ZeruAI: Trust Oracles for Identity-Optional Economies

Behavior intelligence infrastructure for capital allocation, incentives, credit, compliance, and agentic economies

ZeruAI · Whitepaper v0.2

Abstract

Onchain economies are identity-optional: a wallet can trade, borrow, bridge, provide liquidity, claim incentives, and participate in governance without revealing who controls it. Because the identity-linked records that traditional finance uses to allocate capital, credit, and access are unavailable by default, protocols fall back on shallow signals such as wallet counts, transaction counts, and raw volume, which cannot distinguish durable contributors from sybil clusters, mercenary liquidity, or extractive farmers. The result is systematically inefficient capital allocation: crypto does not have a capital shortage; it has a trust shortage. This paper introduces ZeruAI, a trust-oracle infrastructure that converts longitudinal wallet behavior into decision-grade trust, risk, and contribution signals. Its core primitive, zScore, maps wallet activity across DeFi, trading, liquidity, lending, staking, governance, and incentive participation into a behavioral trust signal, computed by a layered architecture that combines a universal wallet-behavior model with domain-specific models for incentives, credit, screening, and campaign allocation. We formalize the trust-allocation problem, describe the data-ingestion, feature-engineering, and modeling pipeline, and present the product surface built on the primitive: the Zaps capital-allocation application; the zDrop, zLend, and zScreen API lines; and the $ZERU coordination token. We report deployment evidence (over 320 million wallets scored, more than 750 thousand API calls, live explorer integrations, and paying protocol customers) and discuss limitations, adversarial risks, and responsible-use constraints for behavioral trust infrastructure.

Contents

1 Introduction

In traditional financial systems, identity is the default trust primitive. Banks, credit bureaus, insurers, payment networks, marketplaces, and trade-finance providers rely on identity-linked records such as accounts, credit histories, repayment records, and compliance files to decide who receives capital, credit, rewards, access, limits, or risk clearance. Onchain economies operate differently. A wallet can trade, borrow, bridge, provide liquidity, claim incentives, participate in governance, interact with protocols, receive stablecoins, join launches, and move capital across ecosystems without revealing who controls it. Identity is optional, and this optionality creates a capital-allocation problem that identity-based infrastructure cannot solve: *how should capital be allocated when identity is optional?*

The scale of the problem is already substantial. Crypto protocols allocate capital continuously through airdrops, token-generation events, liquidity mining, trading incentives, ecosystem grants, app-usage campaigns, public launches, points programs, referral campaigns, yields, and community rewards. As stablecoins, tokenized assets, real-world assets, trade finance, and agentic payments move onchain, the volume of capital allocated to pseudonymous counterparties grows, and with it the cost of allocating that capital poorly.

This paper presents ZeruAI, a trust-oracle infrastructure for identity-optional economies. The central thesis is that where identity is unavailable, trust must be derived from behavior: ZeruAI converts longitudinal wallet behavior into trust, risk, and contribution signals that protocols can act on. The core primitive, zScore, is a behavioral trust oracle for wallets. It maps wallet activity across DeFi, trading, liquidity, lending, staking, governance, incentives, and other onchain behaviors into decision-grade intelligence.

On top of this primitive, ZeruAI operates several product lines, summarized in Table 1: Zaps, a network application for capital-allocation campaigns; three API stacks (zDrop for airdrop and incentive allocation, zLend for behavioral credit and underwriting, and zScreen for wallet screening and compliance-adjacent workflows); and $ZERU, the ecosystem coordination token. ZeruAI's objective is to become the trust-oracle infrastructure layer for wallet-based economies, tokenized capital markets, and agentic economic systems.

Table 1: The ZeruAI product stack.

Layer	Product	Description
Core primitive	zScore	Behavioral trust oracle for wallets
Network application	Zaps	Capital allocation app for campaigns, incentives, launches, liquidity, trading, and protocol growth
API stack	zDrop APIs	Airdrop, incentive, reward, and sybil-aware allocation APIs
API stack	zLend APIs	Behavioral credit, underwriting, borrower segmentation, and yield-access APIs
API stack	zScreen APIs	Wallet screening, AML-support, counterparty risk, and compliance-adjacent APIs
Ecosystem token	$ZERU	Coordination token for access, participation, staking, rewards, and network alignment

The remainder of the paper proceeds as follows. Section 2 characterizes the trust deficit in identity-optional economies and the macro forces that make it urgent. Section 3 defines the trust-oracle category and formalizes the allocation problem. Sections 4–6 describe the system: the infrastructure stack, behavioral feature engineering, and the modeling architecture. Section 7 defines zScore. Section 8 reports the current integration surface. Sections 9–11 present the product layer: Zaps and the API lines. Section 12 situates ZeruAI in the market landscape. Sections 13–14 cover token economics and the business model. Sections 15–16 report traction and the roadmap. Section 17 discusses limitations and responsible use,

and Section 18 concludes.

2 Background: The Trust Deficit in Identity-Optional Economies

2.1 Trust primitives in existing economies

Every economy needs trust primitives. Traditional finance builds trust through identity-anchored infrastructure: KYC and KYB processes, bank accounts, personal and business credit histories, repayment and transaction records, compliance databases, legal entities, and regulatory frameworks. Internet platforms build trust through a parallel set of mechanisms: ratings, reviews, identity-linked accounts, loyalty systems, purchase histories, fraud detection, and platform-level reputation. In both cases, the trust signal ultimately resolves to a persistent, accountable identity.

Onchain economies break this assumption because identity is not always the primitive. A single wallet address may represent a genuine user, a sophisticated trader, a durable liquidity provider, a protocol contributor, a real borrower, a professional incentive farmer, a sybil cluster, a bot, a risky counterparty, or an AI agent, and nothing in the address itself distinguishes these cases.

2.2 The failure of shallow signals

Because identity is optional, protocols allocating capital fall back on signals that are observable but shallow: wallet counts, transaction counts, quest completion, raw trading volume, wallet balances, posted collateral, social attention, referral counts, and static allowlists. These signals are useful but incomplete. They do not reliably answer the questions that actually determine allocation quality: which wallets are durable, which users will retain, which liquidity will remain after incentives end, which wallets are sybil-like, which traders create useful activity, which launch participants are likely to dump, which borrowers deserve underwriting review, which counterparties require enhanced screening, and which communities create economic value.

The consequence is inefficient capital allocation across the industry. Crypto does not have a capital shortage; it has a trust shortage. Figure 1 depicts this gap: identity-optional wallets generate rich onchain behavior, but because that raw activity is hard to interpret, allocation decisions are poor, producing sybil farming, mercenary liquidity, reward dumping, and weak credit. A trust-oracle layer closes the gap by converting the same behavioral record into trust signals that support better incentives, access, credit, compliance, and capital allocation.

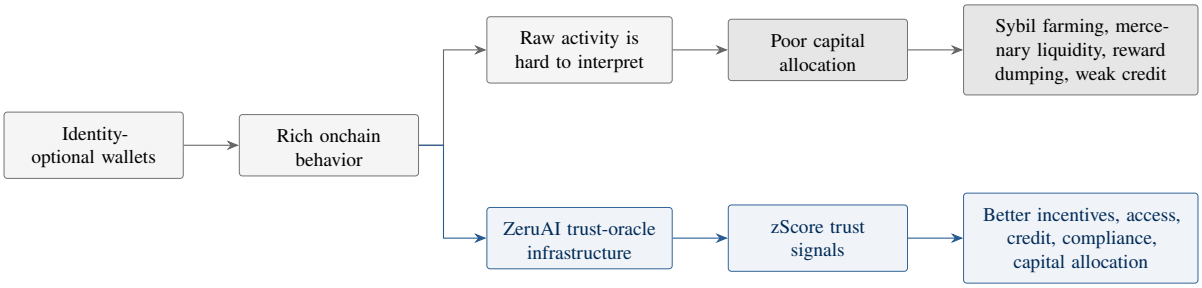

Figure 1: The identity-optional trust gap. Without interpretation, rich behavioral data still yields poor allocation (top path); a trust-oracle layer converts the same data into allocation-grade signals (bottom path).

