# Seren - A Framework for Decentralized Web Infrastructure

**Abstract**

We propose a decentralized web infrastructure protocol that integrates decentralized storage, computation, and financial infrastructure into a cohesive framework for building truly serverless web applications. The protocol addresses fundamental limitations in existing blockchain networks through three interconnected components: a decentralized storage layer, a verifiable ZK-computation layer, and a Layer 1 blockchain. We propose modifications to established blockchain protocols that enable unified cross-component tokenomics and a gasless transaction mechanism.

In addition, we propose a novel walletless interaction model, allowing users to securely interact with decentralized applications (dApps) without downloading any software. This model achieves a level of private key protection comparable to modern browser extension wallets, leveraging secure in-browser environments.



Version: 0.1 (draft)

# Contents


**1 Introduction** **4**

**2 Toward Decentralized Web** **4**

**3 Existing solutions and current industry state** **5**
3.1 Ethereum . . . 5
3.2 Tron . . . 5
3.3 Solana . . . 5
3.4 Ton . . . 5
3.5 The Opportunity for a Unified Protocol . . . 5

**4 The protocol layout** **6**

**5 Enhanced User Experience** **6**

**6 Decentralized storage** **7**
6.1 General-purpose File Storage Layer . . . 7
6.2 Redefining the Arweave Transaction Format . . . 8
6.3 Storage Backend for Executable Programs . . . 8
6.3.1 Task Feed Mechanism and Proof-Driven Settlement . . . 8
6.4 Decentralized UI Delivery . . . 9
6.4.1 Walletless UI . . . 9
6.5 Decentralized Database and Data manifestation . . . 10
6.6 Messaging Medium . . . 11
6.7 Foundational Layer for Decentralized DNS . . . 11

**7 Decentralized Calculations** **11**
7.1 Computations . . . 12
7.1.1 Cryptographic Proof Generation . . . 12
7.2 Service Payments . . . 13
7.3 Delayed execution . . . 13
7.4 Recurrent calculations . . . 14

**8 Decentralized Financial layer** **14**
8.1 Network sharding . . . 14
8.1.1 dApps sharding economy . . . 14
8.2 Gasless transactions . . . 15
8.3 RISC-V virtual machine . . . 15
8.4 Smart contract language . . . 16
8.4.1 AssemblyScript . . . 16

**9 Areas of Application** **16**
9.1 Web3 payments in web2 websites . . . 16
9.2 Decentralized Gaming . . . 16
9.3 Decentralized Social Networks . . . 17
9.4 Pay-to-Stream Video Content and Micropayments . . . 17
9.5 Decentralized Messaging . . . 17

9.6 Decentralized Advertising Networks . . . . . 18
9.7 Digital Art and Intellectual Property Ownership . . . . . 18

**10 Final words** **18**

# 1 Introduction

The concept of a decentralized web has been a persistent vision in technological discourse, yet its precise definition remained elusive for years. Many mistakenly assumed that the internet's distributed network of servers already constituted decentralization. This confusion stemmed from conflating geographical distribution with actual decentralization of control and ownership.

A clearer definition emerged with the rise of serverless architecture - applications and services that operate without reliance on dedicated servers, whether physical or cloud-based. This model challenges the traditional client-server architecture that has dominated the internet for decades, as outlined by Nakamoto [1].

From the user's perspective, modern web applications are composed of four core components: HTML for structure, CSS for styling, JavaScript for interactivity, and backend storage for state and session data. This frontend-backend model has become the standard solution for digital interaction.

The promise of decentralization lies in reimagining these components in a serverless context. Applications can retain their familiar frontend while replacing centralized backends with decentralized, immutable, and permissionless infrastructure. This shift goes beyond technical innovation - it reshapes the principles of data control, moving it from platform providers to end users.

Technologies such as blockchain networks, decentralized file systems, and peer-to-peer protocols have laid the foundation for this vision, but the ecosystem remains incomplete. With continued development and refinement, these technologies are progressively enabling the deployment of applications that are autonomous, censorship-resistant, and resilient to single points of failure.

# 2 Toward Decentralized Web

The first real proof of decentralized applications came with Decentralized Finance (DeFi), demonstrating that financial services can operate without central intermediaries. However, DeFi also exposed key limitations. For example, DeFi primarily focuses on financial operations, while current decentralized infrastructure isn't suitable as a foundation for onboarding more complex applications and supporting their economics.

Later advances introduced Zero-Knowledge (ZK) proofs, enabling verifiable computation while preserving the security guarantees of decentralized systems. This helped bridge critical gaps by supporting more complex and variable operations.

As verifiable computation matured, user expectations also evolved—demanding not just security, but also censorship resistance and transparency in how web applications function. While DeFi didn't address these needs directly, the underlying technologies inherently support them, unlocking new possibilities for decentralized communication, media, and more.

The modern decentralized web is composed of three components: decentralized storage for data persistence, verifiable computation, and decentralized financial layer for value exchange. However, current implementations lack smooth interconnection, relying on overly complex technical solutions and fragmented tokenomics. As a result, these attempts remain experimental or niche products rather than realizing their full potential.

Our infrastructure unifies all three components into a single architecture - expanding the power of decentralization far beyond finance. The protocol represents a significant advancement toward the original vision of a decentralized web. It combines battle-tested blockchain technologies with novel approaches to computation, storage, and user experience - laying the groundwork for a scalable, censorship-resistant, and user-controlled digital future.

# 3 Existing solutions and current industry state

The current blockchain landscape features four major networks that have achieved significant user adoption and technical maturity: Ethereum, TRON, Solana, and TON. While other blockchain networks exist with various technical innovations, they largely resemble the mentioned blockchains, and their detailed consideration would questionably add unique advantages warranting inclusion for large-scale decentralized applications.

