'd love to hear from you.

Footnotes

1. Like shared sequencers, shared provers provide decentralized infrastructure as a service, but there are important differences. Shared sequencers help form a global transaction ordering, which is necessarily non-parallelizable. On the other hand, shared provers can be parallelized, as the applications' proofs do not require a global ordering.

2. In some cases, proof generation can be distributed across multiple operators. But even in this multi-prover setup, the basic market structure remains the same: there is only one proof generation task for which provers are selected at a time.

3. The 3rd major category are RPC providers like Alchemy and Infura

4. We originally believed that ZK-native infrastructure operators would control ZK infrastructure. After further research, we realized that the prevalence of staking and GPUs in proving infrastructure provided a stronger opportunity for validators than we initially anticipated.