2.3 Why now: capital is moving onchain

The onchain economy is expanding beyond crypto-native trading. The next phase includes stablecoin payments, tokenized treasuries, tokenized funds, tokenized private credit, tokenized invoices, tokenized trade finance, real-world-asset trading, DeFi credit, agentic payments, and machine-to-machine commerce. As more real capital moves onchain, the trust problem becomes more serious, because each allo-

cation channel has a distinct failure mode. When airdrops are poorly allocated, protocols waste growth budgets. When liquidity-mining rewards are poorly allocated, capital leaves after incentives end. When credit is poorly allocated, lenders face default risk. When compliance screening is weak, payment companies and virtual-asset service providers face counterparty risk. And as tokenized invoices, stablecoin payments, and trade finance move onchain, wallet behavior becomes a critical signal for underwriting, risk, access, and capital efficiency.

ZeruAI's immediate wedge is crypto-native incentive and capital allocation, where budgets are already deployed and the cost of misallocation is directly measurable. Its long-term market is trust infrastructure for tokenized capital markets.

3 Trust Oracles: Category Definition and Problem Formulation

3.1 Definition

A trust oracle is infrastructure that transforms behavioral, transactional, contextual, and risk data into decision-grade trust signals. The analogy to price oracles is direct. A price oracle answers the question, "what is the price of an asset?" A trust oracle answers a broader and more contextual question: "how should this wallet, user, counterparty, curator, protocol, or agent be trusted in a specific economic context?" In identity-optional economies, trust cannot depend only on static identity; it must be derived from behavior. ZeruAI accordingly defines *behavioral trust* as the inferred reliability, quality, contribution, risk, and economic alignment of a wallet or agent based on observed activity over time.

3.2 Formal problem statement

The trust-allocation problem can be stated formally. Let $\mathcal{W} = \{1, \dots, N\}$ denote the set of wallets (or agents) observed by the system. Each wallet i is associated with a longitudinal event history

$$\mathcal{H}_i(t) \;=\; \big\{(e_k, \tau_k) : \tau_k \le t\big\}, \tag{1}$$

where each event e_k is a decoded onchain action (a swap, transfer, liquidity deposit or withdrawal, borrow, repayment, liquidation, stake, bridge, governance vote, or reward claim) and τ_k its timestamp. A feature map ϕ aggregates the history into a behavioral feature vector

$$\mathbf{x}_i(t) \;=\; \phi\big(\mathcal{H}_i(t)\big) \;\in\; \mathbb{R}^d, \tag{2}$$

whose coordinates correspond to the feature families of Section 5: capital, liquidity, trading, credit, incentive, counterparty, governance, and protocol-specific behavior.

Behavioral trust is a mapping from features to a bounded score. The universal trust signal is

$$s_i(t) \;=\; f_\theta\big(\mathbf{x}_i(t)\big) \;\in\; [s_{\min},\, s_{\max}], \tag{3}$$

where f_θ is a learned model with parameters θ. Because behavior must be interpreted in context (Section 5), the system also produces domain-specific signals

$$s_i^{(c)}(t) \;=\; g_c\big(\mathbf{x}_i(t),\, s_i(t)\big) \qquad \text{for each decision context } c \in C, \tag{4}$$

where C ranges over incentives, liquidity, trading, launches, credit, screening, and curation.

An allocator (a protocol running a campaign, a lender, or a launch platform) holds a budget R and must choose an allocation $\mathbf{a} = (a_1, \dots, a_N)$ with $a_i \ge 0$ and $\sum_i a_i \le R$. Writing v_i for the realized economic value of allocating to wallet i (retention, durable liquidity, repayment, genuine usage), the allocator's objective is

$$\max_{\mathbf{a}} \sum_{i=1}^{N} \mathbb{E}\big[\, v_i \mid \mathcal{H}_i(t) \,\big]\, \mathbf{1}\{a_i > 0\} \qquad \text{subject to} \qquad \sum_{i=1}^{N} a_i \le R. \tag{5}$$

Shallow signals estimate $\mathbb{E}[v_i \mid \mathcal{H}_i]$ poorly: raw volume and wallet counts are weakly correlated with realized value and are cheaply manufactured by sybils. A trust oracle improves allocation precisely by supplying a better estimator; allocation rules become functions of the trust signal, $a_i = \alpha\big(s_i^{(c)}\big)$, for a monotone rule α chosen by the allocator.

3.3 Decision surface

This formulation covers the full range of allocation decisions that trust oracles support, summarized in Table 2. In each row, the oracle supplies the context-specific estimate that the decision requires. The question ZeruAI helps ecosystems answer, across all of these surfaces, is: *who deserves capital, access, credit, rewards, or risk limits?*

Table 2: Decision areas supported by trust oracles.

Decision area	**Trust-oracle question**
Incentives	Who deserves rewards?
Airdrops	Which wallets are genuine contributors?
Liquidity	Which LPs are durable?
Trading	Which activity is organic and high quality?
Launches	Which participants deserve access?
Credit	Which wallets deserve underwriting review?
Payments	Which counterparties are lower risk?
Compliance	Which wallets require enhanced screening?
Agent economies	Which agents can be trusted with limits, tasks, or capital?

4 System Architecture

4.1 Infrastructure stack

ZeruAI converts wallet behavior into trust-oracle outputs through a multi-layer infrastructure stack, shown in Figure 2 and enumerated in Table 3. Raw multichain activity enters through a data-ingestion layer that collects wallet and protocol activity across supported EVM ecosystems. An event-decoding layer converts raw transactions into protocol-specific actions, and a normalization layer standardizes activity across chains, contracts, assets, and protocols. The normalized record feeds a behavioral feature store, from which a universal wallet-behavior layer and a set of domain-specific models (Section 6) infer trust, contribution, risk, and quality. A signal-generation layer produces zScore and domain-specific trust signals, which are delivered through Zaps, APIs, dashboards, explorers, and integrations. Finally, an outcome-feedback layer learns from post-allocation outcomes to improve future decisioning.

The output of this stack is deliberately not a single score. It is a behavior-intelligence layer that can support incentives, distribution, credit, compliance, and capital allocation, with each surface consuming the signal appropriate to its decision context.

Table 3: Infrastructure layers and their functions.

Layer	Function
Data ingestion	Collect wallet and protocol activity across supported EVM ecosystems
Event decoding	Convert raw transactions into protocol-specific actions
Normalization	Standardize activity across chains, contracts, assets, and protocols
Feature engineering	Extract behavioral features from wallet histories
Model inference	Use AI/ML models to infer trust, contribution, risk, and quality
Signal generation	Produce zScore and domain-specific trust signals
Product delivery	Expose signals through Zaps, APIs, dashboards, explorers, and integrations
Outcome feedback	Learn from post-allocation outcomes to improve future decisioning

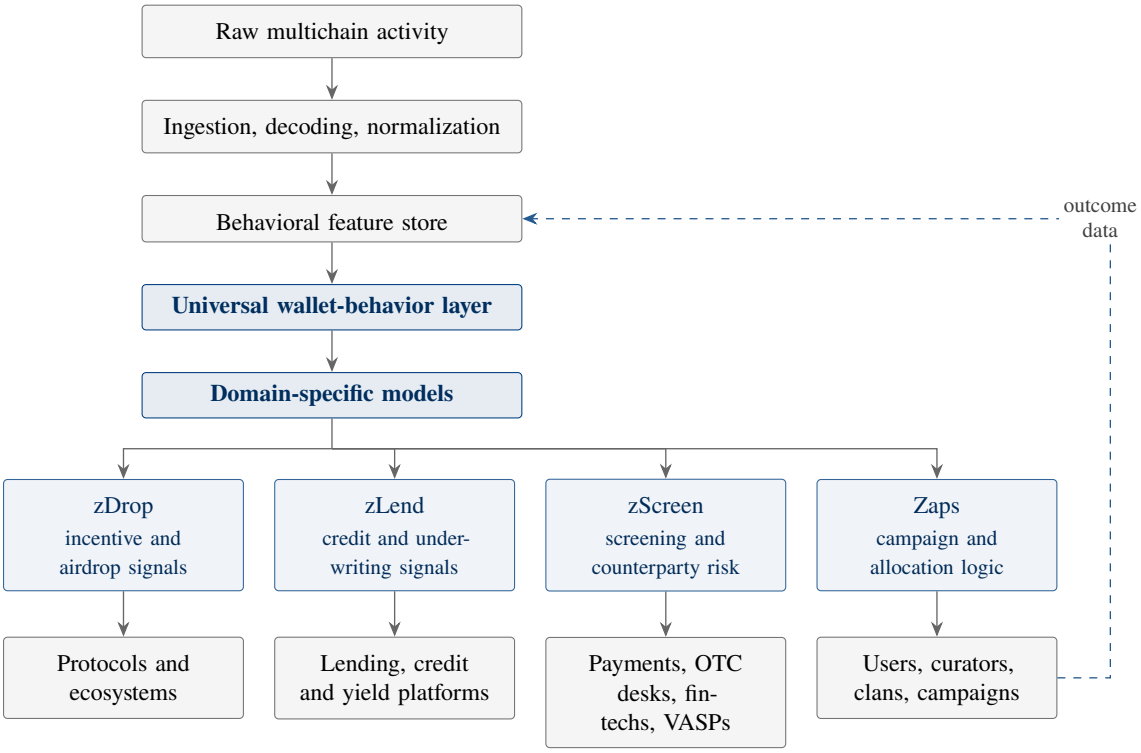

Figure 2: ZeruAI technical architecture. Raw activity is decoded, normalized, and aggregated into a feature store; a universal behavior layer feeds domain-specific models whose signals serve four consumer groups. Outcome data (dashed) feeds back into the feature store.

