SOK | Received 9 May 2026; Revised 18 July 2026; Accepted 1 September 2026; Published 22 September 2026
https://doi.org/10.55092/blockchain20260008

Toward Web 4.0: bidirectional trust between AI agents and blockchain

Yunfeng Xia[1], Chao Li[1,*], Lei Li[1], Chenhao Zhang[1], Li Duan[1], Runhua Xu[2] and Wei Wang[3]

[1] Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China
[2] School of Computer Science and Engineering, Beihang University, Beijing, China
[3] School of Software Engineering, Xi'an Jiaotong University, Xi'an, China

* Correspondence author; E-mail: li.chao@bjtu.edu.cn.

Highlights:

- A bidirectional trust framework examines blockchain support for agents and agent participation in blockchain, with verifiable computation as their shared foundation.
- A formal interaction model specifies security properties, while five evaluation dimensions compare standards, research proposals, and industry projects.
- A taxonomy of agent autonomy, trust model, and operational direction organizes the ecosystem and locates nine open research problems.

Abstract: Autonomous artificial intelligence (AI) agents are increasingly deployed on blockchain platforms, yet the design space governing their interaction remains poorly understood. This convergence, in which autonomous agents operate on and within decentralized systems, characterizes the emerging Web 4.0 paradigm. We organize this Systematization of Knowledge (SoK) around a bidirectional trust framework. For the B $\rightarrow$ A direction (Blockchain $\rightarrow$ Agent), blockchain provides the trust infrastructure needed by agents. This direction follows their on-chain lifecycle: identity and account abstraction establish an agent on-chain; permission and delegation define what it may do; intent-centric execution carries out its goals; and tokenized agent economies support economic participation. The A $\rightarrow$ B direction (Agent $\rightarrow$ Blockchain) concerns participation in core blockchain mechanisms, beginning with security auditing and extending to consensus and governance. Verifiable computation forms the Trust Foundation (TF) shared by both directions. We consider zero-knowledge machine learning (zkML) and optimistic machine learning (opML) alongside trusted execution environments (TEEs). Their trade-offs between trust minimality and computational overhead differ, as does deployment readiness. The Agent–Blockchain Interaction Model (ABIM) formalizes this interaction. We catalog 70 Ethereum Improvement Proposals (EIPs) and Ethereum Request for Comments (ERC) standards in the Appendix and review 127 academic papers. The analysis also covers 20

representative industry projects. This material is compared using five dimensions: Verifiability, Minimality of Trust, Expressiveness, Composability, and Maturity. The comparison reveals three unresolved issues. The agent-specific standards ecosystem is predominantly immature, with only 3 of 13 direct AI/agent ERCs having reached Final status. Intent architectures lack formal analysis. Research on AI participation in consensus and governance remains limited to isolated studies, and a unified security framing that treats AI as a first-class actor at the protocol layer is still absent. We propose a three-dimensional taxonomy of agent autonomy, trust model, and operational direction. We identify nine concrete open problems and outline the principal research opportunities.

Keywords: AI agent; blockchain; account abstraction; intent architecture; verifiable computation; DAO governance

1. Introduction

Large language models (LLMs) and autonomous artificial intelligence (AI) agent frameworks have changed how intelligent systems interact with digital infrastructure [1]. In parallel, blockchain technology has developed beyond value transfer into a programmable trust layer [2]. Smart contracts on Ethereum-like chains support execution patterns such as privacy-preserving timed transactions [3]. Delegated Proof-of-Stake (DPoS) and related consensus designs power widely used Web 3.0 networks, including EOSIO, Steem, and TRON, despite known governance-takeover risks [4,5]. Above these layers, applications support cross-chain access control with decentralized storage [6], non-fungible token (NFT) marketplaces [7], and Web3 social platforms [8].

These developments meet when autonomous AI agents operate on and within blockchain systems. Their interaction opens a design space with implications for trust, security, and economic coordination. In the agent-centric interpretation of Web 4.0, autonomous AI agents become first-class network participants that interact, transact, and coordinate through decentralized infrastructure [9].

Because "Web 4.0" has competing interpretations, we state the operational definition used throughout this paper and compare it with alternative interpretations.

> **Definition (Web 4.0, as used in this paper).** The stage of internet evolution in which autonomous software agents, rather than human users mediated by applications, act as first-class network participants: they hold verifiable on-chain identities, control assets within delegated permissions, express goals as executable intents, and coordinate through decentralized trust infrastructure without per-action human authorization.

This usage follows the agent-centric interpretation in the blockchain literature [9]. It is narrower than the European Commission's 2023 policy framing, which defines Web 4.0 as the convergence of AI, the Internet of Things (IoT), blockchain and other distributed technologies, and immersive virtual worlds [10]: our definition isolates the single axis of that convergence, autonomous agents on decentralized infrastructure, that this survey systematizes, and deliberately excludes the metaverse/immersive dimension. We use the term as a framing device only; no technical claim in this paper depends on it.

The urgency of systematizing this space stems from three converging pressures. First, the economic stakes are already substantial: as of December 2024, the aggregate market valuation of blockchain-based

AI agents reached an estimated $8.6 billion [11], yet the interaction between AI-assisted development and security-critical smart contracts has already surfaced in high-profile incidents. For example, an oracle misconfiguration in the Moonwell protocol resulted in a $1.78M loss; a pull request associated with the affected component carried a "Co-Authored-By" trailer indicating AI tooling involvement, though Moonwell's official post-mortem did not attribute the root cause to AI-generated code [12]. The second concerns the widening trust gap: as agents transition from read-only analytics to autonomous transaction execution, the absence of formal trust models means that each project invents ad hoc security assumptions, creating a fragmented landscape where vulnerabilities compound across layers. The third concern is that the absence of academic foundations is becoming costly: with no prior Systematization of Knowledge (SoK) covering account abstraction, only one formal analysis of intent architectures [13], and no systematic framing for AI participation in consensus or governance, practitioners lack the theoretical foundations needed to reason about safety, incentive compatibility, and failure modes, precisely the properties that matter most as agent autonomy increases.

Ante [11] analyzes 306 projects that combine AI and blockchain technology, illustrating the already substantial scale of this convergence. Ethereum alone hosts more than 70 Ethereum Improvement Proposals (EIPs) and Ethereum Request for Comments (ERC) standards that directly or indirectly affect AI agent operations, covering account abstraction [14,15], agent identity registration [16,17], verifiable machine learning (ML) inference [18,19], intent-centric execution [20,21], smart contract delegation [22], and agent tokenization [23,24]. Industry growth has been explosive, but academic work has not kept pace: no prior SoK covers account abstraction, intent-driven architectures have received only one formal analysis [13], and although isolated studies have examined AI in consensus [25] and governance [26], a systematic security framing for AI agents as participants, rather than mere tools, in blockchain protocol mechanisms remains absent.

Our central thesis is that the relationship between AI agents and blockchain is bidirectional and symbiotic. In what we call the B $\rightarrow$ A direction (Blockchain $\rightarrow$ Agent), blockchain provides trust infrastructure for agents, including identity, permissioned execution, intent specification, and economic incentives. In the reverse A $\rightarrow$ B direction (Agent $\rightarrow$ Blockchain), agents bring intelligent capabilities to the blockchain itself through automated security auditing, consensus participation, and governance decision-making. A foundation of verifiable computation underpins both directions: the ability to cryptographically prove that an agent's reasoning and actions are correct without trusting any single party. The degree to which verifiable computation is needed depends on application context: high-stakes autonomous operations demand strong cryptographic guarantees, while lower-risk assistive agents may operate acceptably under weaker trust models such as reputation or economic bonds.

We organize the B $\rightarrow$ A direction into four layers that mirror the lifecycle of an agent's on-chain operation, each building on the previous one: an agent must first exist on-chain with a verifiable identity (A1: Account & Identity), then be authorized to act within bounded permissions (A2: Permission & Delegation), then express what it wants to achieve through intent specifications (A3: Intent & Execution), and finally be economically sustained through tokenization and incentive mechanisms (A4: Agent Economy). This ordering reflects a strict dependency chain: A2 cannot function without A1 (permissions require identity), A3 presupposes A2 (intent execution requires authorization), and A4 builds on all three (economic

participation requires identity, permissions, and execution capability). The A → B direction is organized into three layers ordered by increasing depth of participation: agents first observe and audit blockchain activity (B1: Security), then potentially validate transactions as consensus participants (B2: Consensus), and ultimately shape protocol evolution through governance (B3: Governance). The Trust Foundation (TF) of verifiable computation is not a separate layer but a cross-cutting substrate: it provides the correctness guarantees that all other layers may draw upon, with the strength of guarantee varying by application risk and agent autonomy. Figure 1 visualizes this architecture; the vertical arrows from TF to both directions indicate this foundational role, while the horizontal dashed arrows between corresponding A and B layers reflect that the two directions are mutually reinforcing rather than independent, since, for example, the same identity infrastructure (A1) that enables agent operations also enables verifiable participation in governance (B3).

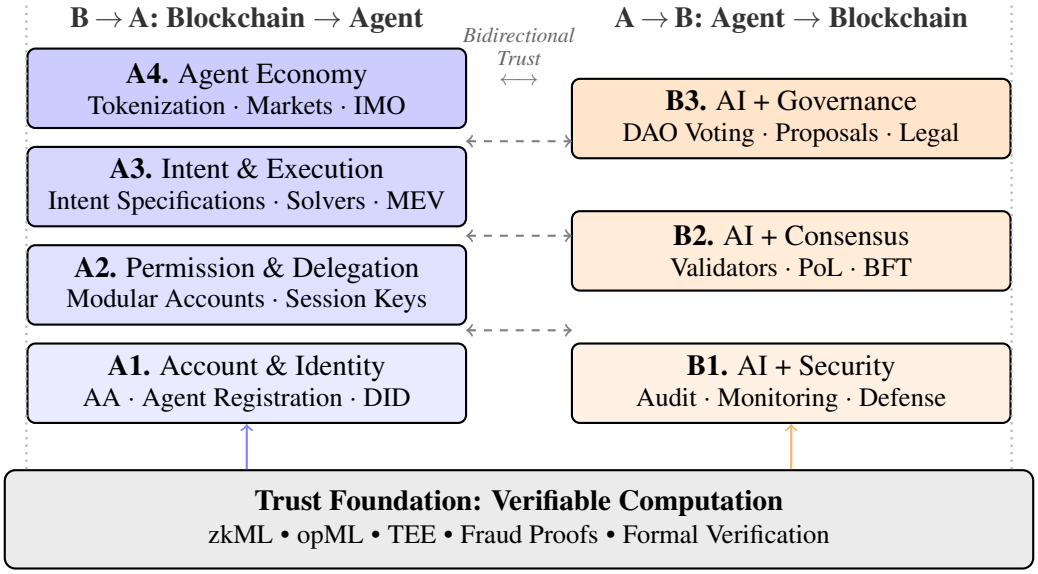

Figure 1. Bidirectional Trust Framework for autonomous AI agents on blockchain. Blockchain supports agents through four layers (left), while agents contribute to security, consensus, and governance (right); verifiable computation underpins both directions. Abbreviations: AA, account abstraction; DID, decentralized identifier; MEV, maximal extractable value; IMO, initial model offering; PoL, proof of learning; BFT, Byzantine fault tolerance; DAO, decentralized autonomous organization.

