Zero Agents: A Ledger-Backed Multi-Agent Intelligence Layer for Web3 and Web2

Litepaper

Abstract. *Agents are rapidly becoming the default interface to software, but practical autonomy in open financial environments is limited by three gaps: missing provenance for the prompts that steer agent behaviour; fragmented, noisy, cross-chain data; and execution risk in adversarial markets. This litepaper presents* Zero Agents*, a modular architecture, research and simulation layer, and guarded execution environment powering* Zero Chat*. We propose a provenance-first mechanism for* prompt mining *built on ERC-7208 on-chain data containers. The ledger gives prompts canonical identity, lineage, telemetry, and micro-royalties and enables stake-backed judging and validation. Combined with an AI-ready cross-chain knowledge graph and a preview-first execution layer, Zero Agents turn intelligence into a composable asset while keeping humans-in-the-loop for high-impact actions. We survey related work in multi-agent LLMs, graph-based retrieval, and blockchain security; formalise the economic loops that bind usage to token utility; and provide an audited roadmap from analysis to simulation to controlled execution.*

Contents

1 Introduction

LLM-based agents can now read, write, and operate tools with competence that was out of reach only a few months ago. The literature has converged on a simple but powerful recipe: make reasoning explicit, equip the model with tools, and provide mechanisms for self-checking. ReAct interleaves chain-of-thought with external actions to reduce hallucinations and to keep plans grounded in evidence [2]. Toolformer shows that language models can learn *when* to call APIs and *how* to use their outputs by distilling their own training signal from unlabeled text, enabling tool use without expensive human annotation or task-specific fine-tuning [3]. Yet these advances do not immediately translate into safe autonomy in Web3. Public ledgers are unforgiving: data is fragmented across contracts and chains; transactions are irreversible; and adversaries exploit transparency to reorder, sandwich, or censor in the mempool [13]. Most importantly, there is no canonical way to identify and reward the prompts, research templates, and policies that generate good outcomes. The lack of *provenance* means that useful prompts disappear into chat logs; there is no measure of downstream impact; and builders have little incentive to share or curate their best work.

We argue that a viable *agent economy* requires three enabling layers. **First**, an AI-ready, cross-chain representation of state and context is needed for correct answers, reproducible simulation, and post-hoc audits. **Second**, multi-agent orchestration must be treated as a first-class programming model with typed messages, shared memory, and tool policies so that research, risk assessment, and execution can be composed safely. **Third**, contributions must be measurable and rewardable. Intelligence, in the form of prompts, templates, and policies, should be addressable objects with identity, lineage, and telemetry so that reuse, royalties, and accountability become automatic.

This litepaper introduces *Zero Agents* and the product they power, *Zero Chat*. Zero Agents bring together (i) a cross-chain knowledge graph; (ii) multi-agent research and simulation; (iii) a guarded execution layer with human-readable previews and policy vaults; and (iv) an on-chain *prompt mining* mechanism anchored by ERC-7208 on-chain data containers [25, 26]. Prompt mining gives prompts a canonical handle, records where and how they were reused, and routes micro-royalties according to explicit policies. Coupled with stake-backed judging and validation, the ledger provides a transparent market for quality. The remainder of the paper develops the architecture, the ledger and its economics, and a delivery roadmap with concrete acceptance tests.

1.1 Contributions

Our contributions are practical and testable:

(a) An end-to-end architecture for community agent orchestration in Web3, integrated into a shipped product, *Zero Chat*, that is live in public beta and backed by real multi-chain data coverage.[1]

(b) A *prompt ledger* based on ERC-7208 that records identity, lineage, judgments, validation, and reuse events with commit–reveal protection and micro-royalty routing; this enables reproducibility, attribution, and composable intelligence markets.

(c) An evaluation and safety model that uses LLM-as-judge as a calibration baseline and directs human effort via disagreement sampling from active learning, with stake-backed roles (authors, judges, validators) providing accountability [22–24].

(d) A token utility design that ties subscriptions, runtime, marketplace actions, data curation, and safety bonds to protocol fees, staking and slashing, and scheduled burns.

(e) A delivery roadmap with concrete acceptance tests and audit gates for moving from analysis to simulation to execution across 2025–2026.

2 Background and Related Work

2.1 Tool-Using and Multi-Agent LLMs

Reasoning-then-acting methods interleave textual thought with external tool calls. ReAct improves decision making in environments where factual grounding matters [2]. Toolformer demonstrates self-supervised API use, letting models select when and how to call tools [3]. Program-aided LLMs (PAL) shift exact computation to generated code, which reduces

[1]Product status is summarised in Section 3.1.

hallucination and increases determinism for maths and logic [8]. Multi-agent frameworks exploit specialized roles and structured communication to improve performance: agents may act as planners, critics, or tool specialists, exchanging messages in a conversation protocol that supports cooperation, debate, and division of labor as shown in AutoGen [4], CAMEL [5], and multi-agent debate frameworks [46].