4.2 Data ingestion and behavioral interpretation

Raw blockchain data is noisy, and the same transaction can mean different things depending on context. A swap may represent disciplined trading, wash-like behavior, or basic portfolio rebalancing. An LP deposit may represent durable liquidity or short-term incentive farming. A bridge transaction may represent ecosystem adoption or sybil wallet rotation. A reward claim may represent earned contribution or extractive farming. A large wallet balance may represent capital strength or temporary custody. Any system that scores wallets directly on raw event counts inherits this ambiguity.

ZeruAI therefore begins with behavioral interpretation rather than raw aggregation. The system analyzes wallet activity across a broad set of categories: swaps, transfers, liquidity provision and LP withdrawals, lending, borrowing, repayments, liquidations, staking, bridging, governance, reward claims,

token retention, contract-interaction history, counterparty exposure, protocol diversity, and campaign participation. It then converts this activity into normalized behavioral features. In the notation of Section 3.2, this is the construction of the event history $\mathcal{H}_i(t)$ of Equation (1) and its aggregation into the feature vector $\mathbf{x}_i(t)$ of Equation (2).

5 Behavioral Feature Engineering

The feature map ϕ of Equation (2) draws on several families of behavioral features, summarized in Table 4. Capital-behavior features capture wallet age, capital deployed and retained, balance volatility, and capital rotation. Liquidity features capture LP duration, pool participation, withdrawal timing, incentive sensitivity, and retained TVL. Trading features capture volume quality, frequency, holding time, drawdown, asset diversity, and wash-like indicators. Credit features capture borrowing and repayment history, collateral management, liquidations, and leverage discipline. Incentive features capture airdrop participation, reward-claim timing, post-reward selling, and quest-farming patterns. Counterparty features capture shared funding sources, risky contract exposure, bridge behavior, and wallet clustering. Governance and community features capture voting, delegation, protocol loyalty, curator activity, and clan participation. Protocol-specific features capture app usage, feature adoption, contribution depth, retention, and repeat engagement.

Table 4: Behavioral feature families.

Feature family	**Examples**
Capital behavior	wallet age, capital deployed, capital retained, balance volatility, capital rotation
Liquidity behavior	LP duration, pool participation, withdrawal timing, incentive sensitivity, retained TVL
Trading behavior	volume quality, frequency, holding time, drawdown, asset diversity, wash-like indicators
Credit behavior	borrowing history, repayment history, collateral management, liquidations, leverage discipline
Incentive behavior	airdrop participation, reward-claim timing, post-reward selling, quest farming patterns
Counterparty behavior	shared funding sources, risky contract exposure, bridge behavior, wallet clustering
Governance and community	voting, delegation, protocol loyalty, curator activity, clan participation
Protocol-specific behavior	app usage, feature adoption, contribution depth, retention, repeat engagement

The key design principle governing how these features are consumed is that *behavior must be interpreted in context*. There should not be one generic score for every decision. A wallet that is a strong trader may not be a good borrower. A wallet that provides durable liquidity may not be a good public-launch participant. A curator that drives attention may not drive high-quality capital. A compliance-screening signal is not the same as a credit-readiness signal. Formally, the same feature vector $\mathbf{x}_i$ can map to different context scores $s_i^{(c)}$ under the domain functions g_c of Equation (4); the ordering of wallets under one context need not be preserved under another.

ZeruAI therefore combines a universal wallet-behavior layer, which captures context-independent structure, with domain-specific model outputs that adapt the interpretation to each decision context. The next section describes this two-layer modeling architecture.

6 Modeling Architecture

6.1 Research foundations

ZeruAI's research introduces zScore as a wallet-behavior-derived reputation primitive using AI neural-network models, with real-world credentials ported onchain through zkTLS [1]. Its DeFi-specific research further demonstrates that wallet scoring should be role-specific: in Uniswap-style environments, liquidity provision and swap behavior require different scoring logic [2]. Liquidity-provision scores and swap-behavior scores should reflect different behavioral patterns: volume, frequency, holding time, withdrawal patterns, pool context, TVL, and fee-tier dynamics. A third line of research extends this lineage from behavioral reputation to reward attribution, where direct financial incentives make adversarial robustness a first-order design requirement [3]; Section 6.6 summarizes it. These findings directly inform the production architecture, which separates a universal behavioral layer from role- and domain-specific heads.

6.2 Universal wallet-behavior layer

The universal layer captures broad wallet behavior across chains and protocols, corresponding to the model f_θ of Equation (3). It supports general inference around consistency, activity quality, protocol diversity, capital behavior, reward behavior, contribution quality, and risk patterns. These are the properties of a wallet that remain informative regardless of the downstream decision.

6.3 Domain-specific model layer

Domain-specific models, the functions g_c of Equation (4), adapt behavioral intelligence to specific decision contexts. Table 5 lists the domains and their outputs: incentive models produce reward eligibility, sybil risk, and contribution quality; liquidity models produce LP durability, capital stickiness, and incentive sensitivity; trading models produce quality-adjusted activity, organic volume, and trader reliability; launch models produce allocation eligibility, dumping risk, and ecosystem alignment; credit models produce borrower quality, repayment likelihood, and credit-readiness signals; screening models produce wallet risk, counterparty flags, and enhanced-due-diligence triggers; and curator models produce economic impact, quality of mobilized users, and retention impact.

Table 5: Domain-specific model outputs.

Domain	Model output
Incentives	reward eligibility, sybil risk, contribution quality
Liquidity	LP durability, capital stickiness, incentive sensitivity
Trading	quality-adjusted activity, organic volume, trader reliability
Launches	allocation eligibility, dumping risk, ecosystem alignment
Credit	borrower quality, repayment likelihood, credit-readiness signal
Screening	wallet risk, counterparty flags, enhanced due-diligence triggers
Curators	economic impact, quality of mobilized users, retention impact

6.4 Workflow-specific decision layer

Above the models sits a workflow layer that converts model outputs into product decisions: campaign leaderboards, allocation rankings, eligibility lists, risk flags, reward tiers, access decisions, underwriting signals, screening reports, and post-campaign analytics. This is the layer at which the abstract allocation rule $a_i = \alpha(s_i^{(c)})$ of Section 3.2 becomes a concrete product artifact.

6.5 The outcome feedback loop

The long-term moat of the architecture is its feedback loop: behavior → allocation → outcome → improved future allocation. Each allocation decision generates observable outcomes (did rewarded wallets retain, did incentivized liquidity remain, did qualified borrowers repay), and these outcomes supervise the next generation of models. Writing y_i for the realized outcome of an allocation to wallet i and L for a loss comparing predicted and realized value, the parameters are refined iteratively:

$$\theta_{t+1} \;=\; \theta_t \;-\; \eta\, \nabla_\theta \sum_{i \in \mathcal{A}_t} L\big(f_\theta(\mathbf{x}_i(t)),\, y_i\big), \tag{6}$$

where $\mathcal{A}_t$ is the set of wallets allocated to in round t and η a learning rate. Each round of allocation therefore improves the estimator that drives the next round, an advantage that accrues only to systems that operate inside the allocation workflow rather than beside it.