This paper makes six contributions. (i) We formalize the interaction through an Agent-Blockchain Interaction Model (ABIM) that captures the trust assumptions and security properties at each layer (Section 2). (ii) We introduce a five-dimensional evaluation framework (Verifiability, Minimality of Trust, Expressiveness, Composability, and Maturity) for systematically comparing EIPs, industry projects, and academic proposals. (iii) We provide a cross-cutting synthesis (Section 7) that jointly analyzes 70 EIPs/ERCs, ABIM compliance across all layers, and 20 representative industry projects, exposing ecosystem-level (mis)alignments invisible to single-layer analysis. (iv) We move beyond the prevailing "blockchain as tool for AI" perspective to examine AI agents as active participants in consensus and governance, a dimension absent from prior surveys (Section 6). (v) We develop a threat taxonomy covering seven attack categories and identify nine concrete open research problems (Section 9). (vi) We

construct a three-dimensional taxonomy (agent autonomy × trust model × operational direction) that maps the ecosystem and highlights under-explored opportunities (Section 8).

Table 1 compares our coverage with four recent surveys. Ante [11] provides the most comprehensive project catalog (306 entries) and proposes a governance framework mapping autonomy against decentralization, but does not examine EIP/ERC standards, formal trust models, or the A → B direction. Karim *et al*. [1] address multi-agent security and scalability but do not analyze EIP/ERC standards; their subsequent survey [27] broadens the scope to the synergistic integration of decentralized and generative intelligence, organizing the field along a trust-to-augmentation progression, which is complementary to our treatment: it maps how generative AI capabilities augment decentralized systems, whereas we systematize the standards landscape, formalize the interaction model, and analyze trust in both directions. Romandini *et al.* [28] propose a four-layer software architecture and a three-class agent taxonomy, and offer the deepest treatment of security and privacy; however, their architecture describes how agent systems are built rather than what trust guarantees each layer provides, and they cover only the B → A direction (agents using blockchain), omit account abstraction and intent architectures entirely, and do not analyze EIP/ERC standards or formalize security properties. The closest work [29] provides the most detailed standards analysis with a five-part integration taxonomy and a comparative capability matrix of 85 systems across 13 dimensions, but does not cover the A → B direction, does not systematize zero-knowledge machine learning (zkML), optimistic machine learning (opML), or trusted execution environment (TEE) approaches, and does not propose a formal interaction model. Ours is, to our knowledge, the first survey to systematize both directions, evaluate all agent-relevant EIPs/ERCs through a unified five-dimensional framework, and formalize the interaction as ABIM.

Table 1. Comparison with related surveys. ● = comprehensive, ○ = partial, — = not covered.

Dimension	*[11]*	*[1]*	*[28]*	*[29]*	***Ours***
Account Abstraction (A1)	—	—	○	●	●
Permission (A2)	—	—	—	○	●
Intent Architecture (A3)	—	—	—	○	●
Agent Economy (A4)	●	—	—	○	●
AI + Security (B1)	—	○	●	—	●
AI + Consensus (B2)	—	○	—	—	●
AI + Governance (B3)	—	—	—	—	●
Verifiable Computation (TF)	—	—	○	○	●
Formal Model	—	—	—	—	●
EIP/ERC Analysis	—	—	—	●	●
Taxonomy	○	○	○	●	●
Threat Model	—	○	●	●	●

The remainder of the paper is organized as follows. Section 2 provides background and formal definitions. Section 3 reports corpus construction and screening. Section 4 examines the Trust Foundation layer of verifiable computation. Sections 5 and 6 cover the B → A and A → B directions, respectively. Having traversed the framework layer by layer, Section 7 then shifts to a cross-cutting view that synthesizes findings across all layers, analyzing privacy and cross-chain tensions, standards maturity and dependency structure, ABIM security-property compliance, and the industry project landscape, so as to expose

ecosystem-level patterns that single-layer analysis cannot reveal. Building on this synthesis, Section 8 develops the three-dimensional taxonomy, and Section 9 identifies open problems. Figure 1 illustrates the overall architecture.

2. Background and problem definition

2.1. Evolution of blockchain accounts toward autonomous agents

To understand how AI agents interact with on-chain systems, we need to trace blockchain account models through three distinct generations.

Since Ethereum's inception in 2015, the dominant account type has been the Externally Owned Account (EOA), controlled by a single secp256k1 private key. EOAs impose rigid constraints: they support only one signature scheme, cannot implement custom validation logic, and require the key holder to individually sign every transaction. These limitations make automated or delegated execution impossible at the protocol level, a serious obstacle for any agent that needs to act on a user's behalf. The concept of account abstraction (AA) relaxes these constraints by allowing smart contracts to serve as first-class accounts. Throughout this paper we annotate each standard with its EIP process status [30]: Draft, Review, Last Call, Final (stable specification), Stagnant (inactive $\geq$ 6 months), or Withdrawn; note that specification maturity does not imply adoption, as many Draft-stage standards are already deployed in production. The idea dates back to EIP-86 [31] (2017, now Stagnant), which proposed abstracting transaction origin and signature verification, and was revisited in EIP-2938 [32] (2020, Withdrawn), which aimed at native consensus-layer support. The current dominant standard, ERC-4337 [14] (2021, Final), takes a different path: it introduces an alternative mempool of UserOperations that are batched by Bundlers and executed through a singleton EntryPoint contract, all without requiring consensus-layer changes. Wang and Chen [33] provide a structured analysis of this architecture, while measurement studies [34] show that creating a SimpleAccount costs 381,489 gas, roughly 18 times a standard EOA transfer (21,000 gas). More recently, EIP-7702 [15] (2024, Final), introduced in the Pectra upgrade, allows EOAs to set contract code, blurring the boundary between the two account types. However, security analysis by Qi *et al.* [35] reveals that the interaction between EIP-7702 and ERC-4337 creates compound attack vectors not present in either standard alone.

The latest wave of standards moves beyond general account abstraction to address AI agent identity specifically. ERC-8004 [16] defines an agent discovery protocol with identity, reputation, and validation registries. ERC-8126 [17] specifies provider-operated verification for agents registered through ERC-8004, including five technical checks, privacy-oriented proof generation, and a quantitative risk score ranging from 0 to 100. ERC-7857 [24] takes a different approach, representing agents as NFTs whose model parameters and metadata can be encrypted using TEE and zero-knowledge proofs (ZKPs). Of these three standards, ERC-8126 and ERC-7857 are Final, while ERC-8004 remains Draft, so stable verification and private-metadata interfaces now coexist with a still-evolving discovery layer. Among the projects we surveyed, PAN Network [36] is one of the few that explicitly implements ERC-8004, combining it with an HTTP-native x402 payment protocol to enable agent-to-agent service discovery, price negotiation, and settlement. DeAgentAI [37] addresses the same triad of challenges, namely identity, consensus for

probabilistic outputs, and state continuity, through sovereign identity on Sui and BNB Smart Chain (BSC). Additional standards address agent consent and interoperability: ERC-7517 [38] defines metadata fields for declaring whether an agent's outputs may be used for AI/ML training, ERC-8033 [39] introduces a multi-agent council oracle model with commit-reveal and staking incentives, and ERC-7913 [40] enables signature verification for non-secp256k1 keys (including TEE-generated keys), broadening the authentication mechanisms available to agents. Recent academic work further explores unified identity delegation frameworks [41], authentication based on DIDs and verifiable credentials [42], and the intersection of proof-of-personhood with ZKP for AI alignment [43]. Figure 2 visualizes this three-generation evolution.

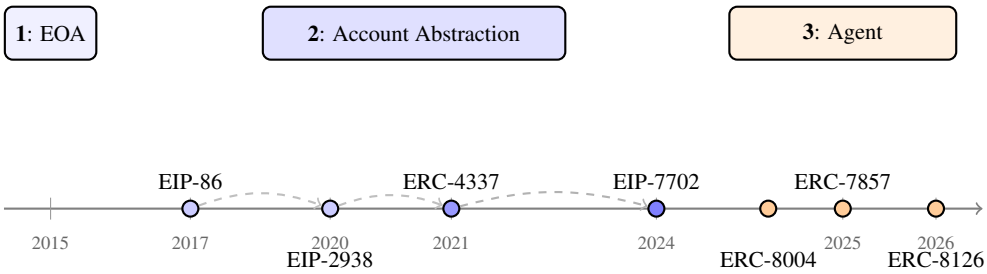

Figure 2. Evolution of blockchain account models: EOA, account abstraction proposals, and agent-specific identity standards.

2.2. From rule-based to LLM-agent: the evolution of AI autonomy

The development of AI techniques applied to blockchain can be broadly characterized by three overlapping phases, a pattern most clearly documented in the smart contract security literature [44,45]. Early on-chain automation relied on deterministic rule-based scripts, which were fully verifiable but extremely limited in scope. Subsequently, ML-assisted systems introduced learned models for tasks such as vulnerability detection and anomaly identification, though humans remained in the execution loop. Most recently, LLM-powered autonomous agents have begun to independently interpret, plan, and execute multi-step operations, handling everything from decentralized finance (DeFi) trading to governance participation [1].

A crucial observation runs through this progression: each generation increases expressiveness while decreasing verifiability. A rule-based bot can be formally verified against its specification; a 13B-parameter language model cannot, at least not with current proof systems (Section 4.1). This tension between expressiveness and verifiability is the central challenge motivating our trust framework.

2.3. Formal model: Agent-Blockchain Interaction Model

We define the Agent-Blockchain Interaction Model to state security properties across the framework. It provides a shared formal vocabulary for each layer's required guarantees and makes their underlying assumptions explicit. We first define the model's components, then state the security properties that each layer must satisfy.

2.3.1. Model definition

ABIM is defined as a six-tuple:

$$\text{ABIM} = \langle \mathcal{A}, \mathcal{B}, \mathcal{V}, \mathcal{P}, \mathcal{I}, \mathcal{E} \rangle \tag{1}$$

The components are defined as follows.

Agent Set $\mathcal{A}$. $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ is a finite set of agents. Each agent a_i is a five-tuple:

$$a_i = (id_i,\ keys_i,\ caps_i,\ M_i,\ \alpha_i) \tag{2}$$

where $id_i \in \mathbb{ID}$ is a unique on-chain identifier (e.g., an ERC-4337 account address), $keys_i \subseteq \mathbb{K}$ is a set of cryptographic keys the agent controls, $caps_i \subseteq \mathbb{C}$ is the set of declared capabilities (e.g., "DeFi trading," "governance voting"), M_i is a reference to the agent's underlying AI model (which may be opaque), and $\alpha_i \in \{\text{ASSISTIVE}, \text{SEMI-AUTO}, \text{AUTO}\}$ denotes the agent's autonomy level.