## 3.1 Ethereum

Ethereum established itself as the pioneer of smart contract platforms, introducing programmable blockchain functionality [2, 3]. Its robust security model and decentralized nature come at the cost of mid to high transaction fees and limited throughput. While Layer 2 solutions attempt to address scalability, they introduce additional complexity and potential security considerations.

## 3.2 Tron

TRON network introduced the innovative concept of system programs - built-in system programs with privileged access to transaction state, enabling deeper blockchain interaction than traditional smart contracts. It offers higher transaction throughput and low to mid fees through its proof-of-stake consensus. This consensus mechanisms have demonstrated its capability to dramatically improve transaction throughput compared to proof-of-work systems.

## 3.3 Solana

Solana prioritizes high performance through a novel proof-of-history consensus mechanism, enabling high throughput and low transaction costs [4]. Its innovative multi-instruction transaction schema, somewhat similar to Bitcoin's multi-input model, allows a single transaction to perform multiple operations.

## 3.4 Ton

TON introduces a multi-blockchain architecture with dynamic sharding [5]. Each application or contract can live on its own workchain or shardchain, which allows the network to scale horizontally as demand grows. Its asynchronous messaging between shards enables parallel execution, and built-in mechanisms for inter-shard communication make it highly suitable for complex, modular applications.

## 3.5 The Opportunity for a Unified Protocol

While established protocols like Ethereum, TRON, Solana, and TON have laid the foundation for the decentralized web, particularly in financial applications, they have only begun to explore the broader landscape of online interactions. Their innovations in decentralized finance (DeFi) proved that trustless systems can replace traditional intermediaries in specific areas. However, their real-world impact remains limited to a small fraction of the global digital economy.

Today, decentralized technologies have influenced only a small percentage of online value exchange, mostly within crypto-native ecosystems. In contrast, the vast majority of digital transactions, from subscriptions and content to cloud storage and advertising, remain rooted in centralized Web2 systems. This multi-trillion-dollar economy remains largely untouched by decentralization, in part because existing blockchain infrastructure lacks the usability, scalability, and flexibility needed to integrate with mainstream applications.

The Web2 economy represents a massive, largely untapped opportunity for decentralized technologies. According to estimates, the global digital economy is projected to reach $16.5 trillion by 2028 [6]. Despite the success of blockchain in DeFi, the lack of integrated decentralized storage and computation means these systems have only scratched the surface of what's possible.

This gap presents a major opportunity for

new projects to step in. By addressing Web2 concerns, such as offering cheaper decentralized storage, verifiable off-chain computations, and a seamless user experience, next-generation protocols can help Web2 platforms benefit from tamper-resistant infrastructure while unlocking substantial liquidity into the Web3 space. These advancements would not only close current gaps but also accelerate the decentralized web's adoption across the broader digital economy.

# 4 The protocol layout

The proposed decentralized web infrastructure rests composed of 3 fundamental components, each serving a distinct purpose while maintaining interoperability:

The decentralized data storage component refers to each uploaded file by its hash sum, similar to existing decentralized storage solutions. For human-readable names, an internal Resource Name Service (RNS) can optionally map an assigned name to the hash of the latest version of a file, automatically or manually updating pointers upon content modification. Files can be stored as mutable, with a dedicated metadata file defining the internal file organization, content update policies, associated fees, tips, and taxes. Different entities can update internal regions of a file under predefined terms, with RNS dynamically adjusting to point to the latest version. Instead of implementing an isolated tokenomics model, the storage tokenomy can be unified with other components through a message exchange mechanism. This mechanism provides secure message exchange between components, enabling seamless information and value coordination across the system without isolating components state or tokenomics.

The computation component is based on a verifiable computation environment. It is closely integrated with decentralized data storage, where users upload calculation requests consisting of program code, input data, and an attached tip to incentivize resource providers. Providers execute the requested computations and then return the result along with proof of execution which is stored next to the original request. This model offloads complex calculations from the Layer 1 network, unrelated to financial operations, significantly reducing computation costs associated with on-chain calculations.

The financial component implements the Layer 1 blockchain using established and proven technologies. While it incorporates existing battle tested elements such as consensus mechanisms and smart contracts, its primary innovation lies in its deep integration with the storage and computation protocols, along with the introduction of new transaction schema, a new gas operation paradigm, and new smart contract operations, enabling gas top up while operation execution. A multi-operation transaction can execute multiple tasks sequentially, where the transaction itself may be initiated with a zero balance and a first operation swapping a traded token into gas, which covers subsequent transaction execution costs. The concept of a post-operation gas check is critical to enabling this flow. This approach eliminates the need for users to hold native coins to pay for gas prior to the transaction while preserving a simple and clear transaction structure.

Each component provides standardized interfaces for cross-component interactions and features global shared tokenomy.

# 5 Enhanced User Experience

The adoption of new technologies is heavily driven by the ease of user onboarding and interaction with blockchain ecosystems. Solana demonstrated strong adoption within the meme ecosystem largely due to technical optimizations that reduced onboarding times from hours to minutes.

One way to improve user's experience is to let users interact with dApps without installing additional wallet software - which we refer to as Walletless experience. By supporting walletless user interactions, the protocol removes a major onboarding barrier. Users can interact with decentralized applications securely, with-

out needing to install separate wallet software.

To further improve user experience, blockchain will feature a simple mechanism to cover transaction fees in any token along with deep integration across computation and storage layers. This enables a new generation of serverless, general-purpose applications.

Combination of these two mechanisms, integrated and aligned together - walletless UI and gasless transactions - eliminate two major barriers to mass adoption:

- No need to download and initiate 3rd party wallet software.
- No need to hold native blockchain tokens upfront the transaction.

This simplifies the user journey, bringing dApps much closer to mainstream usability.