In parallel, work on evaluation has explored LLMs as judges. Zheng et al. argue that models can provide useful comparative judgments when carefully prompted and calibrated, especially as a triage signal that focuses scarce human attention where disagreements are largest [22]. This idea informs our use of LLMs as a *baseline* in the prompt mining pipeline: models do not decide rewards but help highlight likely failure modes so humans can spend time where it matters.

2.2 Knowledge Graphs, RAG, and Graph Learning

Retrieval-augmented generation (RAG) injects evidence at inference time and consistently improves factuality in knowledge-intensive tasks [9]. For financial and on-chain data, pure text retrieval is insufficient: relationships matter. Knowledge graphs (KGs) provide a typed, link-rich substrate where entities (modelled with RDF triples and queried with SPARQL) [40, 41] and where (addresses, tokens, pools, proposals) and relations (transfers, swaps, votes) can be traversed and validated [10]. Graph learning methods such as GraphSAGE and GCNs offer embeddings for pattern discovery. In Zero, we blend symbolic and neural retrieval. Symbolic queries answer precise questions with provenance: "which approvals expose wallet W to protocol P and when do they expire?" Neural embeddings augment this with similarity and clustering: "which pools are structurally similar to this pool in terms of fee behaviour and volatility?" This hybrid allows agents to move naturally between fact lookup, structural reasoning, and statistical generalisation.

2.3 Execution Risks and Commit–Reveal

Open mempools expose pending transactions to adversaries who can reorder or sandwich to extract value (MEV) [13–15]. A classical countermeasure is commit–reveal: a user commits to a hash of data and reveals the data later, preventing frontrunners from copying the intent [16]. We apply commit–reveal in two places. **First**, prompt mining protects authors during the publication window, avoiding copy-and-paste plagiarism before reveal. **Second**, sensible commit, reveal parameters (bond size, reveal window, slashing rules) ensure that the scheme remains both secure against front-running and usable in practice [7]. **Third**, randomized judging and validator audits provide an additional layer of quality control and abuse resistance. The broader security posture borrows from safe RL and AI safety: avoid unbounded objectives that incentivise extreme behaviour; instrument agents with monitors and rate limits; and keep humans in the loop for high-impact actions [30, 31].

3 System Overview

3.1 Zero Chat and the Role of Zero Agents

Zero Chat is a live, public beta that exposes natural-language research, analytics, and reporting across more than fifteen blockchains, with many more in integration. It is powered by the modular *Zero Agent* reasoning and orchestration engine and supports wallet analytics, token and protocol intelligence, and chain-aware prompts designed to flow into simulation and guarded execution. The beta went live on 1 July 2025 at https://askzero.projectzero.io. The product is positioned as an integrated intelligence service rather than a narrow analytics tool. Users ask questions in plain language; agents map intent to plans; the system retrieves evidence from the knowledge graph and live feeds; and it returns narrative answers with charts and, as execution rolls out, human-readable transaction previews. The goal is not to replace wallets or explorers but to unify their core functions behind a single auditable interface that reduces cognitive load for non-experts and accelerates expert workflows.

The engine is designed for both B2C and B2B deployment. For individual users, it provides portfolio views, token and DeFi insights, governance updates, and strategy previews in natural language. For businesses and institutions, it provides an architecture that can be integrated into existing dashboards and workflows, offering secure connectors, API access, and enterprise features. Requests flow through an intent understanding routine that classifies the task, identifies entities, and selects a plan template (research, risk, simulation, execution). Plans are realized as conversation graphs among specialized agents. The knowledge graph serves structured state, while connectors provide price feeds, gas estimates, governance activity, and off-chain metadata. Each response carries a machine-readable trace linking back to sources so that judgments and downstream actions are auditable.

Observed traction has been strong. Since launch the product has onboarded thousands of users who have submitted several thousand prompts. The community footprint across X, Telegram, and Discord continues to grow, and chain-specific queries have been featured in public posts and demos. Coverage already includes BNB Chain, Arbitrum, Polygon, Avalanche, Scroll, and others, with additional networks in progress.

3.2 Zero Knowledge Graph and Semantic Store

Zero maintains an AI-ready knowledge graph that harmonises on-chain events, off-chain metadata, and derived analytics. Core entities include addresses, tokens, pools, protocols, proposals, and time-indexed states; relations capture transfers, approvals, swaps, votes, and governance edges. Two complementary modes are supported.

First, symbolic traversals answer exact questions such as "which approvals expose wallet W to protocol P?" or "what was the utilisation history of vault V?" These queries yield deterministic answers with full provenance, suitable for auditing, compliance, or risk reporting.

Second, neural retrieval uses learned embeddings to capture similarity in behaviour and structure. Agents exploit this to find comparable pools for benchmarking, to cluster addresses by activity patterns, or to propose priors for simulation tasks.

The graph is versioned and timestamped so that any result can be reproduced exactly by querying the appropriate snapshot. This property is essential for research credibility and for reward distribution in prompt mining, since outcomes must be traceable back to the prompts and the precise data context in which they were executed.