6.6 Adversarial-robust reward attribution

Because reward attribution attaches direct financial payoffs to the score, the scoring function itself becomes an attack surface: a reputation score misjudging a wallet produces a mislabeled profile, whereas a reward function misjudging a wallet pays the attacker. ZeruAI's attribution research [3] addresses this threat model with a framework whose design and evaluation inform the Zaps allocation engine. Attribution is computed per wallet–protocol pair (w, p) rather than per wallet, and each pair receives a composite score built from three independently bounded dimensions: percentile-normalized activity volume $v_{w,p}$, capped temporal engagement $h_{w,p}$, and a behavioral quality signal $q_{w,p}$ backed by the zScore engine,

$$s_{w,p} \;=\; \alpha\, v_{w,p} \;+\; \beta\, h_{w,p} \;+\; \gamma\, q_{w,p}, \qquad \alpha + \beta + \gamma = 1, \quad \alpha, \beta, \gamma > 0, \tag{7}$$

so that no single component dominates the composite through scale alone. Volume is normalized against a high percentile of each protocol's own participant distribution, which bounds whale dominance while preserving differentiation across the remaining population.

A cross-domain weighting mechanism then prices each protocol's economic significance. Writing V_p for total volume at protocol p, $V_{\sigma(p)}$ for total volume in its sector, and $V_{\mathcal{E}}$ for total ecosystem volume, the effective weight and the wallet's total attribution are

$$\omega_p \;=\; \frac{V_p}{V_{\sigma(p)}} \cdot \frac{V_{\sigma(p)}}{V_{\mathcal{E}}} \;=\; \frac{V_p}{V_{\mathcal{E}}}, \qquad R_w \;=\; \mu(w) \sum_{p \in \mathcal{P}_w} s_{w,p}\, \omega_p, \tag{8}$$

where $\mathcal{P}_w$ is the wallet's active protocol set and $\mu(w) \in (0, 1]$ a graduated penalty multiplier. The framework proves that the attribution any single wallet can obtain at a protocol is bounded by that protocol's global volume share, $s_{w,p}\, \omega_p \le S \cdot V_p / V_{\mathcal{E}}$ for the per-dimension scale S, independently of how cheaply local dominance at that protocol can be purchased [3]. This closes the most common structural exploit in volume-based reward programs: farming a niche protocol is structurally unprofitable regardless of the attacker's local share.

The penalty multiplier $\mu(w)$ is fed by a four-layer adversarial detection stack: transaction-level integrity gates (same-block round trips and implausible fee-to-volume ratios contribute nothing to any score); a parallel anomaly ensemble pairing a one-class reconstruction model, trained on labeled malicious wallets only, with an isolation-forest outlier detector; post-distribution memory, under which wallets that immediately liquidate rewards carry that record into future allocation cycles; and graph-based sybil clustering over the funding-provenance graph, which resolves coordinated multi-wallet operations that per-wallet detectors cannot reach. Penalties are graduated rather than binary, bounding the harm of a false positive to a proportional reduction instead of total exclusion.

The framework is evaluated empirically [3]. On a labeled corpus of 1,073 malicious wallets spanning 124,638 transactions, the anomaly ensemble reaches 0.923 ± 0.013 ROC-AUC against 0.891 ± 0.016 for the reconstruction model alone, with the gain conditional on fitting the isolation forest on the benign population; fitting it on the pooled population inverts its polarity and destroys the ensemble gain entirely. Under controlled adversarial simulation, the defense stack removes 30–90% of reward capture across attack archetypes (diversity farmers, spam bots, flash-loan exploiters, sybil operators) while legitimate-user scenarios shift by only 1–8%.

7 zScore: The Behavioral Trust Primitive

zScore is ZeruAI's core behavioral trust primitive, the concrete realization of the trust signal s_i of Equation (3). It maps wallet behavior into interpretable trust signals that can be consumed by applications, protocols, users, APIs, explorers, and ecosystem partners.

The primitive is designed to answer the questions that recur across allocation decisions: whether a wallet is economically active; whether its activity is high quality or shallow; whether it is durable or extractive; whether it retains rewards or farms and exits; whether it provides useful liquidity; whether it trades organically; whether it shows healthy credit behavior; whether it shows risky counterparty exposure; and whether it creates measurable economic contribution. Each of these questions corresponds to structure in the behavioral feature vector, and the score is intended to summarize that structure in a form a decision-maker can act on.

zScore may be surfaced through explorer integrations, wallet profiles, API endpoints, campaign logic, protocol dashboards, and onchain or offchain attestations where appropriate. The breadth of the exposure surface matters: a trust primitive is useful in proportion to how close it sits to the moment of decision, whether that moment is a user inspecting a counterparty on a block explorer or a protocol computing reward tiers.

Two properties bound the primitive's interpretation. First, zScore is not a moral judgment; it is a decision-support signal, and its output should inform rather than replace the allocator's decision rule. Second, its value depends on context, transparency, explainability, and continuous improvement. These properties are enforced architecturally by the domain-specific layer of Section 6 and the feedback loop of Equation (6), and operationally by the responsible-use constraints of Section 17.

8 Integration and Exposure Surface

ZeruAI has already moved beyond concept stage: the infrastructure has meaningful exposure across explorer surfaces, public products, API usage, and protocol workflows.

8.1 Explorer visibility

zScore is live through explorer surfaces, including Etherscan and Basescan, and is designed for EVM-ecosystem expansion across Etherscan-family explorers and supported EVM environments. Explorer exposure matters because block explorers are where users, protocols, analysts, and counterparties already inspect wallets, transactions, contracts, and onchain history. By appearing at the wallet-inspection layer, zScore becomes visible at precisely the moment users are evaluating trust.

8.2 EVM coverage

The system is architected for EVM-compatible activity coverage. This allows wallet behavior to be evaluated across Ethereum, Base, HyperEVM / Hyperscan surfaces, and other EVM ecosystems as integrations and indexing support expand. Cross-chain coverage strengthens the feature vector of Equation (2): a wallet's behavioral record is the union of its activity across chains, not a single-chain slice.

8.3 The Zaps application

The Zaps beta is live at app.zaps.wtf. Zaps allows users and protocols to interact with ZeruAI's trust-oracle infrastructure through campaigns, clans, zScore interactions, leaderboards, and capital-allocation workflows. Section 9 describes the application in detail.

8.4 API customers and workflow integrations

ZeruAI APIs and workflow integrations have already supported protocol use cases across airdrop analysis, sybil reduction, incentive filtration, wallet-quality segmentation, credit-related workflows, programmable yield, and capital-allocation decisioning. Current and past protocol customers include GAIB, Citrea, Unlloo, Bion, Upscale, and ChangeNOW.

9 Product Architecture and the Zaps Application

9.1 Product architecture

ZeruAI is not a single app; it is core behavior-intelligence infrastructure with several products built on top. Figure 3 shows the product architecture: the trust-oracle infrastructure produces zScore, and on that primitive sit the Zaps capital-allocation application and the three API lines. Zaps spans trading campaigns, liquidity mining, airdrops, public launches and IDOs, protocol usage, and curators and clans. zDrop covers airdrop allocation, sybil-aware reward logic, and incentive filtration; zLend covers credit readiness, borrower segmentation, and yield access; zScreen covers wallet screening, counterparty risk, and AML-support workflows.

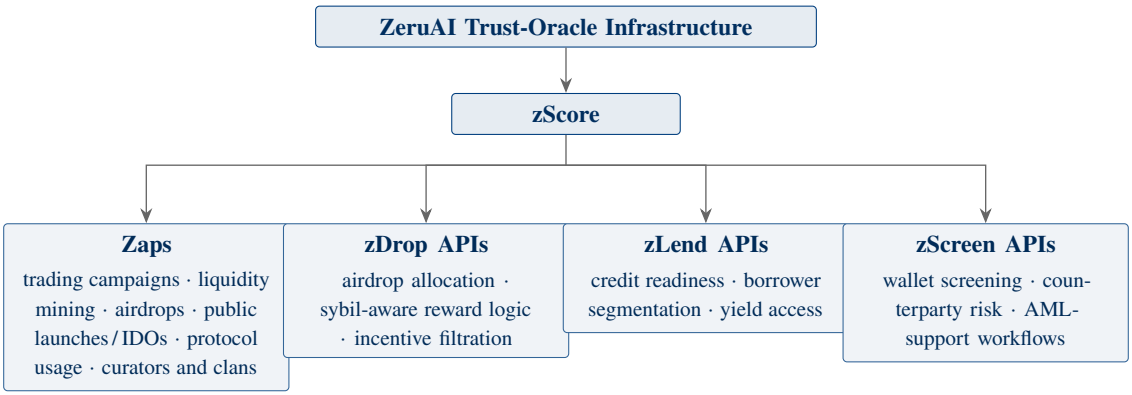

Figure 3: ZeruAI product architecture: one primitive, four product surfaces.