# 6 Decentralized storage

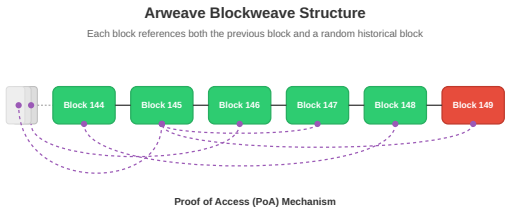
Our decentralised storage protocol is based on Arwave [7], with some modifications. The storage features following critical properties: deterministic resource addressing, resource name service, resource structure manifestation and tip allocation, along with traditional properties like permanent availability and tamper resistance. Files stored in this system receive unique identifiers that simultaneously serve as both blob pointer and checksum, ensuring content integrity. The high replication factor of stored data guarantees permanent availability across the network.

This component serves multiple roles: a general-purpose file storage layer, a memory backend for executable programs, a decentralized UI delivery, a decentralized database and data manifestation, a messaging medium, and a foundational layer for decentralized DNS.

## 6.1 General-purpose File Storage Layer

At the core of our decentralized infrastructure lies a robust, general-purpose file storage layer inspired by the Arweave protocol. This layer is designed to provide long-term, immutable data storage through content-addressable architecture. Every file added to the network is assigned a unique hash, which functions both as an identifier and a cryptographic checksum. This guarantees the integrity of stored data and enables deterministic retrieval - a file can always be fetched using its original hash.

In line with Arweave's philosophy, our system ensures permanent availability through a high-redundancy replication model, where data is distributed across many nodes. This redundancy not only safeguards against data loss but also reinforces censorship resistance and resilience. A crucial component of Arweave's architecture is its unique consensus mechanism based on the Blockweave, a blockchain-like structure where each new block is cryptographically linked not only to the immediately preceding block, but also to a randomly selected previous block from the chain's history. This design, known as Proof of Access (PoA), requires miners to demonstrate knowledge of randomly selected historical data in order to produce a valid block. As a result, miners are incentivized to store as much of the network's historical data as possible, reinforcing decentralization and durability. This contrasts with traditional blockchains, where miners only need to maintain the latest state. By requiring deep access to the archive, Arweave ensures long-term availability and embeds data permanence directly into the mining process.




This layer is not only foundational for static file storage but also supports a variety of applications, including executable code distribution, decentralized frontend delivery, and composable data infrastructure for higher-order decentralized services.

## 6.2 Redefining the Arweave Transaction Format

To better align the protocol with generalized computation, modular storage, and service-level coordination, we introduce a fundamental redefinition of the standard Arweave transaction structure. This evolution departs from the original currency-transfer model in favor of a more expressive, action-oriented architecture, optimized for programmability and decentralized infrastructure orchestration.

In the revised format, the target and quantity fields will be removed entirely. These fields, originally designed to facilitate token transfers between wallets, are no longer relevant in the context of execution-driven storage and programmatic task resolution. Instead, the focus shifts from monetary movement to declarative intent and atomic interaction with the network's decentralized components.

The central element of the new transaction model is the action list - a sequential collection of discrete operations encoded within each transaction. Each action is self-contained and may perform one or more of the following: commit data to permanent storage, allocate space for deferred writes, assign metadata to the allocated space, which can define write validation rule (e.g., ZK-proof requirement), and optionally a tip to incentivize external actors to fulfill the allocated space. This model introduces a clear boundary between what the transaction author declares and what the network fulfills, enabling asynchronous and verifiable coordination.

To ensure network stability and enable efficient validation, each individual action is restricted to a maximum payload of 256KB. Additionally, a transaction may include up to 40 such actions (chunks), resulting in a per-transaction ceiling of 10MB. This granular structure aligns with the original Arweave transaction's data limitations, preserving compatibility with existing node architectures while enabling transactions to express richer, multi-step behaviors in a single, verifiable unit.

## 6.3 Storage Backend for Executable Programs

The decentralized computation model resembles a traditional CPU, where program code and input data share a common memory space. Resource providers monitor a live feed of available computation tasks [8], each containing a reference to the program, input, and an execution tip. Upon claiming a task, a provider runs the program, computes the result, and generates a cryptographic proof of correct execution. The result and proof are then stored, and a validation mechanism ensures integrity before distributing the tip.

The execution environment is based on a RISC-V architecture, using Risc Zero's verifiable virtual machine as a foundation. Unlike traditional blockchain environments that prioritize deterministic, state-bound execution, this model is designed for flexible and general-purpose computation, verified off-chain. The virtual machine will be extended for performance while maintaining verifiability.

For further details check Section 7.1.

### 6.3.1 Task Feed Mechanism and Proof-Driven Settlement

The decentralized computation layer functions through a dynamic task feed system, acting as the core coordination point between resource providers and requesters. This feed broadcasts computation requests across the network, each specifying a program hash (pointing to the executable stored on the decentralized storage layer), input parameters, and an execution tip denominated in the network's native coin. Resource providers subscribe to this feed using a pub-sub protocol, filtering tasks based on computational demands, hardware capabilities, and economic incentives.

In the revised architecture, each computation request is written as a structured metadata object into the storage layer before it enters the task feed. This metadata includes references to the program code, the computation input, pre-allocated storage spaces for both the result

and the proof, a description of the proof validation mechanism (e.g., a zk-SNARK verifier), and the tipping information that defines the reward conditions. Once published, nodes monitor the storage layer for new requests and begin exhibiting them as part of a continuously updated stream. External listeners, such as computation providers, can subscribe to this stream to receive new requests as they become available.

Upon claiming a task, the provider launches the referenced program within a sandboxed RISC-V virtual machine, processes the input, and produces two essential outputs: the computation result and a zero-knowledge proof (zk-SNARK) certifying correct execution. This proof guarantees that the program was executed faithfully according to its immutable bytecode.

Both the result and the proof are submitted to the decentralized storage layer as another transaction, containing a single action that attempts to fulfill the allocated space. If the submitter successfully verifies the cryptographic validity against the program's original hash stored on-chain and confirms that the input and output data align with the task specifications, the submission is accepted and a message with tip releasing is submitted to the financial layer.