Blockchain State Machine $\mathcal{B}$. $\mathcal{B} = (S, s_0, T, \delta)$ is a deterministic state machine, where S is the set of all valid blockchain states, $s_0 \in S$ is the genesis state, T is the set of all well-formed transactions, and $\delta : S \times T \rightharpoonup S$ is a partial state transition function that maps a current state $s \in S$ and a valid transaction $t \in T$ to a successor state $s' = \delta(s,t)$, or is undefined ($\perp$) if t is invalid in state s. Agents interact with $\mathcal{B}$ exclusively by submitting transactions; the model does not introduce a separate operation-level abstraction, since all agent actions must ultimately be encoded as transactions to produce on-chain state transitions.

Verification Function $\mathcal{V}$. $\mathcal{V} : T \times \Pi \to \{\text{accept}, \text{reject}\}$ determines whether a transaction is accompanied by a valid proof of correctness. Here Π is the space of all proofs, which includes cryptographic proofs (e.g., zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs)), fraud proofs (with associated challenge periods), hardware attestations (e.g., TEE remote attestation), and the null proof $\pi_\emptyset$ (no verification). The strength of the guarantee depends on the proof type, as formalized by the Trust Foundation properties below.

Permission Policy $\mathcal{P}$. $\mathcal{P} : \mathcal{A} \times T \times S \to \{\text{allow}, \text{deny}\}$ is a state-dependent permission function that determines whether agent a is authorized to submit transaction t in state s. The state dependence captures dynamic constraints such as session-key expiry, spending limits, and time-bounded delegation.

Intent Function $\mathcal{I}$. $\mathcal{I} : NL \to 2^{Constraint}$ maps a natural language specification $nl \in NL$, where NL denotes the natural-language input space, to a set of formal constraints $C = \{c_1, c_2, \ldots\}$, where each constraint c_j is a predicate over post-execution states: $c_j : S \to \{\text{true}, \text{false}\}$. A solver $solv : 2^{Constraint} \times S \to T^*$ produces a transaction sequence $\bar{t} = (t_1, \ldots, t_k)$ that, when executed from state s, is intended to reach a state satisfying all constraints.

Economic Incentive Function $\mathcal{E}$. $\mathcal{E} : \mathcal{A} \times T \times S \times S \to \mathbb{R}$ assigns a real-valued reward (or penalty, if negative) to agent a for submitting transaction t that transitions the blockchain from state s to state s'. This function captures staking rewards, slashing penalties, gas refunds, revenue-sharing distributions, and solver fees.

2.3.2. Security properties by layer

Using the components defined above, we state the security property that each framework layer must satisfy. These properties are formal specifications for evaluating standards, protocols, and implementations. Current systems do not necessarily satisfy them; the difference between the required properties and current capabilities motivates the open problems identified throughout this paper.

Trust Foundation: Verification Soundness. The verification function must be sound: if it accepts a transaction with proof π, the transaction's claimed output must be correct.

$$\mathcal{V}(t,\pi) = \text{accept} \implies output(t) = claimed(t) \tag{3}$$

The strength of this guarantee varies by proof type: cryptographic proofs (zkML) provide computational soundness under standard cryptographic assumptions; fraud proofs (opML) provide game-theoretic soundness assuming ≥ 1 honest challenger within the dispute window Δ; hardware attestation (TEE) provides soundness conditional on the manufacturer's honesty.

A1 (Identity): Uniqueness. No two distinct agents may share the same on-chain identifier.

$$\forall a_i, a_j \in \mathcal{A} : a_i \neq a_j \implies id_i \neq id_j \tag{4}$$

Non-forgeability, the requirement that no adversary can produce valid transactions under id_i without controlling $keys_i$, is inherited from the underlying cryptographic signature scheme and is not restated here.

A2 (Permission): Enforcement Completeness. If the permission policy denies a transaction, the protocol must ensure that the transaction does not produce a state transition. This must hold even when the agent composes multiple individually permitted transactions.

$$\forall a \in \mathcal{A}, t \in T, s \in S : \mathcal{P}(a,t,s) = \text{deny} \implies \delta(s,t) = \perp \tag{5}$$

A3 (Intent): Solver Faithfulness. For any intent with constraint set C and solver-produced transaction sequence $\bar{t}$, executing $\bar{t}$ from the current state s must reach a state s' that satisfies every constraint.

$$\forall C = \mathcal{I}(nl),\ \bar{t} = solv(C,s) : \bigwedge_{c \in C} c\big(\delta^*(s,\bar{t})\big) = \text{true} \tag{6}$$

where $\delta^*(s,\bar{t}) = \delta(\cdots\delta(\delta(s,t_1),t_2)\cdots,t_k)$ denotes sequential execution of the transaction sequence.

A4 (Economy): Incentive Compatibility. The economic function must be incentive-compatible: an agent maximizing its expected reward should submit transactions that are beneficial to the system (or at minimum, non-harmful).

$$\forall a \in \mathcal{A} : \arg\max_t \mathbb{E}_{t_{-a}}[\mathcal{E}(a,t,s,s')] \in T_{\text{beneficial}} \tag{7}$$

where $T_{\text{beneficial}} \subseteq T$ is the set of transactions that satisfy application-specific utility criteria (e.g., faithful model inference, honest governance voting), and the expectation $\mathbb{E}_{t_{-a}}$ is taken over the concurrent transactions of other agents, which determine the realized state $s' = \delta^*(s,\bar{t})$.

B1 (Security): Detection Completeness. An agent serving as a security auditor should detect all vulnerabilities in the target contract within the scope of its declared capabilities.

$$\forall v \in Vuln(s) \cap scope(a.caps) : \Pr[detect(a,v) = \text{true}] \geq 1 - \varepsilon \tag{8}$$

where $Vuln(s)$ is the set of true vulnerabilities in state s, and ε is the model's inherent false-negative rate. In practice, current systems achieve ε values ranging from 0.02 (LLM-SmartAudit on benchmarks) to 0.40 (independent evaluations), as detailed in Section 6.1.

B2 (Consensus): Validator Verifiability. Every AI validator's consensus decision must be accompanied by a proof that the verification function accepts. Let $Val \subseteq \mathcal{A}$ denote the set of agents participating

as validators. A validator's attestation is modeled as a transaction $t_a^{\text{att}} \in T$ that encodes its decision $dec_a \in \{\text{commit}, \text{abort}\}$ and the proposed block, with accompanying proof $\pi_a \in \Pi$.

$$\forall a \in Val : \mathcal{V}(t_a^{\text{att}}, \pi_a) = \text{accept} \tag{9}$$

This property is in tension with the probabilistic nature of AI models: an honest AI validator with error rate ε_a may produce incorrect decisions with probability ε_a, so the classical BFT assumption of deterministic correctness must be replaced by a probabilistic bound (Section 6.2).

B3 (Governance): Bounded Influence. The aggregate voting power of AI agents must not exceed a fraction τ of total voting power, ensuring that governance outcomes reflect the preferences of human stakeholders.

$$\sum_{a \in \mathcal{A}_{\text{AI}}} w(a,s) \leq \tau \cdot \sum_{p \in \mathcal{A}_{\text{all}}} w(p,s) \tag{10}$$

where $w(a,s)$ is the voting weight of participant a in state s (which may be token-weighted, quadratic, or delegated), $\mathcal{A}_{\text{AI}} \subseteq \mathcal{A}$ is the set of AI agent voters, and $\tau \in (0,1)$ is a governance parameter. No current DAO enforces such a bound, as discussed in Section 6.3.

2.3.3. Model properties and limitations

The ABIM formalization captures the essential structure of agent-blockchain interactions with several deliberate abstractions. First, it models blockchain state transitions as deterministic and sequential, omitting concurrency and mempool dynamics (which are relevant to MEV, addressed in Section 5.3). Second, the intent function $\mathcal{I}$ is assumed to produce well-formed constraint sets, whereas in practice, LLM-based intent parsing may produce ambiguous or contradictory constraints. Third, the model treats each agent as a single entity, whereas real agents may operate as multi-agent systems internally (as in LLM-SmartAudit's multi-agent auditing pipeline, Section 6.1). Fourth, the six-tuple is structurally biased toward the B $\rightarrow$ A direction: $\mathcal{V}$, $\mathcal{P}$, $\mathcal{I}$, and $\mathcal{E}$ each map directly to a B $\rightarrow$ A layer, whereas the A $\rightarrow$ B properties (Equations (8)–(10)) introduce auxiliary notation ($Vuln(s)$, $detect()$, Val, $w(a,s)$) that is not grounded in the tuple. This asymmetry reflects the current state of the field: the B $\rightarrow$ A direction has mature infrastructure that admits compact formal characterization, while the A $\rightarrow$ B direction remains exploratory and resists premature unification.

The eight security properties stated above are of different types. Properties 4 and 5 are protocol invariants that can, in principle, be enforced on-chain. Properties 3, 8 and 9 are assurance bounds whose strength depends on cryptographic assumptions or probabilistic error rates ε. Properties 6, 7 and 10 are mechanism design objectives that can only be evaluated through equilibrium analysis, not enforced by protocol logic alone. This three-way distinction matters when assessing gap severity in later sections: an unmet invariant is an engineering deficit, an unmet assurance bound calls for stronger proof systems, and an unmet mechanism objective requires new theoretical frameworks.

Even with these simplifications, the model retains enough expressive power to state the security properties that distinguish our framework layers and identify where current systems fall short of the stated guarantees.

2.4. *Five-dimensional evaluation framework*

To evaluate EIPs, ERCs, academic proposals, and industry projects on a common basis, we define a five-dimensional evaluation framework. Each dimension captures a distinct property essential for reasoning about the suitability of agent–blockchain infrastructure. Together, they span the design trade-off space from cryptographic soundness to real-world deployability.