Finally, the graph emits an event stream suitable for monitoring and alerting. Agents can subscribe to wallets, pools, or proposals and receive structured change notifications. This capability underpins portfolio health checks, treasury exposure monitoring, and governance tracking for both individuals and institutions.

3.3 Evolving Federated Reasoning Agents (EFRA)

EFRA are specialised Zero Agents that coordinate through a federated conversation protocol. Roles include a researcher that curates hypotheses and a reading plan; an analyst that extracts features and fits simple models; a risk agent that evaluates exposures and stress scenarios; and an execution agent that proposes guarded actions subject to policy. Shared memory retains facts, intermediate computations, and critiques with timestamps so that reasoning is auditable.

We combine evolutionary reinforcement learning with federated orchestration to allow these agents to adapt strategies collaboratively while preserving role specialisation. The training loop respects safety: agents are never permitted to change policy limits or bypass preview gates. Where on-chain execution is involved, EFRA emits a transaction preview that enumerates contract calls, parameters, expected state deltas, and fee estimates. Users can accept, modify, or decline.

This preview-first pattern provides a consistent, explainable control point before any state change, ensuring that all autonomous reasoning is transparent and subject to human oversight when needed.

3.4 Developer Surface: SDK and Agent Studio

The platform will expose a full software development kit (SDK) that allows developers to compose agents using typed conversation graphs and to query the knowledge graph through a stable interface. This SDK is designed not only as a thin wrapper but as an expressive programming surface that enables the construction of role-specialised agents, orchestration flows, and integration with external data sources. Developers can use the SDK to specify conversation protocols, assign roles, configure memory policies, and enforce preview-first execution guards. The goal is to make the process of building domain-specific agents straightforward, while preserving the safety and auditability standards that underpin the wider Zero ecosystem.

Third-party data and execution providers integrate through a plugin architecture. Each plugin exposes a set of declared capabilities, input and output schemas, and rate limits. This ensures that external providers can contribute value without compromising the stability or safety of the platform. Data plugins may deliver specialised feeds, such as compliance checks, off-chain sentiment metrics, or industry-specific risk signals. Execution plugins may surface transaction routers, bridging services, or analytics pipelines, each bound by explicit rate-limiting and policy constraints. The plugin system is designed to be open but accountable: every plugin has an identifiable maintainer, a reputation profile, and token-denominated permission stakes to ensure responsible behaviour.

For users without programming experience, the system provides a no-code Agent Studio. This interface allows non-programmers to assemble agents from reusable templates, configure their behaviour, and deploy them into the registry. In practice, this means a user can select a template for a research agent, supply domain-specific instructions, attach plugins for data feeds or execution endpoints, and declare simple policies such as maximum transaction size or alert thresholds. Once published, these agents inherit the same guarantees as those built with the SDK: typed conversation graphs, memory management, preview-first execution, and audit-ready telemetry.

Permissions and throughput across the system are governed by token-backed agent keys. Each agent that operates within the network must present a valid key that defines its allowed rate of queries, access tier, and execution limits. These keys are obtained by staking tokens, which provides economic weight behind responsible usage. If an agent misbehaves, by exceeding declared limits, producing harmful outputs, or engaging in abuse, the corresponding key can be revoked, and the associated stake can be partially or fully slashed. This mechanism ensures that both developers and enterprises align their incentives with the health of the ecosystem.

The registry serves as the catalogue and discovery layer for the entire agent economy. Every agent published, whether through the SDK or Agent Studio, is registered with metadata including its description, capabilities, author, performance history, and reputation. Users and institutions can search the registry to find agents that meet their needs, compare performance metrics, and review usage histories. Reputation scores accumulate based on reliability, user ratings, and on-chain outcomes, allowing the most effective agents to gain visibility while discouraging low-quality or malicious contributions. The registry also enables attribution: if one agent builds upon another's work or reuses prompts from the prompt ledger, lineage is recorded and upstream contributors receive credit or royalties.

Together, the SDK, plugin framework, no-code studio, and registry create a complete developer and user infrastructure. Developers benefit from a structured programming surface and economic incentives to contribute high-quality components. Non-programmers can still participate by composing agents through intuitive interfaces. Enterprises gain integration points for advanced analytics and compliance features. And the community as a whole benefits from a transparent, discoverable, and accountable marketplace of agents, each governed by token-linked permissions and auditable execution guarantees. In this way, the Agent Studio and Developer Infrastructure transform the platform from a closed intelligence layer into a participatory ecosystem where innovation, governance, and accountability are shared across the network.

4 Prompt Mining: A Ledger for Intelligence

4.1 Why Prompts Need Identity and Telemetry

Prompts encode operational know-how: research plans, risk checklists, governance synthesis, and execution playbooks. Unlike code, which benefits from repositories, licences, and metrics, prompts today are ephemeral text. There is no canonical identifier to reference a prompt programmatically; no agreed way to share royalties; no lineage tracking to attribute derivatives; and no telemetry to measure performance. Provenance research shows that without verifiable trails, reproducibility and accountability suffer, since downstream outcomes cannot be reliably tied back to the inputs and assumptions that generated them.