## 6.4 Decentralized UI Delivery

Modern web technologies are evolving, and decentralized UI mechanisms are being actively researched. As mentioned earlier, from the browser's perspective, a web application consists of an HTML document served on request, along with additional CSS and JS files loaded through subsequent requests. Traditional domain-based systems introduce centralization risks, where domain ownership can become a single point of failure or a target for censorship. With deterministic addressing, content is directly verifiable, making any tampering by storage providers or intermediaries detectable at the time of delivery.

Reliable delivery of content to browsers has been researched, discussed, and addressed for a long time. Partially, this situation has been improved, but there remains space for further enhancements, which are actively pursued by multiple research groups, especially to counter the increasing threat and sophistication of content tampering attacks.

One partial solution is the mechanism called Subresource Integrity (SRI), which adds a checksum to secondary files so that the browser can verify their contents against this checksum when loaded. This mechanism ensures tamper-resistant delivery of side content within a webpage. However, SRI only protects secondary resources and cannot guarantee the integrity of the HTML document itself. A similar mechanism for root HTML verification is currently being researched and actively discussed.

One promising approach involves DANE with DNSSEC, where cryptographic records in DNS, secured by DNSSEC, can authenticate services or content. Building on that, a DNS-based "Content Verification" mechanism is under active discussion. This would allow domains to publish content hashes via DNS TXT records, enabling browsers to verify content integrity at load time, which is discussed in detail in Section 6.7. Such a system could solve key trust issues without relying on third parties. It is also possible that privacy-focused browsers like Brave could adopt these features before mainstream browsers such as Chrome.

There are certain advantages to combining decentralized storage with decentralized DNS services, which make TXT records more suitable for content verification. However, advanced security considerations must be preserved to ensure the necessary content safety measures.

### 6.4.1 Walletless UI

Web standards are evolving, and a new HTML element, FencedFrame, is now supported in Chrome. It is designed to provide secure isolation for web components within a parent web page. Think of it as a sandbox environment, similar to those used by browser extensions like

crypto wallets, which could serve as inspiration for future implementations.

Traditionally, browser-extension-based wallets operate in isolated namespaces (e.g., `chrome-extension://<ID>`) and communicate with the main page via the postMessage mechanism. This architecture ensures that private keys remain secure within the extension's context.

Inspired by this model, it is conceivable that a wallet interface could securely run within a dApp's web page using the FencedFrame context, offering similar private key isolation. However, this concept requires meticulous configuration and comprehensive security measures to ensure adequate protection. For example, browser-level safeguards, including content namespacing and strict origin policies, help prevent unauthorized access from insecure elements like iframes. These protections lay the foundation for a walletless model.

A critical component of this setup is establishing a secure communication channel between the web page and the wallet. Since FencedFrame is intentionally designed not to support postMessage, a potential workaround could involve a decentralized messaging mechanism. In this model, the browser and wallet would establish a communication session via a decentralized storage infrastructure and exchange messages. This approach would be somewhat akin to WalletConnect, but with a significantly higher degree of decentralization.

Browsers are actively moving in this direction. Emerging features like WebAssembly sandboxing, WebContainers, and the future evolution of isolated iframes all point toward a future where embedded wallet interfaces could be implemented in new ways, each offering security guarantees comparable to those of traditional blockchain extensions. FencedFrame represents the first and most immediate step in this journey. In the coming years, it's likely that we'll see multiple implementations of secure, walletless experiences tailored to various use cases, all providing the required levels of security.

**Secure updates** The problem of secure version upgrade can be addressed via RNS pointing to the latest version of the UI. Code integrity verification can be performed by the browser upon content arrival, with additional protections such as watermarks embedded inside the container to prevent spoofing.

**Seed phrase backup and recovery** The issue of private key or seed phrase backup can be solved through a cross-browser migration mechanism, where an encrypted seed backup is transferred between user devices over the protocol's infrastructure. This acts as an additional layer of security for protecting users' keys during device changes, while also addressing the existing complexities of seed phrase storage and management.

**Prospectives** By combining decentralized UI delivery with in-browser wallet execution, users gain a seamless experience: no downloads, no extensions, and no native tokens required upfront. This significantly reduces onboarding friction and supports mass adoption without compromising security or decentralization.

### 6.5 Decentralized Database and Data manifestation

The decentralized equivalence of databases builds upon Arweave's immutable storage model by introducing a dynamic, chunk-based architecture in which files are decomposed into granular units. Each chunk is assigned a content-derived identifier and indexed within an associated Merkle-Patricia trie. This structure enables efficient tracking of revisions and supports partial updates.

At the core of the system is the manifest file: a structured metadata schema that coordinates data structure, access permissions, validation logic, and economic incentives. The manifest defines policies for scalable storage (e.g., expansion or compaction), and embeds cryptographic references to external resources such as validation programs, related datasets, or executable modules.

Data integrity is maintained through a validation layer. When a chunk is updated, the network validator executes the specified validation logic, according to manifest file's specifics. The verifier confirms that the new chunk complies with the manifest's rules, such as maintaining schema integrity, enforcing ownership permissions, or satisfying economic conditions. Approved changes are recorded immutably. Chunk modifications incur costs based on their size and validation complexity. These payments are assigned to the validator.

### 6.6 Messaging Medium

To enable secure, transient messaging between off-chain parties, a lightweight communication protocol is built on top of the decentralized storage layer. This system operates in a decentralized fashion similar to WalletConnect, allowing participants to establish encrypted communication channels without relying on centralized servers or persistent on-chain transactions. Messages are treated as ephemeral data chunks with defined time-to-live (TTL) values and are temporarily stored in a message pool. Unlike permanent storage operations, these messages are not mined into the canonical chain and are excluded from long-term replication, reducing storage overhead and enabling low-latency delivery.