- D1. Verifiability measures the strength of correctness guarantees. We distinguish four levels: High (cryptographic proof, e.g. zkML or ZKP-based verification, where correctness follows from mathematical assumptions), Medium (economic or game-theoretic guarantee, e.g. fraud proofs with staked collateral, where correctness holds under incentive-compatibility assumptions), Low (reputation or attestation only, e.g. a trusted registry or hardware attestation, where correctness depends on the honesty of a designated party), and None (no on-chain verification mechanism).
- D2. Minimality of Trust counts the minimum number of trusted entities required beyond the blockchain protocol itself for the evaluated functionality. A score of 0 means fully trustless (only cryptographic assumptions); 1 means one trusted role or entity set (e.g. a TEE manufacturer or a single honest challenger in optimistic verification); 2 means two independently required roles or entity sets; and 3+ means at least three. Lower is better; this dimension captures how well the standard or system aligns with blockchain's trust-minimization ethos.
- D3. Expressiveness measures the range of agent behaviours or use cases that the standard can support. We rate this as Very High (an open-ended goal interface that extends across domains or chains, or accepts goal-level input such as natural language), High (an extensible interface supporting a broad class of operations within one domain), Medium (a bounded, parameterized family of operations), or Low (one narrowly defined, fixed-schema operation).
- D4. Composability measures how easily a standard can be combined with other standards or integrated into existing ecosystems. We rate this as Very High (reuses established primitives and works with existing applications without per-application changes), High (defines clean reusable interfaces requiring only modest shared infrastructure), Medium (requires a domain-specific adapter), or Low (uses closed, proprietary, or hardware-specific interfaces, or still depends on an unadopted protocol change).
- D5. Maturity reflects the governance status of the standard as defined in the EIP process (Section 2.1): Final (adopted, stable specification), Review (under formal peer review), Draft (initial proposal, interfaces may change), or Stagnant/Withdrawn (inactive or retracted). For academic proposals and industry projects that do not follow the EIP process, we map their development stage to the closest equivalent.

The first four dimensions capture inherent design properties that involve fundamental trade-offs: for example, stronger Verifiability typically comes at the cost of lower Expressiveness (proving correctness limits the complexity of supported operations), and higher Trust Minimality is often achieved by sacrificing Composability (trustless designs may require custom interfaces incompatible with existing infrastructure). Maturity is an orthogonal, temporal dimension that tracks progress toward stable specification rather than a design trade-off.

Scoring procedure. In the evaluation tables throughout Sections 4–6, each standard is scored against Table 2 using its specification text, reference implementations, and (where available) third-party audits. For Minimality of Trust, one score unit denotes one independently required trusted role or entity set; substitutable actors serving the same role count once, optional roles are excluded unless their functionality is the evaluated path, and counts of three or more are collapsed into 3+. An entity that can delay or censor but cannot forge is counted only when liveness or censorship resistance is part of the evaluated functionality. ERC-4337 therefore counts one bundler role for alternative-mempool liveness (Trust = 1), while its optional paymaster is excluded from the core execution path. Equal numerical counts can still differ qualitatively: reliance on a fixed, unsubstitutable party (e.g., a hardware manufacturer) is a stronger assumption than reliance on any honest member of an open set (e.g., an optimistic challenger).

Table 2. Quick-reference decision rules for the three judgment-based dimensions.

Dimension	Primary Decision Rule
Trust	Count independently required trusted roles or entity sets beyond the blockchain protocol: 0, 1, 2, or 3+.
Expressiveness	Classify the interface scope: one fixed-schema operation (Low); a bounded operation family (Medium); an extensible interface within one domain (High); or an open-ended goal interface across domains or chains (Very High).
Composability	Classify integration friction: closed or dependent on an unadopted protocol change (Low); domain-specific adapter (Medium); reusable interface with modest shared infrastructure (High); or established primitives with no per-application changes (Very High).

We apply three further boundary rules to keep the categorical scores consistent. First, if a standard supports but does not require a stronger verification mode (e.g., an optional proof field), Verifiability follows its default configuration. Second, if verification is mandatory but several proof backends are standardized (e.g., ERC-7857's TEE-or-ZKP interface), the strongest standardized backend determines the table entry and weaker deployments are noted in prose. Third, Composability is assessed at the integration surface available during the evaluation period: a finalized protocol feature such as EIP-7702 is not penalized merely because its adoption required a consensus change, whereas a proposal that still requires a new protocol change to compose is Low. Borderline cases are assigned according to the standard's stated scope rather than its roadmap.

The resulting thresholds match the evaluated standards. ERC-4337 is High rather than Very High on Expressiveness because arbitrary calldata under programmable validation remains within the account-execution domain; goal-level or cross-chain semantics require an intent standard. ERC-7683 and ERC-7845 reach Very High through different qualifying paths, respectively a generic cross-chain order interface and chain-abstracted goal requests that can originate in natural language. ERC-8004 is High rather than Very High on Composability because its reusable interfaces require new registry infrastructure, whereas ERC-6551 works with existing ERC-721 tokens and applications without per-application changes.

Relation to ABIM. The five dimensions are not independent of the formal model: each of the first four operationalizes, at the granularity of a ratable standard, a facet of the ABIM security properties, while Maturity tracks specification stability, with deployment-level enforcement of the properties assessed separately in Section 7.4. Table 3 makes this mapping explicit. Concretely, a standard's Verifiability level describes which class of proof $\pi \in \Pi$ its verification path admits, and therefore how strongly Verification

Soundness (Equation (3)) and Validator Verifiability (Equation (9)) hold when the standard is used; Minimality of Trust counts the trusted roles or entity sets assumed by its proof class together with operational parties outside the verification path, such as registry maintainers or oracle committees; Expressiveness measures the richness of the transaction and constraint spaces the standard exposes to $\mathcal{P}$ and $\mathcal{I}$ (Equations (5) and (6)); and Composability asks whether the protocol invariants (Equations (4) and (5)) are preserved when the standard is combined with others. The anchoring is deliberately many-to-many and approximate: Equation (5) informs both Expressiveness and Composability, and the remaining properties (Equations (7), (8) and (10)) have no dimensional counterpart, since they are mechanism-design or assurance properties of deployed systems rather than of specifications, and are assessed directly in Section 7.4.

Table 3. Mapping between the five evaluation dimensions and ABIM security properties.

Dimension	**ABIM Anchor**	**What the Score Expresses**
Verifiability	$\mathcal{V}$; Equation (3), Equation (9)	Proof class π admitted; strength of soundness
Minimality of Trust	Equation (3) assumptions + operational parties	Number of honest parties assumed beyond the protocol
Expressiveness	$\mathcal{P}$, $\mathcal{I}$; Equations (5) and (6)	Richness of transaction/constraint space
Composability	Invariants Equations (4) and (5)	Integration without breaking invariants
Maturity	Orthogonal (see Section 7.4)	Specification stability (EIP status; development stage otherwise)

2.5. Threat model

We identify seven categories of threats to agent–blockchain interactions (Table 4). Three of them, agent impersonation, privilege escalation, and intent deception, target the B $\rightarrow$ A layers and exploit the gap between what agents are authorized to do and what they actually do. Two others, inference tampering and model backdoors, target the Trust Foundation, undermining the correctness guarantees that all other layers depend on. Solver collusion and agent collusion target multi-party interactions where individually rational behaviour can produce collectively harmful outcomes. The "Gap" column in Table 4 highlights how many of these threats remain unaddressed by current standards and research.

Table 4. Threat model for agent-blockchain interactions.

Threat	**Capability**	**Layer**	**Gap**
Agent Impersonation	Forge identity	A1, B1	No Sybil-resistant agent identifier
Privilege Escalation	Exceed scope	A2	No formal verification tools
Intent Deception	Unfaithful solver	A3	Immature intent verification
Inference Tampering	Wrong AI result	TF	LLM verification infeasible
Solver Collusion	MEV extraction	A3	Insufficient theory
Agent Collusion	Manipulate governance	B3	Completely unexplored
Model Backdoor	LLM backdoor	TF, B2	Completely unexplored

3. Survey methodology

We constructed the literature corpus in February 2026. The systematic search covered publications from 2017 through February 2026, while a small number of foundational earlier works, including capability-based security and the Fischer–Lynch–Paterson (FLP) impossibility result, entered the corpus through backward snowballing. We searched Google Scholar, arXiv, IEEE Xplore, ACM Digital Library, Semantic Scholar, IACR Cryptology ePrint Archive, SSRN, ScienceDirect, and Frontiers. Queries were organized by framework layer using one keyword group per layer; for A1, for example, the terms included account abstraction ERC-4337, EIP-7702 security analysis, decentralized identity AI agent, and proof of personhood sybil resistance. Candidate lists were expanded through backward snowballing from core papers and recent surveys and forward snowballing through citation indices. Standards were enumerated directly from the Ethereum EIP/ERC repositories, and their process statuses and specification-dependent evaluations were re-verified on July 18, 2026; industry project data were collected separately from RootData and CoinMarketCap using a February 2026 snapshot.

We then screened each candidate against four inclusion criteria. It had to address the intersection of AI agents and blockchain rather than either field alone, and map to at least one framework layer or provide essential context. Eligible source types were peer-reviewed publications, preprints, standard documents, and authoritative technical reports. Each candidate also had to contribute a mechanism, formal analysis, measurement, or systematization. We excluded single-field work, material without technical substance, and superseded versions; project documentation was retained only as industry-landscape evidence, not as academic evidence. Applying these criteria to the February 2026 search yielded 122 academic publications, 70 EIP/ERC standards, and 20 industry projects, with each source assigned to the relevant framework layer during screening. Five additional academic references were incorporated during the July 2026 revision cycle, including work outside the original search window, bringing the final academic corpus to 127 publications.

4. Trust foundation: verifiable computation

Both the B $\rightarrow$ A and A $\rightarrow$ B directions involve a central question: how can one verify that an AI agent's reasoning and outputs are correct without re-executing the computation? The verification soundness property in Equation (3) states the requirement formally: if the verification function $\mathcal{V}$ accepts an action with proof π, the action's output must be correct. Cryptographic verification is not required for every agent interaction. For example, a read-only analytics agent or a low-value assistive bot may operate under weaker trust models. Verifiable computation nevertheless becomes increasingly critical as agent autonomy and transaction stakes increase.

Three families of techniques have emerged, which we organize by decreasing trust minimality (or equivalently, increasing practical deployability). First, zkML (Section 4.1) achieves the strongest guarantee, cryptographic proofs that require zero trust assumptions beyond mathematics, but at the cost of extreme computational overhead that limits the scale of models it can verify. Second, opML (Section 4.2) relaxes the requirement to a game-theoretic guarantee, assuming at least one honest challenger exists, in exchange for practical scalability to LLM-scale models. Third, TEE (Section 4.3) shifts trust entirely to hardware

manufacturers, achieving the best performance and broadest model support at the cost of a centralized trust assumption. This ordering, from strongest-but-slowest to weakest-but-fastest, defines the fundamental design trade-off that agent system architects must navigate. We then provide a comparative analysis (Section 4.4) using our five-dimensional framework, followed by a discussion of the central open problem (Section 4.5).

4.1. zkML

The line of work on zkML aims to generate succinct, non-interactive proofs that a particular model produced a particular output on a particular input, without revealing either the model weights or the input data.