9.2 Zaps: a one-stop capital allocation app

Zaps is the primary network application built on ZeruAI infrastructure. Its long-term vision is to become a one-stop capital-allocation app for crypto-native economies, serving three constituencies whose needs are complementary.

Protocols use Zaps to allocate incentives, liquidity rewards, trading rewards, airdrops, access, launch allocations, public-sale opportunities, app-usage rewards, curator rewards, differentiated yield, and eventually credit opportunities. Users use Zaps to check and mint zScore, join campaigns, join or create clans, follow or publish trading calls, provide liquidity, participate in launches, build behavioral reputation, earn rewards, and unlock future opportunities. Curators use Zaps to create clans, mobilize communities, publish strategies or calls, bring users into campaigns, stake credibility, prove economic impact, and earn rewards based on measurable outcomes.

Zaps should therefore be understood as a capital-allocation app, not merely a trading product. Product rollout, however, is deliberately modular: trading is the first wedge because it is social, frequent, measurable, and community-driven, and the roadmap expands into mini-apps for liquidity, launches, airdrops, protocol usage, and credit-readiness.

9.3 Product strategy

Three product strategies were considered, compared in Table 6. Launching as a fully developed one-stop app from day one carries a strong category story but is too broad and slows product focus. A trading-first app is a fast wedge with a social loop and frequent usage, but it undercuts the capital-allocation-network positioning and risks being read as a trading social app. The recommended strategy is the third: position Zaps as the one-stop Capital Allocation App, ship Trading Zaps as the first live wedge, and roll out the remaining workflows (liquidity, launches, airdrops, protocol usage, and credit) as modular mini-apps (Figure 4). This preserves the full category narrative while remaining execution-focused; its cost is that it requires a clear roadmap and product architecture, which Sections 10 and 16 supply.

Table 6: Zaps product-strategy comparison.

Strategy	Description	Advantage	Risk	Assessment
One-stop app from day one	Zaps launches with all workflows equally developed	Strong category story	Too broad, slower product focus	Not ideal for initial execution
Trading-first app	Zaps focuses mainly on trading calls and trading campaigns	Fast wedge, social loop, frequent usage	Undercuts CAN positioning and may look like trading social app	Useful as launch wedge, but not full positioning
Capital allocation app with modular mini-apps	Zaps is positioned as one-stop CAN, with Trading Zaps as first wedge and other workflows rolling out as mini-apps	Strong category + focused execution	Requires clear roadmap and product architecture	Recommended

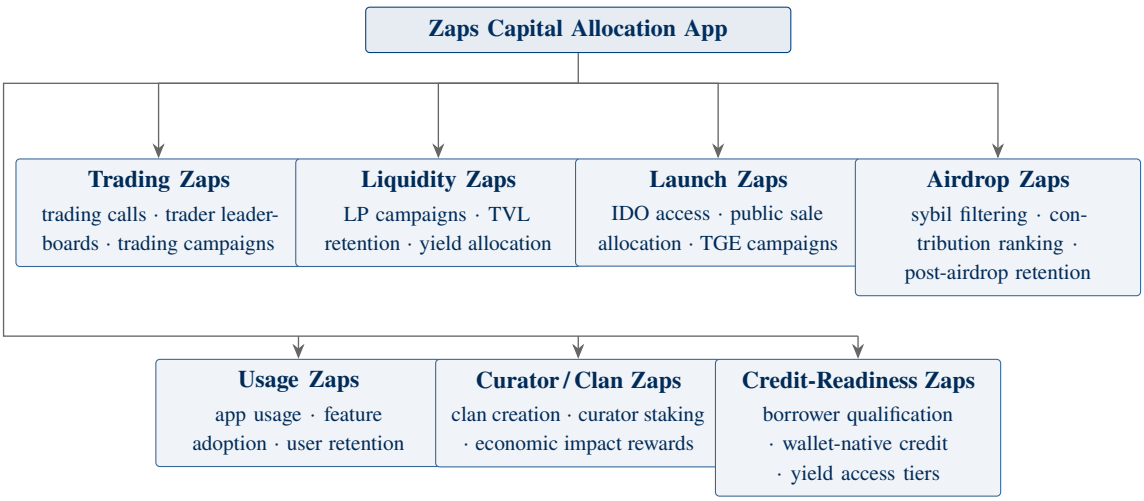

Figure 4: The Zaps modular product model: seven mini-apps sharing one allocation engine.

10 The Zaps Allocation Workflow

10.1 Stakeholders

Zaps coordinates three stakeholders in a single allocation loop. Protocols contribute capital, campaigns, rewards, and access, and receive better allocation, wallet ranking, analytics, and outcome measurement. Users contribute wallet behavior, liquidity, trading, and app usage, and receive rewards, access, reputation, and future eligibility. Curators and clans contribute distribution, research, and communities, and receive attribution, economic-impact rewards, and clan reputation. Table 7 summarizes the exchange.

Table 7: Stakeholders in the Zaps allocation loop.

Stakeholder	**Contribution**	**What Zaps provides**
Protocols	Capital, campaigns, rewards, access	Better allocation, wallet ranking, analytics, outcome measurement
Users	Wallet behavior, liquidity, trading, app usage	Rewards, access, reputation, future eligibility
Curators / clans	Distribution, research, communities	Attribution, economic-impact rewards, clan reputation

10.2 Campaign lifecycle

Figure 5 traces a campaign through the system. A protocol creates a campaign, defines its objective, and deposits rewards into an allocation pool. Zaps requests wallet trust and behavior signals from the zScore layer. Curators create clans and activate their communities; users join the campaign, connect wallets, and act by trading, providing liquidity, using the protocol's app, or joining a launch. Zaps evaluates both historical and in-campaign behavior against the zScore layer, then produces rankings, eligibility lists, and reward tiers for the protocol. After the protocol approves the allocation, Zaps distributes rewards, access, or eligibility to users, attributes curator impact, and feeds outcome data back into the model layer, closing the loop of Equation (6).