A *prompt ledger* remedies this [45]. Each prompt receives a stable handle; events record when it was mined, where it was reused, how it performed, and how it evolved over time. With identity and telemetry in place, marketplaces, royalties, and curation mechanisms become straightforward to implement. Crucially, agents and dashboards can reference prompts programmatically, turning "intelligence" into a composable building block rather than a one-off chat interaction. This transforms prompts from transient strings of text into durable, auditable, and monetizable assets within the agent economy.

4.2 ERC-7208 On-Chain Data Containers

ERC-7208 defines interfaces that decouple data storage from logic, introducing *On-chain Data Containers* (ODCs) and standardised accessors [25–28]. A prompt is represented as a container with: (i) a salted content hash; (ii) minimal metadata (author, timestamp, domain tags, model family); (iii) pointers to encrypted full text stored off-chain (IPFS/Arweave); (iv) lineage links (parents/children); and (v) a policy describing reuse, royalties, and distribution. These elements together provide a robust and interoperable substrate for representing prompts as durable digital objects.

Two properties make ERC-7208 especially useful for prompts. First, the standard is *omnichain*: a container mined on one network can be referenced from another without losing provenance, which is essential for cross-chain agent workflows. Second, the commit–reveal flow prevents front-running: the author commits the salted hash, then reveals the content within a fixed window; failure to reveal forfeits the bond, deterring gamesmanship. The result is a compact, durable handle that agents, dashboards, and smart contracts can rely on for attribution, audit, and automated royalty routing, turning prompts into composable and monetizable primitives within the agent economy.

4.3 Mining, Judging, and Validation

Mining. Authors "mine" prompts by staking *PZERO* and submitting a commitment. After reveal, the container is finalised and an event with its ID is emitted. Agents and apps can call the container ID with input parameters; if the policy requires payment, micro-royalties are routed automatically.

Judging. Judges review revealed prompts across four criteria: usefulness, novelty, reproducibility, and safety. Each judge posts a bond; accuracy, measured against eventual consensus and validator audits, determines rewards. LLM-as-judge provides a calibration baseline and flags disagreements; human attention is directed using *disagreement sampling* from active learning [22–24].

Validation. Validators audit a sample of judgments and post higher stake, earning larger rewards for catching mistakes or collusion. Appeal rounds and stake-weighted penalties create meaningful downside for abuse. Because all events land on-chain, anyone can reconstruct a prompt's history, judge reliability, and validator performance. This MJV triad yields a robust, stake-backed scoring market with clear incentives for quality.

4.4 Turning Quality into Payable Signals

We compute a monthly *quality score* Q per container as a weighted blend of three signals:

$$Q = w_r\,\phi(\text{reuse}) + w_p\,\psi(\text{performance}) + w_j\,\gamma(\text{judgment}).$$

Here ϕ applies a square-root transform to unique caller counts after sybil-resistant deduplication; ψ normalises domain outcomes to $[0,1]$ (in trading: risk-adjusted PnL subject to drawdown caps and Sharpe-like ratios [44]; in governance: acceptance and impact; in research: citations, dwell time, and expert endorsement); and γ aggregates human scores with variance penalties and judge reliability weights. A gentle inequality control (e.g., a capped-log transform) prevents over-concentration... The Prompt Pool distributes rewards proportional to Q under transparent weights. If concentration rises too quickly, weights or caps can be adjusted by governance, with the adjustments themselves recorded on-chain for auditability.

4.5 Royalties, Derivatives, and Bundles

Stable container IDs make *micro-royalties* feasible. Each reuse can charge a tiny fee, which accumulates significantly for widely adopted intelligence. Derivatives declare parents, and a decaying split routes a small percentage of royalties upstream, keeping credit flowing without creating infinite tails. The mechanism is analogous to, but distinct from, NFT royalty signalling (EIP-2981) [29].

On top of individual prompts, curators can package *bundles*, for example collections such as "Top DeFi Backtests This Quarter" or "Best Governance Summaries for Layer 2s." Each bundle is itself represented as a container with its own metadata and policy, referencing the constituent prompts. Agents and dashboards can subscribe to bundles as feeds, making curated intelligence reusable at scale and creating an additional economic role for community members who organise and highlight the most effective work.

5 Token Utility and Economics

5.1 Active at TGE (Day 1)

At token generation event (TGE), *PZERO* is used to pay for Zero Chat subscriptions with tiered limits and discounts, metered agent runtime and execution previews, marketplace transactions in Agent Studio, developer staking for elevated permissions, premium datasets and analytics gating, and a scheduled protocol fee burn. These utilities link real product usage to recurring token demand and controlled supply reduction while avoiding equity-like claims. Jurisdiction-specific regulatory treatment targets recognition of *PZERO* as a utility token rather than a security. In the European Union, it is aligned with the MiCA framework as a utility crypto-asset; in the United Kingdom, it is positioned as an exchange token; in Singapore, it is classified as a digital payment token under the Payment Services Act; and in the United States, distribution is managed carefully with external legal opinions to limit exposure until clarity is established. This positioning ensures that the token's role remains focused on product access, network operations, and incentive alignment, rather than on speculative or profit-sharing attributes.