The earliest practical results targeted convolutional neural networks (CNNs). Liu *et al.* [46] showed that the sumcheck protocol can be adapted to verify CNN predictions in time linear in the circuit size (zkCNN), proving VGG-16 (15M parameters) in 88.3 seconds with 341 KB proofs. Around the same time, Weng *et al.* [47] demonstrated that quantized networks as deep as ResNet-101 (over 100 layers) can be handled through efficient conversion protocols for subfield Vector Oblivious Linear Evaluation (sVOLE)-based ZKPs (Mystique), achieving a 7× improvement in matrix multiplication proofs over prior work. These results established feasibility, but the models involved, image classifiers with millions of parameters, are orders of magnitude smaller than the language models that power contemporary agents.

Sun *et al.* [48] closed this gap by proposing zkLLM. It was the first system capable of proving inference for a 13B-parameter language model. Two novel primitives, tlookup for non-arithmetic operations and zkAttn for the attention mechanism, bring proving time for the entire inference process (sequence length 2048) down to under 15 minutes. This is a milestone, but the latency is still far too high for interactive agent scenarios requiring sub-second responses.

Complementary work has focused on tooling and compilation. The zkML compiler [49] achieves up to 5× faster proving and 22× smaller proofs compared to prior CNN-focused systems (zkCNN, vCNN), while extending support to production-scale models including GPT-2 and Twitter's recommendation pipeline. ZEN [50] pioneered rank-1 constraint system (R1CS)-friendly quantization, achieving 5.4–22.2× constraint reduction (15.4× average). zkPyTorch [51] bridges PyTorch directly to ZKP engines, proving VGG-16 inference in 2.2 seconds per image and Llama-3 8B in 150 seconds per token. zkRNN [52] extends this line of work to recurrent architectures with polylogarithmic proof size and millisecond-scale verification. On the training side, Abbaszadeh *et al.* [53] introduced Kaizen for verifying deep neural network (DNN) training via Goldwasser–Kalai–Rothblum (GKR)-style interactive proofs with recursive proof composition (1.63 MB proofs, constant verifier cost), while Garg *et al.* [54] provided early experimental validation of training proofs for logistic regression with streaming-friendly memory usage. Two on-chain standards complement this body of work: ERC-7992 [18] standardizes ML model registration and zero-knowledge inference proof verification, and ERC-7007 [19] defines verifiable AI-generated content NFTs. Several surveys provide broader context for zkML, each examining the field from a complementary angle: Keršič *et al.* [55] benchmark two leading on-chain frameworks (EZKL and Orion) with end-to-end deployment experiments, Peng *et al.* [56] catalogue 27 zkML studies spanning training, inference, and testing, and Xing *et al.* [57] systematize 56 zero-knowledge-proof-based verifiable machine learning (ZKP-VML) schemes into three main technical routes (GKR/sumcheck, R1CS/circuit, and multi-party-computation (MPC)-based).

4.2. opML

Rather than proving correctness upfront, opML borrows the optimistic paradigm from rollups: inference results are assumed correct unless challenged within a dispute window. Conway *et al.* [58] showed that this approach can support 7B-parameter Large Language Model Meta AI (LLaMA) models running on standard personal computers without graphics processing unit acceleration. The trade-off is latency: results are not finalized until the challenge period expires. So *et al.* [59] combined the privacy guarantees of zkML with the efficiency of opML in a hybrid framework (opp/ai), suggesting that the two approaches are more complementary than competing.

4.3. TEE

TEEs, exemplified by Intel Software Guard Extensions (SGX) and ARM TrustZone, provide hardware-attested execution in an isolated enclave. Tramèr and Boneh [60] showed that inference can be split between a TEE enclave (for verification) and an untrusted graphics processing unit (for computation), achieving 6–20$\times$ throughput improvement for verifiable inference and 4–11$\times$ for private inference on VGG-16 and MobileNet. Subsequent work has tackled the primary bottleneck, SGX's limited Enclave Page Cache of roughly 90 MB, through memory-efficient inference designs: Truong *et al.* [61] reduce VGG-16 inference overhead from 26$\times$ to only 1.09$\times$ via Y-plane partitioning, while InferONNX [62] achieves a compact 46 MB runtime footprint with 1.5–4$\times$ faster inference via model partitioning. A survey by Chaudhuri *et al.* [63] provides a comprehensive overview of TEE implementations across central and graphics processing units, covering NVIDIA H100 confidential computing and related approaches. TEE is the approach most widely adopted in industry today (e.g., bAI Fund [64] executes agent fund operations entirely within TEE enclaves on the Morph Layer-2 (L2) network), but its trust assumption, that the hardware manufacturer's attestation is honest, sits uncomfortably with blockchain's trust-minimization ethos. An alternative trust paradigm, fully homomorphic encryption (FHE), enables computation on encrypted data without decryption; Mind Network [65] is among the few projects in our survey that adopt FHE as a core trust primitive through its HTTPZ protocol, providing quantum-resistant guarantees but with significant computational overhead.

4.4. Comparative analysis

Figure 3 visualizes the three approaches along our five dimensions (defined in Section 2.4). The scores are assigned as follows. Verifiability: zkML scores 5 (cryptographic, non-interactive proof), opML scores 3 (probabilistic, contingent on a challenger appearing), TEE scores 2 (hardware attestation, no cryptographic guarantee of computation correctness). Trust Minimality: zkML requires 0 trusted parties (only mathematical assumptions), opML requires 1 (at least one honest challenger), TEE requires 1 (the hardware manufacturer); opML nevertheless scores above TEE on this axis because its honest challenger can be any member of a permissionless set, whereas TEE's manufacturer is a fixed, unsubstitutable party. Expressiveness: TEE scores 5 (supports any model on commodity hardware), opML scores 4 (demonstrated on 7B-parameter LLMs [58]), zkML scores 2 (limited to small-to-medium models, with LLM-scale proofs requiring $\sim$15 minutes for a full inference pass (Section 4.1)). Composability: zkML scores 4 (standard proof formats integrate with on-chain verifiers via ERC-7992), opML scores 3 (challenge-period delays complicate

synchronous composition), TEE scores 2 (hardware-specific attestation formats limit portability). Maturity: TEE scores 5 (deployed in production, e.g., bAI Fund [64]), opML scores 3 (early deployments, e.g., opML on Optimism), zkML scores 2 (research-stage with limited production use).

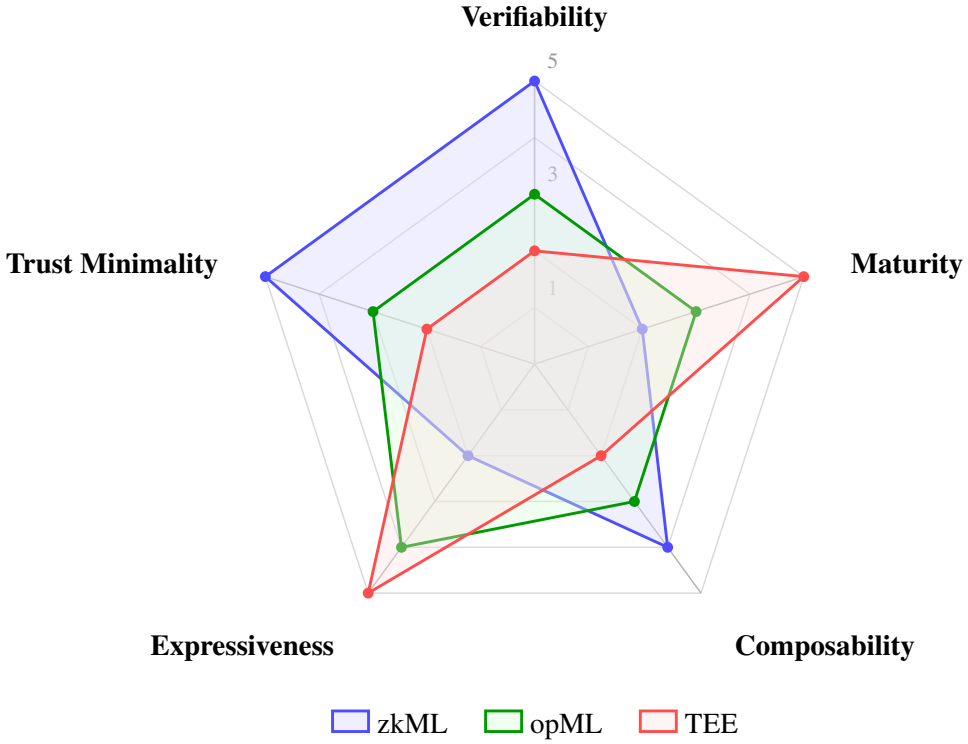

Figure 3. Five-dimensional radar chart comparing verification approaches. zkML achieves the highest verifiability and trust minimality but lags in expressiveness and maturity. TEE excels in expressiveness and maturity but requires hardware trust. opML provides a balanced middle ground. Scores range from 1 (worst) to 5 (best).

No clean winner emerges from the comparison; instead, one tension dominates. zkML and TEE occupy opposite ends of the trust-performance spectrum, with opML between them on every dimension. Thus, the application context determines which verification approach to choose, rather than any intrinsic superiority. Precisely because no approach dominates, hybrid architectures are attractive as the near-term path. In opp/ai [59], partitioning a model between zkML for privacy-sensitive submodels and opML for efficiency-sensitive submodels can balance privacy and computational cost. The same principle could extend to TEE–zkML combinations: TEE would handle latency-sensitive operations, while zkML would provide asynchronous auditability. The maturity gap between deployed TEE and research-stage zkML makes this situation precarious. Industry is converging on the weakest trust model by default. If TEE attestation is compromised, as has happened with Intel SGX side-channel attacks, the entire agent infrastructure built on it loses its security foundation. From an ABIM perspective, verification soundness (Equation (3)) is an assurance bound whose strength depends on the proof type: zkML provides computational soundness; opML provides game-theoretic soundness contingent on challenger availability; TEE provides hardware-conditional soundness. At LLM scale, no current approach satisfies Equation (3) with guarantees strong enough for high-stakes autonomous operations.

4.5. OP1: LLM verifiability bottleneck

We call this central open problem the LLM verifiability bottleneck. As shown in Section 4.1, the most capable zkML system to date requires roughly 15 minutes to prove a full inference pass for a 13B-parameter model, while agent-facing applications demand sub-second responses. Recent compiler advances narrow this gap: zkPyTorch [51] reduces Llama-3 8B proving to 150 seconds per token, and zkRNN [52] achieves second-scale proving for recurrent architectures, but orders-of-magnitude improvements are still needed. Peng *et al.* [56] survey 27 zkML studies and organize them into verifiable training, testing, and inference, while the end-to-end verifiable AI framework of [66] identifies a critical gap: no existing work connects proofs of training to proofs of inference into a single trust chain. Related systematization efforts include a SoK on verifiable federated learning [67] and a survey on privacy-preserving LLM inference [68], which identifies TEE-based solutions as deployable today while crypto-augmented designs (TEE + MPC/FHE) can reduce hardware trust in the medium term. Both efforts draw attention to the gap between theoretical feasibility and practical deployment. Promising directions include hierarchical proofs that verify only critical reasoning steps, approximate proofs that trade a small soundness gap for dramatic speedups, and hybrid TEE–ZKP architectures that use hardware attestation for latency-sensitive operations while generating cryptographic proofs asynchronously. Until this bottleneck is substantially reduced, the practical deployment of verifiable autonomous agents will remain constrained to the opML and TEE trust models.