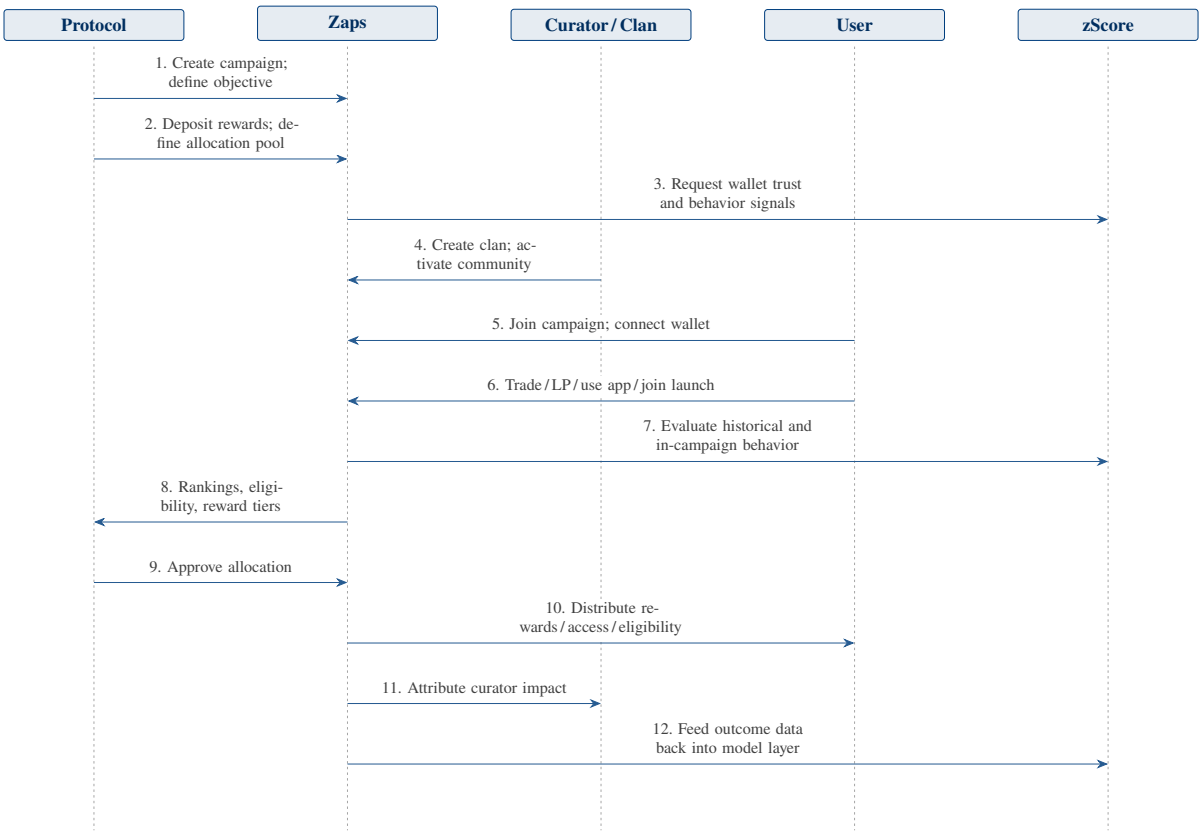

Figure 5: The Zaps capital-allocation workflow. Messages are numbered in campaign order; step 12 closes the outcome-feedback loop.

In the notation of Section 3.2, the campaign implements the allocation rule $a_i = \alpha(s_i^{(c)})$ with c the campaign's decision context, and curator attribution decomposes realized campaign value across the users each curator mobilized. Writing $\mathcal{U}_k$ for the set of users referred by curator k and y_i for user i's realized in-campaign value, curator k's economic impact is measured as

$$I_k = \sum_{i \in \mathcal{U}_k} y_i, \tag{9}$$

so that curators are rewarded for the measurable economic outcomes of the users they bring, not for attention alone.

10.3 Mini-app workflows

Each mini-app specializes the same loop to one allocation surface, differing only in the behavioral inputs it weighs.

Trading Zaps supports trading calls, trading campaigns, trader leaderboards, and curator-led trading clans. A DEX, perpetuals protocol, or exchange can use it to reward quality-adjusted trading activity rather than raw volume alone, weighing trading frequency, volume quality, risk-adjusted behavior, drawdown, return participation, wash-like indicators, and historical wallet quality.

Liquidity Zaps supports LP campaigns, TVL incentives, DeFi looping campaigns, and yield-access workflows. A protocol can reward users not merely for supplying liquidity but for supplying durable and useful liquidity, weighing LP duration, withdrawal timing, capital rotation, prior LP history, incentive sensitivity, retained TVL, and pool context.

Launch Zaps supports public launches, IDOs, TGE access, community-sale allocations, and access-gated opportunities. A project can allocate launch access on behavioral quality rather than first-come-

first-served participation or simple social quests, weighing prior launch participation, reward dumping, ecosystem alignment, capital commitment, wallet age, and sybil relationships.

Airdrop Zaps supports retroactive allocations, contribution scoring, sybil filtering, wallet segmentation, and post-airdrop monitoring. An ecosystem can reward wallets that created genuine value before the reward and track whether they remain after distribution, weighing breadth and depth of ecosystem usage, transaction timing, retention, app diversity, reward history, governance behavior, and post-airdrop selling.

Usage Zaps supports app onboarding, feature adoption, retention, referrals, consumer-crypto activation, and loyalty programs. A protocol can reward milestones rather than one-off actions, weighing activation quality, repeated usage, funded-wallet behavior, referral quality, retained balance, and post-campaign activity.

Curator and Clan Zaps supports creator distribution, trading clans, LP communities, ecosystem communities, and campaign-specific groups. Curators are rewarded on economic impact rather than attention alone, per Equation (9), weighing the quality of referred wallets, capital deployed, retained TVL, trading quality, app usage, and user retention.

11 API Product Lines

ZeruAI's API stack converts the same trust-oracle infrastructure into enterprise- and protocol-facing products. Where Zaps operates the allocation loop end-to-end, the APIs let external systems consume the signals inside their own workflows.

11.1 zDrop APIs

zDrop APIs support airdrops, incentives, public launches, and reward allocation. Use cases include sybil filtering, wallet-quality ranking, airdrop eligibility, reward tiering, launch allocation, dumping-risk analysis, incentive strategy, and post-campaign reporting. Past and current use cases include airdrop and incentive workflows with protocols such as GAIB and Citrea.

11.2 zLend APIs

zLend APIs support credit, lending, yield, and underwriting workflows. Use cases include borrower segmentation, credit-readiness scoring, repayment-behavior analysis, undercollateralized-lending support, differentiated yield access, and wallet-native risk analytics. Past and current use cases include credit-related workflows with customers such as Unlloo and Bion.

11.3 zScreen APIs

zScreen APIs support AML-adjacent wallet screening, counterparty intelligence, and compliance-support workflows. Use cases include wallet risk screening, source-of-funds context, suspicious-behavior indicators, pre-transaction checks, risky-pattern detection, and enhanced due-diligence triggers. zScreen is designed to complement regulated AML and compliance tooling, not to replace legal, compliance, or MLRO judgment.

12 Related Work and Market Landscape

The market around ZeruAI has three major categories, each of which validates part of the thesis while leaving the intersection unoccupied.

12.1 Reputation and ranking protocols

Reputation systems validate the need for trust, ranking, and reputation in open networks. OpenRank-style systems show that contextual reputation graphs and algorithms such as EigenTrust can produce useful rankings and trust signals. However, reputation infrastructure often struggles when it does not own the recurring workflow where capital moves: a score with no allocation surface attached must persuade every downstream consumer separately, and it never observes the outcomes of the decisions it informs.

12.2 Distribution and incentive platforms

Distribution platforms, including Galxe, Zealy, Layer3, Merkl, Kaito, and Cookie3, validate that protocols already spend capital to acquire users, distribute rewards, drive liquidity, reward attention, and activate communities. These platforms prove that budgets exist. However, distribution often rewards observable actions such as quest completion, social engagement, supplied TVL, raw trading volume, or referrals. Those signals are useful, but they do not always prove recipient quality, durability, retention, or economic contribution. This is precisely the estimator gap identified in Equation (5).

12.3 Wallet analytics, risk, and API businesses

Analytics and risk businesses provide data, wallet intelligence, labels, screening, or API access. They are useful, but they often sit outside the actual allocation workflow: they help interpret data, but they do not always operate the capital-allocation loop, and therefore cannot benefit from the outcome feedback of Equation (6).

12.4 ZeruAI's position

ZeruAI sits at the intersection of these categories, combining trust depth with distribution ownership. Table 8 summarizes the comparison: reputation protocols prove that trust signals matter but have weak distribution ownership, which ZeruAI addresses by tying trust to allocation workflows; distribution platforms prove that protocols pay to move capital but carry shallow or campaign-specific trust, which Zaps addresses by combining distribution with behavioral trust; wallet-analytics and API businesses prove that wallet data is valuable but often sit outside the decision workflow, whereas ZeruAI both exposes APIs and runs allocation products; compliance tooling proves that risk screening is required but focuses on AML rather than contribution, which zScreen extends with behavior intelligence; and credit scoring proves that underwriting needs trust but is hard to adapt to identity-optional wallets, which zLend addresses with wallet-native behavioral credit signals.

Table 8: Market categories and ZeruAI's differentiation.

Category	What it proves	Limitation	ZeruAI differentiation
Reputation protocols	Trust signals matter	Weak distribution ownership	ZeruAI ties trust to allocation workflows
Distribution platforms	Protocols pay to move capital	Shallow or campaign-specific trust	Zaps combines distribution with behavioral trust
Wallet analytics / APIs	Wallet data is valuable	Often outside decision workflow	ZeruAI exposes APIs and runs allocation products
Compliance tooling	Risk screening is required	Focused on AML, not contribution	zScreen adds behavior intelligence to screening workflows
Credit scoring	Underwriting needs trust	Hard to adapt to identity-optional wallets	zLend provides wallet-native behavioral credit signals

Figure 6 plots the landscape on two axes: trust depth and distribution ownership. Reputation infrastructure (OpenRank) is deep on trust but low on distribution; distribution platforms (Galxe / Zealy, Layer3, Merkl, Kaito / Cookie3) own distribution but carry shallower trust; wallet analytics is low on both. Zaps targets the upper-right quadrant, the Capital Allocation Network position. The category ZeruAI is building is *trust-oracle infrastructure for identity-optional economies*.