Utility	**What it enables**	**Primary users**
Subscriptions & feature access	Pay in *PZERO* for tiers; discounts and higher limits (streams, context, alerts)	Individuals, teams
Runtime & execution fees	Metered compute, tool calls, human-readable previews	Power users, protocols
Agent Studio marketplace	Trigger/subscribe/license third-party agents/plugins	Users, devs, enterprises
Developer staking	Stake for elevated limits and sensitive ops; slash on abuse	Agent builders, data/tool providers
Premium datasets & analytics	Access curated KG slices, risk scores, compliance checks	Funds, DAOs, enterprises
Protocol fee sink	Portion of protocol fees is burned on schedule	Network-wide
Governance bootstrap	Ratify fees, burn bands, listing standards	Token holders

5.2 Planned Utilities (Post-TGE, Phased)

Prompt mining begins with minting and usage telemetry, followed by judging and validation with stake-backed roles. Specialised DeFi agents charge execution fees with refundable safety bonds that are enforced by policy vaults. Agent permission keys are stake-locked to regulate throughput and sensitive operations. Data-oracle curation and usage mining reward accuracy and legitimate activity, creating incentives for high-quality feeds and sustained engagement. Cross-chain gas abstraction and enterprise entitlements are introduced as the platform scales, providing convenience for retail users and service guarantees for institutions.

These utilities diversify demand sources and put meaningful balances at stake, reducing circulating supply while strengthening accountability. In combination, they create a spend–stake–burn loop: product usage generates demand for *PZERO*, stakes tie participation to responsible behaviour, and fee burns or slashing events reduce outstanding supply. The result is a token economy linked directly to real network activity rather than speculative promises, ensuring that value accrues from usage, quality assurance, and long-term sustainability.

5.3 Economic Impact Map

Demand, burn, staking, and treasury flows together determine net supply and incentives. Subscriptions and runtime generate recurring spend. Marketplace activity routes protocol fees and creator revenues. Prompt mining introduces stake-gated roles with slashing for

Utility	Description	ETA
Prompt mining	ERC-7208 containers for prompts; stake to mint, judge, validate; rewards from pool	Q4 2025
DeFi agent execution	Specialised agents with policy vaults; fees and safety bonds	Q4 2025
Agent permission keys	Stake/lock for throughput and features; revocable	Q4 2025
Data curation & bounties	Stake to curate feeds; rewards for accuracy; slashing	Q4 2025
Usage mining credits	Credits for heavy legitimate usage from fees under caps	Q4 2025
Dev grants & incentives	Grants paid in *PZERO* with milestones	Ongoing
Cross-chain gas abstraction	"Gas-as-a-service" paid in *PZERO*	Q1 2026
Enterprise entitlements & SLAs	Reserve capacity, private connectors, retention	Q2 2026
Compliance & attestations	Risk checks, attestations, audit trails	Q2 2026

abuse. Execution adds refundable safety bonds. Data curation ties rewards to accuracy and reliability of feeds. A scheduled burn of a fraction of fees paid in *PZERO* converts network activity into structural scarcity. Governance tunes parameters such as stake minimums, burn ranges, and reward weights in order to maintain balance between growth incentives and supply discipline. By adjusting these levers in response to observed telemetry, the system sustains a spend–stake–burn loop that rewards legitimate participation, penalizes abuse, and ensures that long-term value creation is tied directly to product usage and ecosystem health.

Driver	Demand	Burn	Stake//	Treasury
Subscriptions & runtime	High, recurring	Optional %	—	Yes
Agent marketplace	Medium–high	Optional %	Dev perms	Yes
Prompt mining	Medium	Pool top-ups	Authors/judges/validators	Yes
DeFi agent exec.	Medium	Optional %	Safety bonds	Yes
Data curation	Medium	—	Curator stakes	Yes

6 Roadmap and Delivery Plan (2025–2026)

The roadmap sequences capability in a measured way: from research, to simulation, to guarded execution, and finally to an open developer ecosystem. Each phase has explicit objectives, scope, acceptance gates, and risk controls to ensure safety, correctness, and usability.

6.1 Phase 1: Foundational Intelligence Layer (Live in Beta)

Objective. Deliver the base version of Zero Chat with intelligent insights across a growing set of chains and data domains. Focus areas include ingestion stability, schema harmonisation, analytics validation, and user-facing features that demonstrate the value of the knowledge graph.

Scope.

- Wallet analytics: balances, transfers, approvals, exposures.
- Token and protocol exploration.
- NFT and DeFi dashboards (liquidity, yields, utilisation).
- Smart contract explorer with gas estimation.
- Market metrics, liquidity views, and treasury positions.
- On-chain analytics built with standard parsing techniques [42].
- Integration with fifteen-plus chains and off-chain APIs.
- Modular *Zero Agent* orchestrating intents into research flows.