5. B → A: Blockchain as trust infrastructure for agents

5.1. A1: Account and identity

Any agent on-chain first faces the question of how to establish a verifiable presence. Three progressively deeper sub-problems must be addressed. An account mechanism must support programmable validation and delegated signing, capabilities that legacy EOAs lack. With such an account in place, the agent must be discoverable and assessable by users and other agents; this calls for identity and reputation standards. The system must then ensure that each agent identity is unique and non-duplicable to prevent Sybil attacks at the agent layer. The logical progression is account infrastructure → identity semantics → uniqueness guarantees.

5.1.1. Account abstraction: the foundation

As described in Section 2.1, the evolution from EOA to ERC-4337 [14] to EIP-7702 [15] provides progressively more flexible account infrastructure for AI agents. Meta-transaction precursors such as ERC-2771 [69] first established the relayer/forwarder pattern that allows accounts without Ether to interact via gas-paying intermediaries, but lacked native protocol support and required each recipient contract to opt in by trusting specific forwarders; ERC-4337 generalizes this idea into a protocol-level account abstraction with bundlers and paymasters. This layer must enforce the identity uniqueness property stated in Equation (4). Here we focus on the security implications and supporting standards not covered in the background.

EIP-3074 [70], the predecessor to EIP-7702, introduced AUTH and AUTHCALL opcodes but was withdrawn due to security concerns around unrestricted delegation.

Gas figures provide a useful scale but not a direct comparison. Lin *et al.* [34] measured an average of 381,489 gas to create an ERC-4337 SimpleAccount on Goerli. EIP-7702 instead charges 25,000 gas per authorization tuple and grants a nominal 12,500-gas refund for an existing authority, subject to the transaction refund cap [15]. Because these values describe different operations and ERC-4337 costs vary with implementation and bundling, they do not establish an end-to-end cost ratio. EIP-7702 avoids contract deployment for an existing EOA, whereas ERC-4337 provides bundling and paymaster support.

Supporting infrastructure standards round out the A1 stack: ERC-7562 [71] restricts the Ethereum Virtual Machine (EVM) operations permitted during validation to prevent denial-of-service attacks on the mempool, EIP-5792 [72] and ERC-7821 [73] standardize batch call interfaces and minimal executor interfaces, the now-withdrawn ERC-7766 [74] proposed signature aggregation across multiple UserOperations to reduce gas costs, ERC-7836 [75] provides session-key-based call preparation, and ERC-7677 [76] defines a standardized Paymaster web service. Prior SoKs on cryptocurrency wallets [77] and wallet security evaluations [78], which found that on average 10.2% of on-chain wallet contracts are vulnerable, provide useful baselines for assessing the security of agent-facing account designs.

5.1.2. Agent identity standards

The three agent identity standards introduced in Section 2.1, namely ERC-8004 [16] (discovery), ERC-8126 [17] (provider-operated verification for ERC-8004-registered agents), and ERC-7857 [24] (agent-as-NFT with encrypted metadata), address complementary aspects of the identity problem. Beyond on-chain standards, academic work explores off-chain identity delegation: Saavedra [41] proposes a "Delegation Tetrahedron" extending self-sovereign identity with a Delegate role for cryptographically anchored authority transfer, while Rodriguez Garzon *et al.* [42] demonstrate DID/verifiable credential (VC)-based mutual authentication for LLM agents, though their evaluation reveals that agents occasionally agreed to skip authentication entirely, highlighting the fragility of delegating security-critical orchestration to LLMs.

5.1.3. Sybil resistance for agents

Proof-of-Personhood protocols [79,80] were designed to ensure one-person-one-account guarantees, but extending them to AI agents raises a different set of questions: an agent's "uniqueness" must be defined in terms of its model, operator, and declared capabilities rather than biological identity. ERC-4361 [81] provides Ethereum-based sign-in for off-chain services, and ERC-5573 [82] extends this with capability-based ReCaps, but neither was designed with non-human agents in mind.

5.1.4. Synthesis and evaluation

Table 5 now shows mixed maturity rather than a simple account–identity split: ERC-4337, EIP-7702, ERC-8126, and ERC-7857 are Final, while ERC-8004 remains Draft. ERC-8126 directly depends on ERC-8004, strengthening composability between discovery and verification. Its core checks nevertheless remain provider-operated and off-chain; proof generation can support independent validation, but the standard does not require an on-chain proof to gate a state transition. The main gap therefore remains Sybil-resistant uniqueness and broad adoption.

Trust scores count only the minimum external role required by the evaluated path: one bundler role for ERC-4337, none for self-sponsored EIP-7702, and one selected validator, verification provider, or sealed-execution service for ERC-8004, ERC-8126, and ERC-7857, respectively.

The gap is that the A1 layer currently provides account and registration interfaces, but not the Sybil-resistant uniqueness needed for agents to operate as first-class on-chain entities. Until a Sybil-resistant discovery and registration standard reaches Final status with broad adoption, agent discovery will remain ad hoc and platform-specific, limiting the composability that the B $\rightarrow$ A direction requires. In ABIM terms, identity uniqueness (Equation (4)) is a protocol invariant that should be enforced on-chain. ERC-8126 adds provider-operated verification to ERC-8004-registered agents, but its security considerations retain provider-trust and Sybil assumptions; neither standard prevents an operator from creating multiple identities. This gap propagates downstream: bounded influence (Equation (10)) in governance cannot be enforced if identity uniqueness is not assured.

Table 5. Five-dimensional evaluation of A1 (account & identity) standards.

Standard	Verifiability	Trust	Expressiveness	Composability	Maturity
ERC-4337	Medium	1	High	High	Final
EIP-7702	Medium	0	High	High	Final
ERC-8004	Low	1	Medium	High	Draft
ERC-8126	Low	1	Medium	High	Final
ERC-7857	High	1	High	Medium	Final

5.1.5. OP9: Sybil-resistant agent identity

A stable, uniquely-identifying registration mechanism is only half of what Equation (4) demands. The harder half is Sybil resistance: preventing a single operator from spawning many agents that appear, to the protocol, as independent entities. Proof-of-Personhood protocols [79,80] address this for humans by anchoring uniqueness to biometric or social-graph proofs, but neither approach transfers to AI agents, whose "personhood" is not tied to any biological substrate and whose capabilities can be duplicated at near-zero marginal cost. The problem is to define an agent-appropriate notion of uniqueness (perhaps grounded in model fingerprints, operator attestations, or staked collateral) and to design a registration protocol that enforces it without centralizing trust in a single registrar. The downstream impact is broad: both governance bounds (Equation (10), targeted by OP3) and legal accountability (OP8) presuppose that AI voters can be counted as distinct entities, a guarantee that no current standard provides.

5.2. A2: permission and delegation

Once an agent has an on-chain identity (A1), we examine what it is permitted to do and how those permissions are enforced. We address this through three sub-problems that move from theory to practice. First, we review the theoretical foundations of capability-based access control, which provides the formal underpinning for reasoning about least-privilege and delegation. Second, we examine how these ideas are instantiated in modular smart contract account standards, which decompose account logic into composable, auditable modules. Third, we identify the gap between general-purpose delegation standards and the specific requirements of AI agent delegation, where the delegate is a probabilistic system rather than a

deterministic script. Following this progression from theory → general standards → agent-specific gap, we identify where existing infrastructure falls short.

5.2.1. Theoretical foundations: capability-based security

When an agent operates on behalf of a user, the central security question is how to bound what the agent is permitted to do. This layer must satisfy the enforcement completeness property of Equation (5): denied actions must be non-executable, even under composition. Capability-based access control (CapBAC), originating from Dennis and Van Horn [83], provides the natural theoretical foundation: rather than maintaining access control lists, capabilities are unforgeable tokens that grant the bearer specific permissions. Nakamura *et al.* [84] adapt this model to smart contracts, achieving more fine-grained delegation than prior work by allowing subjects to independently delegate individual action tokens to and from multiple parties. Xu *et al.* [85] introduce federated authorization delegation with BlendCAC for IoT scenarios, demonstrating a decentralized, scalable solution feasible for resource-constrained devices. Ali *et al.* [86] extend this to support both event-based and query-based delegation with SPIN model checking under linear temporal logic properties. Zhang *et al.* [87] propose a three-contract framework (access control, judge, and register contracts) with both static policy validation and dynamic behaviour monitoring. Gao *et al.* [88] address multi-hop delegation in the context of electronic health record sharing, using ciphertext-policy attribute-based proxy re-encryption (CP-ABRPE) with smart contracts that enforce a controllable maximum delegation depth at each hop. Sherazi *et al.* [89] tackle a complementary problem, emergency delegation, where standard authorization policies must be temporarily overridden based on contextual constraints. Zero-trust frameworks [90] further inform agent permission design, reporting 14% encryption efficiency improvements and 18% reduction in data access latency compared to traditional methods.

5.2.2. Modular smart contract accounts

Several modular account architectures compete in the current standards landscape. ERC-7579 [91] decomposes account functionality into Validator, Executor, and Fallback Handler modules that can be independently installed, upgraded, and audited. ERC-6900 [92] offers an alternative modular account architecture that decomposes functionality into Validation, Execution, and Hook modules, enabling features such as session keys and spending limits to be installed as composable modules rather than built into the account itself. ERC-7780 [93] pushes modularity further by separating Policy, Signer, and Stateless Validator modules, giving developers fine-grained control over which components handle which aspects of account security. To support these designs, ERC-7484 [94] provides a module registry that allows third parties to attest to the security of individual modules.

5.2.3. Agent-specific delegation

Despite the proliferation of modular account standards, few address agent delegation directly. ERC-7710 [22] is among the few that explicitly lists AI agents as a motivating use case. ERC-5639 [95] provides delegation at the wallet, contract, and token level, ERC-7741 [96] enables operator authorization through EIP-712 signatures supporting meta-transactions and cross-chain scenarios, while ERC-7715 [97]

defines application programming interfaces (APIs) for requesting permissions, but none addresses the specific trust model required when the delegate is a probabilistic AI system rather than a deterministic script.

Auditors identified two high-severity vulnerabilities in a Quantstamp audit [98]: session keys with ERC-20 spend limits could drain the modular smart contract account by exploiting pre-execution hook assumptions about unchanged state across multiple calls, and incorrect storage key derivation caused all session keys to share the same permissions rather than having individual scopes. This finding emphasizes that modular flexibility comes at the cost of implementation complexity, and that no formal verification tools currently exist for reasoning about the composition of agent permission policies.