Secure channel establishment begins with a cryptographic handshake for mutual authentication. Party A initiates communication by generating an ephemeral key pair and broadcasting an encrypted message. This message includes Party B's public key, a TTL (e.g., 5 minutes), and a session key derived from Party A's ephemeral private key and Party B's public key. The message is distributed across the network and temporarily held in the recipient's inbound message pool.

Party B monitors their pool, locates the message addressed to them, and decrypts the session key using their private key. All subsequent messages between the parties are then encrypted using AES-GCM with the shared session key, ensuring confidentiality and integrity throughout the session.

### 6.7 Foundational Layer for Decentralized DNS

TODO

## 7 Decentralized Calculations

The decentralized computation model mirrors the architecture of a traditional CPU, where both program code and input data coexist in a shared memory space, which is a decentralized storage in our case. Within this system, calculation resource providers continuously monitor a live feed that broadcasts newly available computation tasks. Each task includes a reference to the program code, input data, and an associated tip to incentivize execution.

Once a provider selects a task, it executes the designated program, computes the result, and generates a cryptographic proof attesting to the correctness of the execution. Both the result and the proof are then written into a designated storage location. A decentralized storage's validation mechanism independently verifies the proof before the computation is accepted and the tip is distributed. This ensures the integrity and reliability of results within a tamper-resistant environment, establishing trust in the outputs without requiring trust in the executor.

The execution environment for these tasks is to be RISC-V compliant, leveraging the capabilities and architecture pioneered by Risc Zero. Their virtual machine serves as a robust foundation, offering a verifiable execution environment tailored for general-purpose computation. Unlike traditional smart contract VMs, which are often narrowly scoped and optimized for deterministic blockchain operations, the adapted machine in this model must support a broad range of computational tasks beyond on-chain logic.

To meet these demands, the virtual machine must be extended and optimized to handle diverse computational workloads while preserving its core feature: generation of verifiable ex-

ecution proofs. Risc Zero's work provides both the architecture and tooling required to achieve this, making it an ideal starting point for building a scalable and secure decentralized computation layer.

## 7.1 Computations

The decentralized computation model employs a RISC-V-compliant execution environment [8], extending principles pioneered by Risc Zero to enable verifiable general-purpose computation. Program code and input data reside in a decentralized storage layer, functioning as a shared memory space accessible to computation resource providers. These providers monitor a dynamic task feed, where each task specifies the program code, input parameters, and a tip to incentivize execution. Upon selecting a task, the provider executes the program against the input data, leveraging Risc Zero's virtual machine (VM) to generate a succinct cryptographic proof of correct execution.

The cryptographic proof, generated via zero-knowledge succinct arguments (zk-SNARKs), attests that the program was executed faithfully according to its defined logic. This proof, alongside the computed result, is committed to decentralized storage, where validators independently verify its authenticity. The verification process ensures computational integrity without relying on trust in the provider, establishing a tamper-resistant framework. Successful verification triggers the release of the tip to the provider, aligning incentives with honest participation while maintaining auditability.

Risc Zero's VM architecture is central to this model, combining RISC-V instruction set compatibility with specialized extensions for proof generation. Unlike conventional smart contract environments constrained by deterministic execution and limited computational scope, this VM supports arbitrary computations, including complex algorithms and data-intensive workloads. The VM operates as a verifiable runtime, isolating program execution while preserving cryptographic traceability. This design enables the execution of off-chain logic with on-chain verifiability, bridging decentralized and traditional computing paradigms.

### 7.1.1 Cryptographic Proof Generation

Risc Zero employs a zk-STARK-like system [9] internally adapted to generate zero-knowledge proofs of computation [8]. At its core, this involves generating a cryptographic claim that a RISC-V program $P$ was executed on input $x$ to produce output $y$, without revealing internal state transitions and without the verifier having to rerun the computation.

**Formal Statement of the Proof** The system constructs a proof $\pi$ for the following statement:

$$\pi : \text{I know a program } P \text{ and an input } x \text{ such that } y = P(x) \tag{1}$$

This is backed by a commitment to the execution trace:

$$\text{commit}(P, x, y) \to \pi \tag{2}$$

Where:

- $P$ is compiled into a sequence of RISC-V instructions.
- $x$ is the public or private input.
- $y$ is the program output.
- $\pi$ is the cryptographic proof that attests to the correctness of the computation.

**Execution Trace and Hashing** The RISC-V VM generates an execution trace $T = \{s_0, s_1, ..., s_n\}$, where each $s_i$ is the state of the VM at step $i$. The prover constructs a hash chain or Merkle structure of these states:

$$H = \text{MerkleRoot}(s_0 \parallel s_1 \parallel ... \parallel s_n) \tag{3}$$

This Merkle root $H$ is included in the proof and commits to the full execution trace, ensuring that any tampering with individual steps can be detected.

**Proof-of-Execution via Polynomial IOPs** The system models the execution trace as a sequence of polynomials over a finite field $\mathbb{F}_q$. Let $f_1, f_2, ..., f_k \in \mathbb{F}_q[x]$ represent polynomial encodings of CPU state transitions. The prover then constructs an interactive oracle proof (IOP) to show that:

$$\forall x \in D, (f_1(x), ..., f_k(x)) \in \text{ValidStateTransitions} \tag{4}$$

Where $D$ is a low-degree domain. The verifier checks that all transitions obey the VM's operational semantics, without seeing the full trace.

**Fiat-Shamir and Non-Interactivity** To transform the interactive proof into a non-interactive one (NIZK), Risc Zero uses the Fiat-Shamir heuristic, replacing random challenges with hash-derived values:

$$\text{Challenge} = \text{Hash}(P, x, y, H) \tag{5}$$

This enables the construction of a compact proof $\pi$ that can be verified independently without interaction.

**Verification Equation** The verifier then checks that:

$$\text{Verify}(P, x, y, \pi) = \text{true} \tag{6}$$

This verification can be done efficiently in time $O(\log n)$ using the Merkle commitments and polynomial evaluation checks, without re-executing $P$.