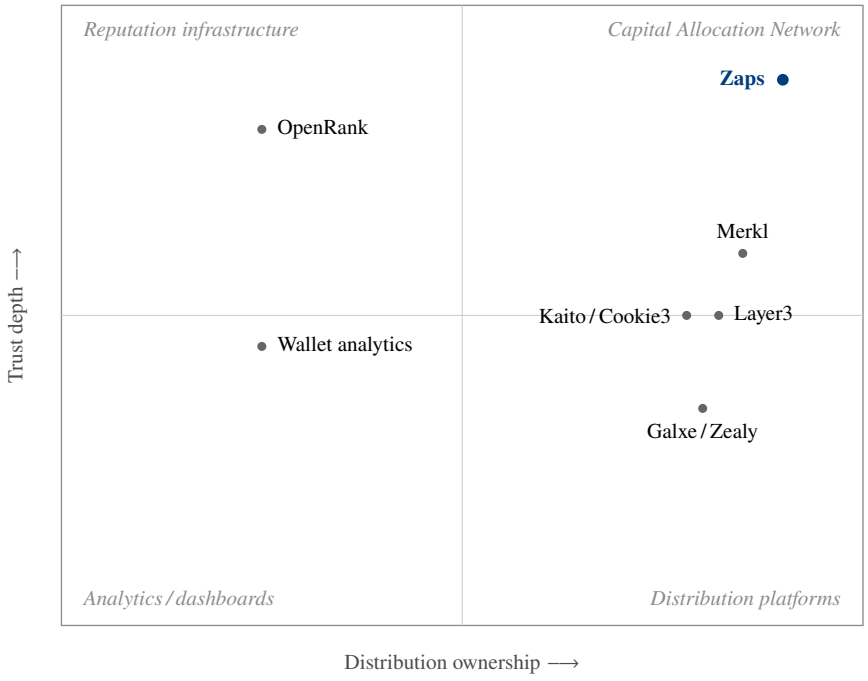

Figure 6: Market landscape: trust depth versus distribution ownership. Coordinates follow the source positioning; Zaps targets the upper-right quadrant.

13 Token Economics

13.1 The role of $ZERU

$ZERU is the ZeruAI ecosystem token, designed to coordinate access, participation, staking, incentives, rewards, and ecosystem alignment across ZeruAI products. It is not positioned as a passive governance token; its role is to support network coordination across protocols, users, curators, clans, API users, and

ecosystem partners.

Table 9 summarizes the token's high-level utility. Users may access selected campaigns or premium tiers; protocols may use or stake $ZERU for campaign setup, fee benefits, or reward matching; curators may stake $ZERU to create clans, access campaigns, or signal credibility; commitment staking may reduce costless participation and improve accountability; users, curators, and communities may earn $ZERU for verified contribution through proof-of-economic-activity rewards; the token may support API access, fee discounts, premium features, or ecosystem privileges; and it may support campaign bootstrapping, partnerships, and user activation.

Table 9: $ZERU utility areas.

Utility area	**Description**
Campaign participation	Users may access selected campaigns or premium tiers
Protocol campaign creation	Protocols may use or stake $ZERU for campaign setup, fee benefits, or reward matching
Curator and clan coordination	Curators may stake $ZERU to create clans, access campaigns, or signal credibility
Commitment staking	$ZERU may reduce costless participation and improve accountability
Proof-of-economic-activity rewards	Users, curators, and communities may earn $ZERU for verified contribution
API and ecosystem access	$ZERU may support API access, fee discounts, premium features, or ecosystem privileges
Network growth	$ZERU may support campaign bootstrapping, partnerships, and user activation

The guiding principle links the token to the trust loop developed in Sections 3–10: behavior creates reputation, reputation determines allocation, and $ZERU coordinates repeated participation across the network. Commitment staking can be read as a mechanism-design device: requiring a stake $\sigma > 0$ to participate imposes a cost on low-quality or sybil participation while remaining negligible for genuine participants, who expect rewards exceeding the stake's opportunity cost. Participation is rational when

$$\mathbb{E}\big[\, r_i \mid s_i \,\big] \;>\; c(\sigma), \tag{10}$$

where r_i is the participant's expected reward under its trust signal s_i and $c(\sigma)$ the cost of staking. Because $\mathbb{E}[r_i \mid s_i]$ is increasing in behavioral quality, the constraint filters exactly the participants the network wants to exclude.

13.2 Token allocation

Table 10 presents the indicative token allocation, visualized in Figure 7: 30% to the community (users, contributors, curators, clans, and proof-of-economic-activity rewards); 20% to investors, public and private (capital formation, strategic participation, and public distribution); 20% to the team and advisors (long-term execution, technical continuity, research, and ecosystem growth); 15% to the treasury (operations, liquidity, infrastructure, security, and ecosystem maintenance); and 15% to a strategic reserve (campaign bootstrapping, reward matching, launch partnerships, market structure, and expansion).

Table 10: Indicative $ZERU token allocation.

Allocation bucket	Share	Purpose
Community	30%	Users, contributors, curators, clans, and proof-of-economic-activity rewards
Investors, public + private	20%	Capital formation, strategic participation, and public distribution
Team and advisors	20%	Long-term execution, technical continuity, research, and ecosystem growth
Treasury	15%	Operations, liquidity, infrastructure, security, and ecosystem maintenance
Strategic reserve	15%	Campaign bootstrapping, reward matching, launch partnerships, market structure, and expansion

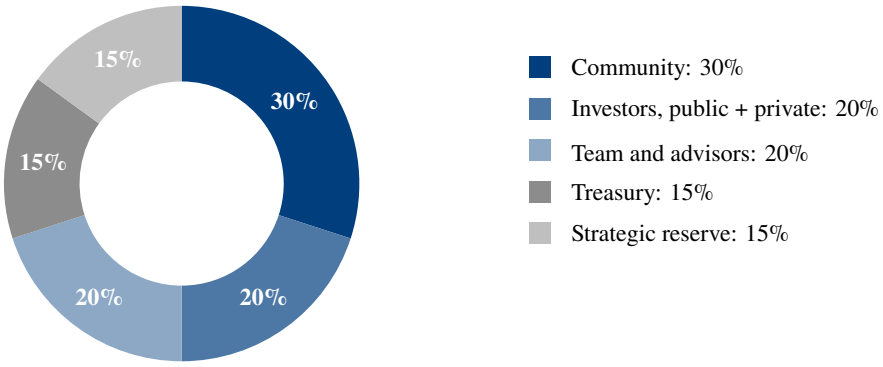

Figure 7: Indicative $ZERU allocation (data from Table 10).

A design constraint governs the allocation: the token supply must remain sufficiently reserved for network utility. If too much supply is consumed by short-term listing, market-making, investor, or promotional obligations, the network may not have enough productive token inventory to scale. The objective is not merely to create a traded asset but a productive coordination asset for trust-based capital allocation.

14 Business Model

ZeruAI's revenue does not depend solely on token appreciation. The token strengthens network coordination, while the infrastructure generates revenue through campaigns, APIs, analytics, enterprise integrations, and ecosystem products. Table 11 enumerates the revenue lines. Zaps produces campaign revenue (campaign setup fees, scoring fees, eligibility fees, and premium analytics) and marketplace revenue (marketplace fees, success fees, and curator-marketplace fees), alongside white-label deployments of protocol-branded campaign and allocation workflows. The API stack monetizes the same signals in enterprise form: zDrop through airdrop analysis, sybil filtering, allocation files, and incentive reporting; zLend through borrower scoring, credit-readiness, underwriting signals, and risk segmentation; zScreen through wallet screening, pre-transaction checks, and counterparty reports. Enterprise integrations add custom scoring, data products, dashboards, and ecosystem partnerships.

Table 11: Revenue lines.