Table 6 evaluates A2 standards along the five dimensions. Across these standards, universal immaturity is the most striking finding: every standard except ERC-5639 (Review) remains at Draft, and none has reached Final, making this among the least mature of the B $\rightarrow$ A standards layers (only A3, where no standard has passed Draft, ranks lower). Verifiability is uniformly Medium or Low, reflecting the absence of formal verification tools for reasoning about module composition, a gap directly illustrated by the Quantstamp audit findings. Expressiveness varies substantially: the modular account architectures (ERC-7579, ERC-6900, ERC-7780) reach High expressiveness, with ERC-7780's separate Policy, Signer, and Stateless Validator types primarily improving composability around permission verification rather than extending behaviour beyond the account-security domain, while ERC-7710, which lists AI agents as a motivating use case, offers only Medium expressiveness. The contrast is another instance of the same tension: standards designed for maximum flexibility were not designed with agents in mind, whereas the one standard that addresses agents does so narrowly. In ABIM terms, enforcement completeness (Equation (5)) requires that a denied transaction never produces a state transition, even under composition. The Quantstamp audit finding, a spending-limit bypass through specific call sequences, demonstrates a concrete violation of this invariant. Until formal verification tools can prove properties about composed module interactions, the A2 layer will remain the weakest link in the B $\rightarrow$ A trust chain.

Table 6. Five-dimensional evaluation of A2 (permission & delegation) standards.

Standard	Verifiability	Trust	Expressiveness	Composability	Maturity
ERC-7579	Medium	1	High	High	Draft
ERC-6900	Medium	1	High	Medium	Draft
ERC-7710	Low	1	Medium	High	Draft
ERC-7780	Medium	1	High	High	Draft
ERC-5639	Low	1	Medium	High	Review
ERC-7715	Low	1	Medium	Medium	Draft

5.2.4. OP2: formal verification of agent permission policies

Given a permission policy $\mathcal{P}$ and a set of declared agent capabilities, can one formally verify that the policy prevents privilege escalation and enforces least privilege? This question extends classical capability-based security theory to the setting of composable smart contract modules, where multiple independently developed modules interact through shared state. The Quantstamp audit finding, a spending limit bypass through specific call sequences, illustrates that informal reasoning about module composition

is insufficient. Promising directions include adapting SPIN model checking [86] to ERC-7579 module graphs, and developing type-level guarantees that prevent cross-module state interference.

5.3. A3: intent specification and execution

With identity (A1) and permissions (A2) in place, the agent must express what it wants to achieve and have that goal faithfully executed. We examine four aspects of this layer, following the lifecycle of an intent from concept to analysis. First, we introduce the intent paradigm itself, explaining how it differs from traditional transaction construction and why it is particularly well-suited to AI agents. Second, we survey the ERC standards that define the on-chain intent infrastructure. Third, we review the formal analysis of intent market structure and MEV dynamics, which reveals fundamental incentive challenges. Fourth, we identify the distinct roles AI agents play in the intent ecosystem as both generators and solvers. This structure moves from concept $\rightarrow$ implementation $\rightarrow$ theory $\rightarrow$ application, reflecting the current state of a paradigm that is deployed before it is fully understood.

5.3.1. The intent paradigm

The intent paradigm inverts the traditional model of transaction construction. Instead of specifying exact calldata, namely which operation to call, on which contract, and with which parameters, users declare desired outcomes, and a competitive market of solvers races to find execution paths that satisfy those constraints. The solver faithfulness property (Equation (6)) requires that any solver-produced execution must reach a state satisfying all original intent constraints. This inversion is particularly significant for AI agents: an LLM can naturally express a high-level goal ("swap token A for at least X of token B before deadline D") but would struggle to construct the optimal sequence of contract calls across liquidity pools, bridges, and aggregators. Nagesh [99] provides an industry analysis of the intent-centric thesis, identifying fragmented intent pools as a key barrier and AI as a critical enabler for handling intent complexity.

5.3.2. Intent standards

Four ERC standards currently define the on-chain intent infrastructure. ERC-7521 [20] introduces the UserIntent object and allows any solver to participate in combining and executing intents. ERC-7683 [21] extends this design to cross-chain operations, defining a "filler" network for settlement across domains. ERC-7806 [100] unifies the roles of relayer, paymaster, and bundler into a single solver entity (building on EIP-7702) with an open execution model where any solver can participate. ERC-7845 [101] standardizes a universal orchestrator endpoint that, among other use cases, can serve AI-driven systems by providing a structured JavaScript Object Notation-Remote Procedure Call (JSON-RPC) interface for intent submission.

5.3.3. Formal analysis of intent markets

Academic analysis of intent markets remains sparse. The only formal treatment we are aware of, by Chitra *et al.* [13], models execution costs as barriers to entry and proves that the equilibrium number of solver entrants k^* is asymptotically $O(\sqrt{n})$ for exponential price distributions and $O(n^{1/3})$ for uniform distributions, producing oligopolistic structures. In a related setting, Ma *et al.* [102] establish the existence of Nash equilibria in order flow auctions (OFAs) under proposer-builder separation

and show that the more capable builder earns disproportionately higher revenue, a driver of builder centralization. A sharper negative result comes from Bahrani, Garimidi, and Roughgarden [103], who prove that no non-trivial transaction fee mechanism can simultaneously satisfy dominant-strategy incentive compatibility (DSIC) for users and block-producer incentive compatibility when producers are active MEV extractors. Rasheed *et al.* [104] use Shapley values to achieve fair MEV redistribution, with polynomial-time computability when searcher valuations are additive.

Complementary work examines execution quality and mitigation. Bachu *et al.* [105] quantify that Dutch auction-based OFAs (UniswapX, 1inch Fusion) provide statistically significant price improvements averaging 4–5 basis points over on-chain-only routing, while Watts *et al.* [106] propose "failure costs" as a mechanism whose asymmetric payoff structures ultimately benefit users. RediSwap [107] contributes a provably DSIC, individually rational, and Sybil-proof MEV redistribution mechanism for constant-function market makers, and a parallel game-theoretic analysis [108] shows that searcher competition follows Bertrand-style dynamics, creating a prisoner's dilemma with empirically observed bid-to-value ratios of 0.889. The broader DeFi context is captured by Werner *et al.* [2], whose foundational SoK situates MEV within the decentralized finance landscape. Despite this progress, no work has analyzed the specific dynamics that arise when AI agents, rather than human traders, generate and solve intents.

5.3.4. AI agent roles

We identify two distinct roles that agents can play in the intent ecosystem. As intent generators, agents translate natural language goals into structured intent specifications via $\mathcal{I}$, relieving users of the need to understand protocol-specific semantics. As solvers, agents compete to find optimal execution paths, leveraging their ability to process large volumes of on-chain state and market data. In practice, Olas [109], the most technically mature agent infrastructure ($13.8M funding, 4000+ deployed agents across 8+ chains), coordinates agent instances via Tendermint BFT consensus running finite state machine applications, with Gnosis Safe multisig for on-chain execution. Alternative architectures include AgentFi's [110] "agent-as-NFT" design that embeds smart contract wallets within NFTs (inspired by ERC-6551), and Maiga's [111] hybrid approach combining the ElizaOS framework with centralized GPT-4o inference for DeFi trading on BNB Chain. Neither role has been systematically studied in the academic literature.

Table 7 reveals that all four intent standards remain at Draft, making A3 the only layer where no standard has progressed beyond the initial proposal stage. Expressiveness is the strongest dimension (High to Very High), reflecting the deliberate design goal of supporting flexible outcome specifications; ERC-7683 and ERC-7845 achieve Very High expressiveness by supporting cross-chain settlement and natural language input, respectively. However, Verifiability is uniformly Medium or Low: no standard provides cryptographic proof that a solver's execution faithfully satisfies the original intent, instead relying on economic incentives (staked collateral, competitive auctions) that create game-theoretic rather than mathematical guarantees. This gap is compounded by the formal analysis: oligopolistic solver markets [13] and the fundamental impossibility of simultaneously incentive-compatible fee mechanisms [103] suggest that the economic security assumptions underlying current intent architectures may be weaker than assumed. Solver faithfulness (Equation (6)) is a mechanism design objective in our ABIM classification: it cannot be enforced by protocol logic alone but must be approximated through

incentive alignment. The absence of any standard providing cryptographic proof that a solver's execution satisfies the original constraints means that Equation (6) currently rests entirely on economic penalties rather than mathematical guarantees.

Table 7. Functional and five-dimensional evaluation of A3 (intent & execution) standards.

Standard	Intent Type	Agent Role	Verifiability	Trust	Expressiveness	Composability	Maturity
ERC-7521	General	Generator/Solver	Medium	2	High	High	Draft
ERC-7683	Cross-chain	Solver	Medium	2	Very High	High	Draft
ERC-7806	EOA-based	Solver	Medium	1	High	High	Draft
ERC-7845	NL/voice	Generator	Low	2	Very High	Medium	Draft

5.3.5. OP6: end-to-end intent-to-proof pipeline

The sharpest research opportunity at this layer is making the full pipeline from natural language intent to on-chain execution verifiable. This would require composing several individually challenging components: natural language understanding (translating user goals to formal specifications, perhaps through languages like DAOLang [26]), solver execution (finding and executing optimal paths), and proof generation (certifying that the execution satisfied the original intent). No existing system addresses more than one of these steps.

5.3.6. OP5: inter-agent game theory

A second open problem concerns the multi-agent dynamics that intent markets create. When multiple AI agents compete on-chain, bidding in MEV auctions, solving intents, and voting in governance, standard game-theoretic analysis assumes rational players with known utility functions. AI agents introduce additional complexity: their strategies are learned rather than designed, their utility functions may be misaligned with their operators' intentions, and they can adapt their strategies in real time. Multi-agent reinforcement learning in blockchain environments and mechanism design for AI solver markets are two promising directions.

5.4. A4: agent economy and tokenization

The final B $\rightarrow$ A layer concerns the economic sustainability of agents, including their ownership, valuation, and incentives. We examine two facets. First, we survey the ERC standards that enable agent tokenization, ownership transfer, and revenue sharing. Second, we review the academic analysis that evaluates whether current tokenomic designs create genuine utility or primarily speculative instruments. This two-part structure reflects a fundamental question at this layer. The standards define how to tokenize agents, but the academic literature questions whether current implementations deliver on the promise of decentralized AI economics.

5.4.1. Agent tokenization standards

If agents can hold identities and execute transactions, the natural next question is whether they can also be owned, traded, and financially incentivized through on-chain mechanisms. This layer must satisfy the incentive compatibility property (Equation (7)): reward-maximizing agent behaviour should

align with system-beneficial outcomes. Several ERC standards address different facets of this problem. ERC-7662 [23] represents agents as NFTs, giving each agent a unique on-chain token that can be transferred or listed on existing marketplaces. ERC-7857 [24] (also evaluated in Section 5.1) extends agent NFTs with encrypted private metadata such as model weights. ERC-6551 [112] takes a different architectural approach, assigning smart contract accounts to NFTs so that the agent's token can itself hold assets, effectively giving agents their own wallets. For economic coordination, ERC-7641 [113] implements built-in revenue sharing for Initial Model Offerings (IMOs), where token holders receive a fraction of the agent's operating revenue, and ERC-7649 [114] provides bonding curve liquidity for AI model marketplaces, tying token price to demand.