Acceptance Criteria.

- Schema coverage for top assets and protocols.
- Concordance tests between independent data sources.
- Latency budgets met (e.g., dashboard refresh <5s for cached data).
- Human-reviewed correctness against a curated gold set.
- Reports include citations and provenance links.

Risk Controls.

- Reorg-aware ingestion pipelines.
- Differential testing between indexers.
- Versioned parsers with regression suites.

6.2 Phase 2: Simulation and Research Layer (15 Sep–15 Nov 2025)

Objective. Enable advanced simulation and research capabilities. Users can run *what-if* scenarios, backtests, and receive risk classifications with clear disclosures.

Scope.

- Simulation engine with scenario builders (staking, farming, swaps, bridging).
- Standardised callbacks for domain metrics:
  - TVL, fees, APY/APR.
  - Slippage, drawdown, Sharpe-like ratios [44].
  - Utilisation and stability indicators.
- Historical backtests and walk-forward validations.
- Multi-turn memory for context persistence across sessions.
- Monitoring and alerts for wallets, contracts, volatility thresholds.

- Governance intelligence: proposal summaries, voter cohorts, outcome sensitivity.

Acceptance Criteria.

- Reproducible backtests using fixed data snapshots.
- Parameterised scenarios with traceable inputs and outputs.
- Calibration studies versus public benchmarks.
- Risk disclosures included in all reports.
- User studies confirm reduced time-to-insight.

Risk Controls.

- Overfitting and regime-shift mitigation through stress tests.
- Walk-forward validation protocols.
- Conservative defaults in reports to avoid false precision.

6.3 Phase 3: Execution and Agent Autonomy (15 Nov–31 Dec 2025)

Objective. Add guarded on-chain execution through the chat interface and launch specialised autonomous agents (EFRA) under strict policy controls.

Scope.

- *Preview-first execution*: all actions summarised with parameters, expected state deltas, gas/fee estimates, and human-readable diffs.
- Agents include:
  - DeFi yield optimisation with risk rebalancing.
  - Trading agents for arbitrage, price triggers, hedging.
  - Governance agents for tracking proposals and coordinated voting.
- Policies constrain actions (allowlists, slippage caps, position limits).
- EFRA (Evolving Federated Reasoning Agents) integrate with the knowledge graph for traceable decision-making.

Acceptance Criteria.

- External audit sign-off for core contracts and execution layer.
- Demonstrated preview-to-execution concordance.
- Dry-run audits of transactions with no critical policy breaches.
- At least one EFRA cohort in controlled domains with stable performance over 30 days.

Risk Controls.

- Commit–reveal patterns for sensitive intents [13, 15].
- Permissioned agent keys with slashing for abuse.
- Safe default parameters in execution previews.

6.4 Phase 4: Agent Studio and Developer Infrastructure (parallel from mid-Nov 2025)

Objective. Open the Zero Agent infrastructure to developers, enterprises, and protocols. Provide SDKs, no-code studio, APIs, and a marketplace.

Scope.

- SDK and public APIs for agent composition.
- No-code Agent Studio for assembling templates (research, risk, strategy).
- Plugin system for third-party data and execution integrations.
- Discovery registry with search, provenance, and reputation.
- Token-linked permissions, staking, and monetisation.
- Marketplace denominated in $PZERO$ with programmable splits for creators and treasury.

Acceptance Criteria.

- At least twelve external agents built by early partners.
- Registry search resolves by skills, provenance, and performance history.
- Marketplace processes transactions with transparent revenue shares.

Risk Controls.

- Sandbox isolation for third-party plugins.
- Permissioned scopes and rate limits on sensitive APIs.
- Curation and reputation filters for registry listings.

7 Safety, Compliance, and Governance

7.1 Guarded Execution and Policy Vaults

We adopt preview-first execution with human-readable diffs and spend caps. Sensitive actions may use commit–reveal to minimise mempool leakage. Policy vaults enforce constraints on agent actions (position limits, whitelists, time locks). Safety design follows guidance from AI safety and safe RL: avoid unbounded objectives; prefer shielded policies; and keep humans in the loop for high-impact decisions [30, 31].

7.2 Provenance, Privacy, and Attestations

Prompt containers carry hashes and lineage; off-chain content is encrypted and signed, with pointers to IPFS/Arweave [32, 33]. Where third-party identity is needed, we rely on W3C Verifiable Credentials and DIDs for portable attestations [34, 35]. The semantic task log links evidence, judgments, and outcomes for auditable trails consistent with provenance literature.

7.3 Governance Dials

A small set of *dials* controls behaviour: minimum stakes and lock periods for authors/judges/validators; fee shares for the Prompt Pool and the burn; weights in the quality score; and decay for lineage payouts. Early values are modest to reduce barriers (small-dollar stakes, 7–30 day locks). Parameters are upgraded via community governance once audits complete, drawing on mechanisms such as quadratic funding where suitable [36, 37].