## 7.2 Service Payments

To facilitate secure and autonomous payment for off-chain computation, the system integrates the cryptographic proof (as described above) with the verification rules explicitly defined in the task's metadata. These two elements together form the condition for tip release. To ensure that only the legitimate computation provider can claim the reward, the provider's reward address must be submitted as the final argument in the input data prior to execution. This binding ensures cryptographic linkage between the computation and the identity of the executor.

Upon completion, the computation result is submitted to the network as an Arweave transaction containing a single action that claims to fulfill the allocated space and release the tip. This action includes the result, the cryptographic proof of correct execution, and the reward address. All of these elements are passed to the verification mechanism, which validates that the result matches the expected output, the proof confirms correct execution according to the specified program, and the reward address matches the one originally embedded in the input. Only upon successful verification of all three components is the tip unlocked and transferred to the rightful provider.

## 7.3 Delayed execution

The protocol natively supports delayed transactions through programmable validation logic embedded within computation tasks. Users can encode a target block height directly into the task's program data, specifying that the result may only be validated and finalized if the current blockchain block exceeds this predefined threshold. This mechanism enforces a mandatory waiting period between task submission and execution, enabling use cases such as time-locked computations or deferred contract settlements. For example, a task programmed to activate after 12 hours would embed a target block number derived from the network's average block time, ensuring execution cannot occur until the requisite delay has elapsed.

When a computation resource provider executes such a task, the program logic first verifies that the blockchain's current block height, accessible via a decentralized oracle or on-chain state query, meets or exceeds the target. This check is cryptographically bound to the computation's execution trace, ensuring it is included in the generated zero-knowledge proof. Validators subsequently re-verify this condition during proof validation by cross-referencing the target block against the blockchain's finalized state. If

the block height is insufficient, the proof is rejected, and the associated transaction remains in a pending state until the required blocks are mined.

### 7.4 Recurrent calculations

The protocol's recurrent computation framework leverages TON's native cross-chain messaging system to enable autonomous, self-sustaining computation workflows. When a computation result and its corresponding proof are submitted, they trigger a tip-carrying message from the decentralized storage to a designated smart contract within the financial layer. This message triggers a smart contract that initiates follow-up logic, such as scheduling the next computation round.

In combination with delayed transaction mechanics, this forms a Cron-like automation system. For example, a smart contract could schedule a task to execute every 12 hours by setting a target block height derived from the network's block time. Each cycle's proof submission generates a new message that resets the delay, thus perpetuating the loop.

## 8 Decentralized Financial layer

The decentralized web demands a new class of financial infrastructure—one that goes far beyond basic token transfers. A fully capable decentralized financial layer must support high transaction throughput (TPS), a powerful virtual machine for expressive programmability, scalable network sharding, application-specific chain branching, and a messaging mechanism that enables both value transfers and cross-layer action execution, particularly in coordination with decentralized storage systems. Among existing protocols, the TON protocol already meets many of these criteria [5] and has proven itself through real-world deployment and performance.

However, to meet the specific needs of our architecture, our financial layer will require key modifications. This includes a redesigned transaction schema supporting multi-operation transactions and a migration to a RISC-V-compatible virtual machine. Such changes will ensure compatibility between on-chain smart contracts and off-chain computational tasks, enabling both to be written, verified, and executed within a unified SDK and programming model.

### 8.1 Network sharding

The decentralized web demands a new class of financial infrastructure - one that extends far beyond simple token transfers. A fully capable decentralized financial layer must deliver high transaction throughput (TPS), support a general-purpose virtual machine, enable scalable network sharding, allow branching into application-specific chains, and offer a robust messaging mechanism capable of both value transfer and action execution - particularly for interacting with decentralized storage. Among existing technologies, the TON protocol stands out as a battle-tested solution already encompassing most of these critical features. Its ultimate strength lies in its native support for network sharding, which enables the seamless creation of application chains - dedicated subnets designed to serve the unique requirements of individual decentralized applications. TON also provides a built-in, transparent cross-chain data migration mechanism, allowing messages and state to flow effortlessly between shards and app-chains without introducing complexity to the end user.

However, to meet the demands of our architecture, additional enhancements are necessary. Our financial layer will extend the TON transaction schema to support multi-operation transactions, enabling more expressive and atomic logic within a single message.

#### 8.1.1 dApps sharding economy

While the network's sharding mechanism will remain largely consistent with the original protocol-where shard chains can be spun up at

minimal cost-we propose introducing a financial incentive for launching dedicated dApp shard chains through native coin staking. This addition is expected to enhance the overall tokenomics and provide clear economic benefits to the associated shard chains.

## 8.2 Gasless transactions

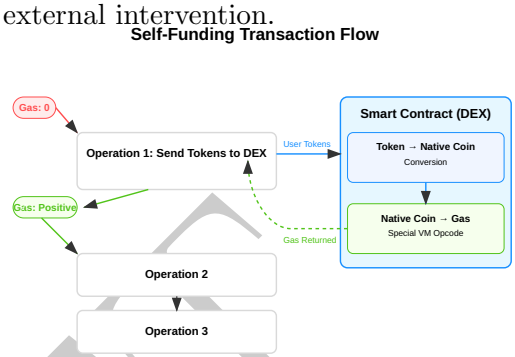
A significant advancement in blockchain user onboarding lies in the concept of gasless transactions, a model where transactions can begin with zero gas and acquire the required gas during execution. This approach eliminates the need for users to pre-fund their wallets with native tokens, thereby streamlining the first-time user experience with decentralized applications (dApps).

Traditionally, blockchain transactions must be fully funded with gas upfront and are typically limited to a single atomic operation. This conventional structure imposes constraints on composability and presents a usability hurdle for new users unfamiliar with token economics. However, modern blockchain platforms such as Solana introduce more flexible transaction paradigms. These allow multiple operations within a single transaction, increasing expressiveness and enabling sophisticated execution flows that redefine the interaction between users and decentralized systems.