Revenue line	**Description**
Zaps campaign revenue	Campaign setup fees, scoring fees, eligibility fees, premium analytics
Zaps marketplace revenue	Marketplace fees, success fees, curator marketplace fees
White-label deployments	Protocol-branded campaign and allocation workflows
zDrop APIs	Airdrop analysis, sybil filtering, allocation files, incentive reporting
zLend APIs	Borrower scoring, credit-readiness, underwriting signals, risk segmentation
zScreen APIs	Wallet screening, pre-transaction checks, counterparty reports
Enterprise integrations	Custom scoring, data products, dashboards, ecosystem partnerships

15 Traction and Proof Points

ZeruAI has already moved beyond concept stage, and the evidence spans scale, revenue, and deployment surface. On scale, more than 320 million wallets have been scored and the APIs have served more than 750 thousand calls. On revenue and adoption, the company has paying protocol customers (current and past customers include GAIB, Citrea, Unlloo, Bion, Upscale, and ChangeNOW) whose use cases span airdrop allocation, sybil reduction, wallet-quality segmentation, credit-underwriting signals, programmable-yield logic, and capital-allocation workflows. On deployment surface, zScore has live explorer exposure with checks running through public wallet-inspection surfaces; API use cases span airdrops, incentives, credit, programmable yield, and wallet intelligence; the Zaps beta is live at app.zaps.wtf; and ecosystem coverage is expanding across EVM-compatible environments.

The attribution framework of Section 6.6 adds measured campaign-level evidence. In live token-distribution campaigns deploying the full adversarial-detection stack, measured against comparable unmitigated campaigns, the framework recorded a 56% reduction in sybil allocation, a 49% increase in quality-wallet participation, and a 50% reduction in post-distribution sell pressure; allocation to low-quality wallets fell from 21.0% to 13.5% of the distributed pool while allocation to high-quality wallets rose from 41.7% to 54.7%, and mean post-distribution holding duration extended from 9.8 to 12.6 days [3].

Read against the formulation of Section 3.2, these figures are evidence that the estimator exists at production scale (wallets scored), that external systems consume it (API calls, explorer checks), that the signal commands a price (paying customers), and that deploying it measurably improves allocation outcomes (campaign results). The next stage is to convert bespoke API and workflow integrations into repeatable products through Zaps and the API stack.

16 Roadmap

ZeruAI's roadmap preserves the full Capital Allocation Network vision while using focused product wedges for execution. Table 12 lays out the seven phases, and Figure 8 summarizes the sequence. Phase 1 establishes the trust-oracle infrastructure (zScore, explorer exposure, and API customers) to establish wallet behavior as a scalable trust primitive. Phase 2 ships the trading-first Zaps beta (trading calls, trading campaigns, clans, leaderboards) to build a frequent user loop and curator participation. Phase 3 adds the incentive and launch mini-apps (Airdrop, Launch, and Usage Zaps) to expand from trading into broader capital allocation. Phase 4 adds liquidity and DeFi mini-apps (Liquidity Zaps, TVL campaigns, yield access) to capture DeFi incentive and liquidity workflows. Phase 5 scales the API stack (zDrop, zLend, zScreen) to serve protocols, credit providers, payments companies, and VASPs. Phase 6 activates the ecosystem token ($ZERU access, staking, rewards, fee benefits) to coordinate users, curators, protocols, and ecosystem partners. Phase 7 extends into tokenized finance and agents (agent reputation,

RWA risk, trade finance, invoice finance), carrying trust oracles into tokenized capital markets.

Table 12: Roadmap phases.

Phase	Product focus	Key modules	Strategic objective
Phase 1	Trust-oracle infrastructure	zScore, explorer exposure, API customers	Establish wallet behavior as scalable trust primitive
Phase 2	Trading-first Zaps beta	Trading calls, trading campaigns, clans, leaderboards	Build frequent user loop and curator participation
Phase 3	Incentive and launch mini-apps	Airdrop Zaps, Launch Zaps, Usage Zaps	Expand from trading into broader capital allocation
Phase 4	Liquidity and DeFi mini-apps	Liquidity Zaps, TVL campaigns, yield access	Capture DeFi incentive and liquidity workflows
Phase 5	API stack scale	zDrop, zLend, zScreen	Serve protocols, credit providers, payments companies, VASPs
Phase 6	Ecosystem token activation	$ZERU access, staking, rewards, fee benefits	Coordinate users, curators, protocols, and ecosystem partners
Phase 7	Tokenized finance and agents	Agent reputation, RWA risk, trade finance, invoice finance	Extend trust oracles into tokenized capital markets

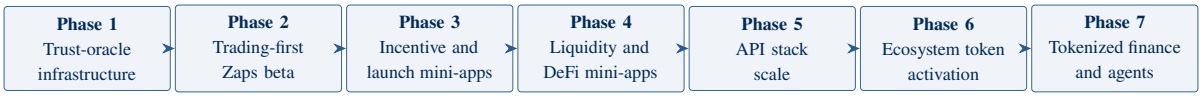

Figure 8: Execution sequence of the seven roadmap phases (data from Table 12).

The sequencing avoids two symmetrical risks: Zaps becoming too broad too early, and Zaps being misread as only a trading product. The external positioning remains constant throughout: Zaps is the Capital Allocation App, while the internal execution sequence begins with the trading-first wedge and proceeds through modular mini-app expansion.

17 Limitations, Risk, and Responsible Use

ZeruAI outputs are designed as decision-support signals. They should not be interpreted as guarantees, legal opinions, compliance clearance, financial advice, investment recommendations, credit approvals, or risk-free assessments. Protocols and enterprises using ZeruAI infrastructure should apply appropriate legal, compliance, risk, operational, and human review.

The limitations of behavioral trust inference deserve explicit statement. Data coverage is incomplete, and chain-specific blind spots exist wherever indexing has not reached; the feature vector of Equation (2) is only as complete as the event history behind it. Wallet behavior itself evolves, and adversarial actors adapt to whatever signal is deployed, so any published estimator invites gaming; the attribution research documents this concretely, noting that offline-fitted detectors are exposed to drift under adaptive adversaries and that single-chain funding-provenance clustering can be evaded by cross-chain funding, making cross-chain identity resolution a consequential open extension [3]. Models drift as the distribution of behavior shifts, and any classifier produces both false positives and false negatives. Jurisdictional requirements and protocol-specific context further condition how a signal may be used, and data-availability constraints bound what can be inferred at all.

ZeruAI's approach to responsible trust infrastructure addresses these limitations structurally rather than rhetorically: contextual scoring instead of one universal judgment; model explainability; risk flags rather than absolute judgments; customer-defined thresholds so that the allocator, not the oracle, sets

the decision boundary; privacy-preserving design where applicable; auditability; and human review for high-stakes decisions. The governing principle is that trust oracles should assist decision-making; they should not remove accountability from the decision-maker.

18 Conclusion

The next phase of crypto and tokenized finance requires a new trust primitive. Identity will remain important in regulated systems, but identity alone cannot serve identity-optional economies; the scalable primitive is behavior. This paper has presented ZeruAI, which converts wallet behavior into trust-oracle infrastructure: zScore is the behavioral trust primitive; Zaps is the one-stop capital-allocation app built on it; zDrop APIs improve airdrops and incentives; zLend APIs support wallet-native credit and underwriting; zScreen APIs support screening and compliance-adjacent workflows; and $ZERU coordinates the ecosystem. The long-term vision is for ZeruAI to become the default trust-oracle infrastructure layer for wallet-based economies, tokenized capital markets, and agentic economies. The core question remains the one posed at the outset, *who deserves capital?*, and ZeruAI is building the infrastructure to answer it.

References

[1] H. Udupi, A. Sahoo, A. S. P., G. S., P. Paul, and P. C. Martens. zScore: A universal decentralised reputation system for the blockchain economy. *arXiv preprint arXiv:2503.05718*, 2025.

[2] D. Kandaswamy, A. Sahoo, A. SP, G. S, P. Paul, and G. G. N. Deep reputation scoring in DeFi: zScore-based wallet ranking from liquidity and trading signals. *arXiv preprint arXiv:2507.20494*, 2025.

[3] G. G N, A. Sahoo, A. SP, G. S, and P. Paul. ZAPs: A reward attribution framework for DeFi ecosystems with adversarial-robust scoring via parallel anomaly ensemble detection. ZeruAI research paper, 2026.