8 Evaluation and Telemetry

We measure: (i) provenance coverage, defined as the share of agent runs that reference ERC-7208 containers; (ii) the distribution of reuse and inequality metrics, which track whether adoption is concentrated in a few prompts or broadly spread; (iii) outcome uplift versus baselines when prompts are used, to determine whether the system materially improves accuracy, efficiency, or returns; (iv) time-to-finality, measured from mining to judging to validation; (v) stake at risk, including locked balances and the observed slashing rate; (vi) fee routing, burns executed, royalty flows, and token velocity; and (vii) judging quality, expressed as false positives and false negatives broken down by domain.

Dashboards will expose these KPIs publicly so that parameters can be tuned in response to evidence rather than intuition. By making telemetry transparent, the system encourages accountability, supports research into agent behaviour, and provides regulators and stakeholders with an auditable trail of activity. Over time, these metrics will guide adjustments to staking thresholds, burn ratios, reward weights, and governance policies, ensuring that the economic and operational health of the network is continually reinforced by observable outcomes.

9 Conclusion

Zero Agents and Zero Chat make Web3 intelligence addressable, auditable, and executable. By combining a cross-chain knowledge graph, multi-agent research and simulation, guarded execution, and an ERC-7208-based prompt ledger, the system turns ephemeral prompts into composable digital assets and aligns token economics with real usage. The roadmap sequences capability safely from analysis to simulation to action, ensuring that each layer is validated before the next is introduced.

We invite collaborators to build agents, mine prompts, and stress-test the governance dials that shape this emerging economy. The goal is not only to deliver a useful product in the near term but also to cultivate a durable ecosystem where intelligence is treated as an asset, contributors are rewarded transparently, and the parameters that drive incentives are continuously refined by evidence. In this way, Zero establishes a foundation for a trustworthy, participatory, and sustainable agent economy that bridges individual users, developers, and institutions across the Web3 landscape.

References

References

[1] Author(s), "Title of the Autogen Paper," Journal/Conference, Year.

[2] Shunyu Yao, et al. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629.

[3] Timo Schick, et al. (2023). Toolformer: Language Models Can Teach Themselves to Use Tools. arXiv:2302.04761.

[4] Yuhui Qin, et al. (2023). AutoGen: Enabling Next-Generation Large Language Model Applications via Multi-Agent Conversation. arXiv:2308.08155.

[5] Guohao Li, et al. (2023). CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society. arXiv:2303.17760.

[6] Guanzhi Wang, et al. (2023). Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv:2305.16291.

[7] A. Canidio (2023). Commit–Reveal Schemes Against Front-Running Attacks. Tokenomics 2022 (Dagstuhl OASIcs). https://drops.dagstuhl.de/opus/volltexte/2023/17133/.

[8] Luyu Gao, et al. (2023). PAL: Program-Aided Language Models. arXiv:2211.10435.

[9] Patrick Lewis, et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP. arXiv:2005.11401.

[10] Aidan Hogan, et al. (2021). Knowledge Graphs. ACM Computing Surveys 54(4).

[11] William L. Hamilton, et al. (2017). Inductive Representation Learning on Large Graphs (GraphSAGE). NeurIPS.

[12] Thomas N. Kipf and Max Welling (2016). Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907.

[13] Philip Daian, et al. (2019). Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges. arXiv:1904.05234.

[14] Raphael Auer (2022). Extractable Value and Market Manipulation in Crypto and DeFi. BIS Bulletin No. 58.

[15] Liyi Zhou, et al. (2021). Mitigating Frontrunning, Transaction Reordering and Consensus Instability. arXiv:2106.07371.

[16] Moni Naor (1991). Bit Commitment Using Pseudo-Randomness. CRYPTO.

[17] Hui (Frank) Chen, et al. (2020). A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses. ACM Computing Surveys 53(3).

[18] Reuven Y. Rubinstein (1999). The Cross-Entropy Method for Combinatorial and Continuous Optimization. Methodology and Computing in Applied Probability.

[19] Tim Salimans, et al. (2017). Evolution Strategies as a Scalable Alternative to Reinforcement Learning. arXiv:1703.03864.

[20] Santosh Khadka and Kagan Tumer (2018). Evolutionary Reinforcement Learning. arXiv:1805.07917.

[21] Max Jaderberg, et al. (2017). Population Based Training of Neural Networks. arXiv:1711.09846.

[22] Lucy Zheng, et al. (2023). LLM-as-a-Judge: Evaluating LLMs with LLMs. arXiv:2306.05685.

[23] Burr Settles (2009). Active Learning Literature Survey. UW-Madison Tech Report.

[24] Steve Hanneke (2014). Theory of Active Learning. Foundations and Trends in Machine Learning.

[25] Ethereum Improvement Proposal ERC-7208 (2023). On-Chain Data Containers. https://eips.ethereum.org/EIPS/eip-7208.