Building upon this foundation, we propose a multi-phase, self-funding transaction model. In this model, a transaction is structured to:

- Start with zero gas.
- Execute an initial operation that swaps a user-held token (e.g., USDC, DAI) for the native blockchain token via an integrated decentralized exchange (DEX).
- Invoke a novel virtual machine opcode to convert the acquired native tokens into usable gas.
- Continue with subsequent operations, each checking that the available gas is sufficient to proceed.

This mechanism ensures that execution remains atomic and secure, while also enabling the transaction to fund itself dynamically. As a result, the transaction becomes capable of paying for its own execution using assets already held within the initiating wallet, without requiring any prior balance of the native token or external intervention.




## 8.3 RISC-V virtual machine

To unify on-chain smart contracts and off-chain verifiable computations, the virtual machine powering our financial layer will be based on the RISC-V architecture. This choice is strategic and foundational: RISC-V offers a modern, open, and extensible instruction set that aligns directly with the design of Risc Zero's zero-knowledge virtual machine, which our system uses to generate cryptographic proofs of off-chain computation. By standardizing on RISC-V, we enable developers to write both smart contracts and verifiable off-chain tasks using the same tooling and development environment. This dramatically simplifies the execution model, allowing a single codebase and SDK to target both domains, and fosters better optimization, portability, and security.

The decision to adopt RISC-V also follows a growing trend in blockchain architecture - echoed by research and proposals in leading ecosystems like Ethereum [10] - to move beyond legacy virtual machines such as the EVM. These systems face inherent scalability and provability bottlenecks. RISC-V, by contrast, is not only more efficient and modular, but also more amenable to zero-knowledge proof generation. Its regular, predictable instruction set significantly reduces the complexity and cost of zk-

proof generation compared to traditional smart contract bytecode, making it an ideal foundation for future-proof decentralized infrastructure.

### 8.4 Smart contract language

To bootstrap the developer experience and align with the performance demands of verifiable execution, our smart contract development environment will initially be based on Rust. Rust offers a strong, modern systems programming foundation with powerful type safety, memory safety without garbage collection, and a mature ecosystem of libraries and tooling. With an estimated developer base of over 3 million, Rust has become a preferred language for building high-performance, secure systems, including blockchain runtimes and cryptographic applications. Its compatibility with the RISC-V architecture and existing support in the Risc Zero virtual machine make it the ideal starting point for building verifiable decentralized applications.

#### 8.4.1 AssemblyScript

Looking forward, we plan to introduce support for AssemblyScript, a lightweight, compiled subset of TypeScript. AssemblyScript combines a familiar JavaScript-like syntax with strong typing, making it accessible to the massive ecosystem of web developers—who represent approximately 70% of all software developers globally. This addition will significantly lower the barrier to entry for building decentralized web services, enabling frontend and full-stack developers to participate in decentralized infrastructure development. AssemblyScript is actively promoted and supported by major industry players such as Shopify, Fastly, and Cloudflare, who value its balance of performance, developer ergonomics, and compatibility with WebAssembly (WASM).

However, the primary challenge lies in adapting AssemblyScript's compilation pipeline to target RISC-V instead of WASM. While this is technically feasible, there are currently no ready-made toolchains or compilers for this transformation. Addressing this gap will be a strategic priority for our development roadmap, as it will unlock a vast pool of talent and bring modern web development paradigms to the decentralized execution layer—bridging the gap between Web2 and Web3 through a unified, verifiable programming model.

## 9 Areas of Application

The protocol's comprehensive integration of decentralized storage, computation, and financial layers enables the development of next-generation decentralized applications across multiple domains.

### 9.1 Web3 payments in Web2 websites

The protocol lowers the onboarding barrier for Web3 by eliminating the need for users to download separate wallet software. This allows any Web2 application to begin integrating Web3 payment flows with minimal friction. Users can participate in the protocol's economy directly through participating Web2 services, using tokens earned or acquired from different sources to pay for transactions across other Web2 or Web3 applications.

### 9.2 Decentralized Gaming

Decentralized Gaming platforms can leverage the computation layer for game logic and state management while using decentralized storage for game assets. The web browser gaming market reached $20.1 billion in 2023 [11] and is projected to hit $49.2 billion by 2030, with these numbers representing only web2 gaming estimates The complete yearly gaming market is estimated at $250 billion, taken into account other platforms. Several DeFi gaming projects have already demonstrated viral potential, with some reaching millions of users, though network congestion often degraded user experience and hindered sustained growth.

Besides economical advantages, a web3 economy brings clear advantages to Massive Multiplayer Online games (MMOs), Approximately 75% of gaming support tickets relate to item trading issues, where users face fraud on third-party platforms and subsequently submit complaints to game support teams. The protocol's integrated trading system eliminates these risks by providing secure, transparent item trading directly within the gaming ecosystem.

The protocol enables new monetization models for existing games through the introduction of unique in-game items and currencies. This represents one of the most practical applications of NFT technology, providing true ownership and verifiable scarcity while maintaining familiar gaming experiences. Crucially, end users remain abstracted from traditional blockchain complexities – they no longer need to download specialized wallet software or purchase native blockchain coins, processes that currently exclude approximately 98% of the web2 user base.

Furthermore, as gaming communities grow, development teams can seamlessly migrate their smart contracts and other L1 data to dedicated sidechains when needed. This ensures user experience remains unaffected by network congestion, while maintaining all benefits of decentralized asset ownership and trading. The migration process remains transparent to users, preserving their gaming experience while scaling to meet demand.

Traditional web2 gaming faces significant challenges that can now be addressed without the complexities introduced by current web3 solutions. The scalability features of our protocol, including dedicated sidechains, prevent the network congestion issues that plagued previous successful blockchain games, ensuring sustained quality of service even during periods of viral growth.

This architecture enables truly ownable in-game items, verifiable randomness for game mechanics, and transparent game economies while maintaining the accessibility and user experience expected by mainstream gamers.