[26] Ethereum Magicians (2023–2024). Discussion: ERC-7208 On-Chain Data Container. https://ethereum-magicians.org/t/erc-7208-on-chain-data-container/14778.

[27] Pooja Ranjan (2024). EIPs Insight (January 2024). https://hackmd.io/@poojaranjan/EIPsInsightJanuary2024.

[28] Nexera Standard Docs (2024). Overview of ERC-7208. https://docs.nexera.network/nexera-standard/overview.

[29] Ethereum Improvement Proposal EIP-2981 (2021). NFT Royalty Standard. https://eips.ethereum.org/EIPS/eip-2981.

[30] Dario Amodei, et al. (2016). Concrete Problems in AI Safety. arXiv:1606.06565.

[31] Javier García and Fernando Fernández (2015). A Comprehensive Survey on Safe Reinforcement Learning. Journal of Machine Learning Research.

[32] Juan Benet (2014). IPFS - Content Addressed, Versioned, P2P File System. arXiv:1407.3561.

[33] Sam Williams (2018). Arweave: A Protocol for Economically Sustainable Information Permanence. Whitepaper.

[34] W3C (2024). Verifiable Credentials Data Model v2.0 (Recommendation). https://www.w3.org/TR/vc-data-model-2.0/.

[35] W3C (2022). Decentralized Identifiers (DIDs) v1.0 (Recommendation). https://www.w3.org/TR/did-core/.

[36] Vitalik Buterin, Zoë Hitzig, E. Glen Weyl (2018). Liberal Radicalism: A Flexible Design for Philanthropic Matching Funds. arXiv:1809.06421.

[37] Steven P. Lalley and E. Glen Weyl (2014). Quadratic Voting. arXiv:1409.0264.

[38] Project Zero (2025). *Project Zero DD Questionnaire Template* (internal document, July–August 2025).

[39] Gavin Wood (2014). Ethereum: A Secure Decentralised Generalised Transaction Ledger (Yellow Paper).

[40] W3C (2014). RDF 1.1 Concepts and Abstract Syntax (Recommendation).

[41] W3C (2013). SPARQL 1.1 Query Language (Recommendation).

[42] Harry Kalodner, et al. (2017). BlockSci: Design and Applications of a Blockchain Analysis Platform. USENIX.

[43] Harry Markowitz (1952). Portfolio Selection. Journal of Finance.

[44] William F. Sharpe (1966). Mutual Fund Performance. Journal of Business.

[45] Rachid Ajaja (2025). Prompt Mining and ERC-7208 (public thread). https://x.com/AjajaRachid/status/1954529512785322416.

[46] Yilun Du, Shunyu Yao, et al. (2023). Improving Factuality and Reasoning in Language Models through Multiagent Debate. arXiv:2305.14325.

[47] Gul A. Agha (1986). *Actors: A Model of Concurrent Computation in Distributed Systems*. MIT Press.

[48] Carl Hewitt (1973). *A Universal Modular Actor Formalism*. IJCAI.

[49] W.M.P. van der Aalst et al. (2003). *Workflow Patterns*. Distributed and Parallel Databases 14(1):5–51.

[50] Patrick Lewis et al. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP*. NeurIPS 2020.

[51] Vladimir Karpukhin et al. (2020). *Dense Passage Retrieval for Open-Domain QA*. EMNLP 2020.

[52] Shaoxiong Ji et al. (2022). *A Survey on Knowledge Graphs: Representation, Acquisition and Applications*. IEEE TKDE 34(5):249–270.

[53] Maximilian Nickel et al. (2016). *A Review of Relational Machine Learning for Knowledge Graphs*. Proc. IEEE 104(1):11–33.

[54] John Schulman et al. (2017). *Proximal Policy Optimization Algorithms*. arXiv:1707.06347.

[55] Tuomas Haarnoja et al. (2018). *Soft Actor-Critic*. ICML 2018.

[56] Miroslav Dudík et al. (2011). *Doubly Robust Policy Evaluation and Optimization*. Stat. Sci. 29(4):485–511.

[57] Nan Jiang, Lihong Li (2016). *Doubly Robust Off-policy Evaluation.* ICML 2016.

[58] Philip Thomas et al. (2015). *High Confidence Off-Policy Evaluation.* AAAI 2015.

[59] Tim Salimans et al. (2017). *Evolution Strategies as a Scalable Alternative to Reinforcement Learning.* arXiv:1703.03864.

[60] Max Jaderberg et al. (2017). *Population Based Training of Neural Networks.* arXiv:1711.09846.

[61] K. Deb et al. (2002). *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II.* IEEE Trans. Evol. Comput.

[62] Rishabh Agarwal et al. (2021). *Deep RL at the Edge of the Statistical Precipice.* NeurIPS 2021.

[63] Lloyd S. Shapley (1953). *A Value for n-Person Games.* Contributions to the Theory of Games II.

[64] Roger B. Myerson (1981). *Optimal Auction Design.* Math. of Operations Research 6(1).

[65] Philip Daian et al. (2020). *Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges.* IEEE S&P Workshops.