### 9.3 Decentralized Social Networks

Decentralized Social Networks can be built with complete transparency of content distribution algorithms and moderation policies. User data remains under individual control while the network's operation becomes fully auditable. Content persistence is guaranteed through decentralized storage, while the computational layer enables sophisticated feed algorithms and content recommendation systems.

### 9.4 Pay-to-Stream Video Content and Micropayments

The protocol enables continuous payment streaming for video content, where access is granted in real-time based on ongoing micropayments. Video segments are served from storage providers with deterministic addressing, while the computation layer manages access control and payment verification. Content providers receive payments continuously as users stream, with automatic content access termination if payments stop. This enables new monetization models where viewers can start and stop paying at any point, paying only for watched content.

### 9.5 Decentralized Messaging

The protocol enables truly decentralized messaging through deterministic addressing and private computations using encryption techniques. Users control their inboxes through address ownership, with messages stored in encrypted form in decentralized storage. The computation layer handles message encryption and routing, while spam protection operates through multiple configurable mechanisms, including optional pay-to-message requirements where recipients can require payments for message delivery. These anti-spam measures can be combined and customized per user, similar to modern SMTP filters but with economic incentives. The system remains censorship-resistant as message blocking occurs only through user-defined rules rather than central authorities.

Address abstraction enables seamless integration with existing email clients through local bridge software, maintaining familiar user experiences while leveraging decentralized infrastructure.

### 9.6 Decentralized Advertising Networks

User-Centric Advertising Networks reverse the traditional advertising model by compensating users for their attention. Smart contracts automatically distribute payments for ad views, while decentralized computation ensures accurate tracking and fair distribution of advertising revenue. This model introduces a revolutionary paradigm shift where web browsers, as crucial infrastructure providers, can finally monetize their services sustainably. Unlike the Web2 paradigm, where browsers struggle to find reliable revenue streams without compromising user privacy, this system enables direct compensation for browsers' role in maintaining the decentralized ecosystem. The transparent nature of the system prevents fraud while protecting user privacy, creating a fair economic model that rewards all participants in the advertising ecosystem.

### 9.7 Digital Art and Intellectual Property Ownership

Current NFT implementations face a fundamental flaw: most store only links to digital assets rather than the assets themselves, compromising true decentralized ownership with ease of substituting the original content. Our protocol solves this through native decentralized storage, enabling direct ownership of large files such as 3D models, technical documentation, and patents. This allows for verifiable ownership of substantial digital assets - from encrypted technical patents to complex 3D models - with guaranteed persistence and immutability. The system's deterministic addressing ensures that owned content remains permanently accessible and unalterable, creating a robust foundation for digital property rights beyond simple tokens. These applications represent early use cases, with the protocol's flexibility enabling new forms of decentralized services as the ecosystem evolves.

## 10 Final words

The evolution of internet architecture has reached a critical juncture where decentralization is not just technically feasible but increasingly desired. Our protocol represents a comprehensive solution to the challenges that have historically limited the adoption of truly decentralized applications. By integrating decentralized storage, computation, and financial infrastructure into a cohesive framework, we provide the foundation for a new generation of applications that maintain user sovereignty while enabling unprecedented functionality.

The protocol's innovative approach to decentralized storage and computation, combined with its flexible transaction model and system-level DEX integration, creates an environment where developers can build sophisticated applications without compromising on decentralization principles. The ability to process transactions in any token and seamlessly handle complex computations through delayed transactions removes significant barriers to mainstream adoption.

Our architecture's emphasis on scalability through network sharding and dedicated application chains ensures that the protocol can grow to meet increasing demands while maintaining performance and security. The integration of both system programs and smart contracts provides the flexibility needed to support diverse use cases, from core protocol functions to general-purpose applications.

The potential applications across social networks, advertising, gaming, and digital art demonstrate the protocol's versatility and its capacity to address real-world needs. By enabling true ownership of digital assets, transparent operation of network services, and fair compensation models for all participants, the protocol lays the groundwork for a more equi-

table digital economy.

As we move forward, the success of this protocol will be measured not just by its technical achievements, but by its ability to enable new forms of human interaction and economic activity in the digital sphere. The framework we have advised provides the tools necessary for developers and entrepreneurs to build the next generation of decentralized applications, bringing us closer to the original vision of a truly decentralized internet.

## References


[1] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system". In: *Decentralized Business Review* (2008), p. 21260.

[2] Vitalik Buterin et al. "Ethereum: A next-generation smart contract and decentralized application platform". In: *White paper*. Vol. 3. 37. 2014.

[3] Gavin Wood et al. "Ethereum: A secure decentralised generalised transaction ledger". In: *Ethereum project yellow paper* 151.2014 (2014), pp. 1–32.

[4] Solana Foundation. *Solana Documentation*. 2022. URL: `https://docs.solana.com` (visited on 03/15/2025).

[5] Nikolai Durov. "Telegram Open Network". In: *White Paper*. 2018.

[6] IDC. *Global Digital Economy Forecast, 2023-2028*. Tech. rep. International Data Corporation, 2023.

[7] Sam Williams and William Williams. "Arweave: A Protocol for Economically Sustainable Information Permanence". In: *White Paper*. 2019.

[8] Brian Fiege, Nick Braithwaite, and Jeremy Church. "RISC Zero: A Zero-Knowledge Virtual Machine". In: *Technical Report*. 2022.

[9] Eli Ben-Sasson et al. "Scalable, transparent, and post-quantum secure computational integrity". In: *IACR Cryptol. ePrint Arch.* Vol. 2018. 2018, p. 46.

[10] Wonji Kim et al. "Ethereum RISC-V: Implementing the Ethereum Virtual Machine with a RISC-V Backend". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2022, pp. 283–303.

[11] Newzoo. *Global Games Market Report*. Tech. rep. Newzoo, 2023.