5.4.2. The state of agent tokenomics

The academic literature on agent tokenization is still catching up with industry practice. Ante [11] provides the most comprehensive industry survey, analyzing 306 projects along a four-quadrant framework and finding that meme and sentiment-driven agents account for 44% of the ecosystem, while trading and analytics represent only 13%. A sharper counterpoint comes from Mafrur [115], who examines actual token utility and concludes that most AI token projects represent an "illusion of decentralized AI," with tokens functioning primarily as speculative instruments and most computation occurring off-chain, blockchain serving mainly as a coordination and payment layer. Our own industry analysis reinforces this finding: among 13 projects with listed tokens, Virtuals Protocol [116] alone captures $\sim$90% of total market capitalisation ($409M), while most application-layer tokens trade below $1M market cap. Virtuals' dominance stems from its Agent Commerce Protocol (ACP), which uses smart contract escrow for standardised agent-to-agent coordination and routes all agent token liquidity through the VIRTUAL base currency, creating network effects and deflationary pressure. Recall [117] ($42M funding) takes a contrasting route through Proof-of-Performance: users stake RECALL tokens and allocate Boosts to agents they predict will perform well, while agents compete in objective, verifiable challenges ranked by risk-adjusted metrics such as the Calmar ratio, creating an ungameable meritocracy for agent capability assessment.

On the theoretical side, Sockin and Xiong [118] develop the formal foundations of utility token economics, showing that tokenization serves as a commitment device preventing platform exploitation while governance limitations exist even with full commitment, providing tools directly applicable to agent contexts. Building on related foundations, Kivilo *et al.* [119] propose a conceptual goal model that decomposes token incentive structures into stakeholder-aligned objectives, and the subsequent Token Economy Design Method [120] extends this into a systematic design process covering incentive, governance, and token model dimensions. A few related applied studies complete this account: BorJigin *et al.* [121] explore AI-governed tokenization of alternative assets with integrated fraud detection; Zhang [122] proposes a blockchain-enabled decentralized AI data marketplace using a hybrid of proof-of-stake (PoS) and practical Byzantine fault tolerance (PBFT), achieving 4750 tx/s throughput and $> 99.8\%$ settlement accuracy; and Jackson [123] demonstrates blockchain-governed AI agents with under 3.2 s decision latency and $\sim$180K gas per enforcement event, connecting the economic layer to the identity and permission layers examined earlier. Table 8 evaluates the current agent economy standards.

Table 8 reveals a notable asymmetry between infrastructure readiness and economic substance. ERC-6551 (Review) and ERC-7857 (Final) demonstrate that the technical mechanisms for tokenizing agents are maturing; ERC-7857 achieves the highest Verifiability (High) in this layer through its proof-gated private-data transfer interface, which admits TEE or ZKP verifiers. Following boundary rule (ii) of Section 2.4, High reflects the ZKP path, although the reference code leaves that branch to implementers; TEE-only deployments provide weaker, attestation-grade guarantees. Trust = 1 counts the sealed-execution service required for authorized use of private metadata, not the on-chain verifier or a ZKP prover. ERC-6551 achieves the highest Composability (Very High) by reusing ERC-721 and ERC-1167 as building blocks. However, the economic coordination standards (ERC-7641, ERC-7649) remain at Draft with only Medium expressiveness, and the speculative dynamics and market concentration documented above suggest that the A4 layer's primary challenge is not engineering but mechanism design. Incentive compatibility (Equation (7)) requires that reward-maximizing agents produce beneficial outputs, yet current evidence [115] indicates the opposite: agents maximizing economic returns engage in token speculation rather than productive service provision, placing Equation (7) among the furthest-from-satisfied mechanism design objectives in our framework.

Table 8. Five-dimensional evaluation of A4 (agent economy) standards.

Standard	Function	Verifiability	Trust	Expressiveness	Composability	Maturity
ERC-7662	Agent NFT	Low	1	Medium	High	Draft
ERC-7857	Private NFT	High	1	High	Medium	Final
ERC-6551	Token-bound Account	Medium	1	High	Very High	Review
ERC-7641	Revenue Sharing	Medium	1	Medium	High	Draft
ERC-7649	Bonding Curve	Medium	1	Medium	Medium	Draft

5.4.3. OP7: token engineering for agent economics

What tokenomic structures best align incentives for agent marketplaces? The question extends utility token theory [118] to agent-specific contexts where value creation depends on model quality, inference cost, and user trust. Key sub-questions include the design of IMOs that avoid speculative dynamics, revenue-sharing mechanisms that incentivize model improvement rather than token speculation, and bonding curve designs that price agent services according to actual demand rather than hype cycles.

6. A → B: agents as participants in blockchain

While the B → A direction treats blockchain as infrastructure that agents consume, the A → B direction reverses the relationship: here, AI agents contribute to core blockchain mechanisms, specifically auditing smart contracts (B1, Equation (8)), participating in consensus (B2, Equation (9)), and shaping governance (B3, Equation (10)). Of the directions considered, this is the most novel, yet it has received the least systematic treatment in the existing literature.

6.1. B1: AI for blockchain security

This layer examines how AI contributes to blockchain security, the most developed area of the A → B direction. We organize the literature along a capability spectrum that progresses from narrow to broad,

and from tool to participant. First, we survey LLM-driven smart contract auditing, where language models are used as specialized vulnerability detectors. Second, we cover deep learning and anomaly detection methods that operate on structured code or transaction representations rather than natural language. Third, we examine the critical transition from tool to participant: the point at which AI systems cease to be invoked by human auditors and begin to autonomously monitor, flag, and respond to security threats. This progression tracks increasing agent autonomy and highlights the dual-use tension: the same capabilities that enable autonomous defence also enable autonomous exploitation.

6.1.1. LLM-driven smart contract auditing

The application of LLMs to smart contract vulnerability detection has advanced rapidly. Wei *et al.* [124] propose LLM-SmartAudit, a multi-agent conversation framework that assigns role-specialized agents (Project Manager, Auditor, Counselor, and Programming Expert) to collaboratively audit smart contracts through two operational modes: Broad Analysis for wide-spectrum scanning and Targeted Analysis with buffer-of-thought reasoning for focused vulnerability scenarios. In its best configuration (GPT-4o with Targeted Analysis), the system achieves 98% overall accuracy on a common-vulnerability benchmark covering ten vulnerability types and identifies 12 out of 13 tested Common Vulnerabilities and Exposures (CVE) entries. Liu *et al.* [125] take a different approach with PropertyGPT: rather than directly classifying vulnerabilities, their system uses LLM-powered retrieval-augmented generation (RAG) to automatically synthesize formal verification properties in a custom Property Specification Language (PSL), achieving 80% recall on a 90-property test set drawn from 623 collected Certora properties (the remaining 533 serve as the RAG knowledge base). This indirect approach detects 9 out of 13 known CVEs and identifies vulnerabilities in 17 out of 24 real-world attack incidents.

These results are encouraging but warrant careful interpretation. The picture of LLM reliability for auditing is more sobering in independent evaluations: the LLMBugScanner study [126] reports 60% top-5 detection accuracy on 108 CVE-labelled contracts (19% improvement over single-model baselines through ensemble methods). GPTLENS [127] finds that its two-stage pipeline improves top-1 contract-level accuracy from 38.5% to 76.9% on 13 CVE contracts, but trial-level accuracy remains at 59%. These limitations suggest that LLM-based auditing is best used as a complement to, rather than a replacement for, traditional static analysis and formal verification.

6.1.2. Deep learning and anomaly detection

Beyond LLMs, deep learning architectures designed for structured data have shown promise. SCVHunter [128] applies heterogeneous graph attention networks to capture the rich structural patterns in smart contract control flow and data flow graphs, achieving 93.7% accuracy on reentrancy detection and 85%–91% on other vulnerability types across 1200 labelled contracts. LightningCat [129] frames vulnerability detection as a sequence-classification task over source code representations, reaching 93.5% F1-score with Optimized-CodeBERT on the SolidiFI benchmark. A multimodal approach by Deng *et al.* [130] fuses three code modalities (source code, operation code, and control-flow graphs) through a stacking-decision method, reaching 89%–95% accuracy across four vulnerability types, while Huang *et al.* [131] apply multi-task learning with a shared attention-based text encoder and

task-specific CNN branches to detect multiple vulnerability types simultaneously. A recent empirical comparison [132] evaluates three LLMs (GPT-3.5-turbo, DeepSeek R1, LLaMA-3) under zero-shot, few-shot, and chain-of-thought prompting on Solidity contracts, finding that LLaMA-3 achieves 96% accuracy in chain-of-thought settings but performance varies significantly across vulnerability categories. For runtime security, BlockScan [45] uses a BERT-style transformer with masked language modeling to detect anomalous patterns within individual transactions (intra-transaction level), achieving superior accuracy and lower false-positive rates than rule-based, traditional ML, and GPT-4 baselines on both Ethereum and Solana transactions.

Complementary perspectives on this field are available in several surveys. Alsunaidi *et al.* [133] cover 108 ML-based detection methods through a structured taxonomy spanning graph neural networks, LLM, and ensemble approaches, while Ivanov *et al.* [134] take a wider scope by classifying 133 threat mitigation solutions, including static analysis, formal verification, and ML, within a five-dimensional taxonomy. Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA)-style methodology underpins two further reviews: Crisostomo *et al.* [135] examine 51 ML articles focused on Ethereum, and De Baets *et al.* [44] systematically review 88 articles on ML-applied vulnerability detection. Complementing these, Jiang *et al.* [136] catalogue ML techniques for smart contract security detection, Liu *et al.* [137] adopt a lifecycle perspective covering threats and mitigations from development through maintenance, and Wang *et al.* [138] catalogue 61 learning-based detection papers, concluding that code representation matters more than model selection. On runtime security specifically, Luo *et al.* [139] report that tree-based models excel in early-stage detection while graph-based methods gain prominence later for capturing complex transaction relationships, and Mounnan *et al.* [140] provide a complementary review of deep anomaly detection methods on blockchain.

6.1.3. From tool to participant

The literature surveyed above treats AI as a tool, a system that a human auditor invokes to assist their work. But AI agents could also serve as active participants in blockchain security: continuously monitoring deployed contracts, autonomously flagging suspicious transactions, and coordinating with other security agents to respond to attacks in real time. EVMbench [141] provides the first rigorous benchmark for evaluating this transition: testing 120 curated vulnerabilities from 40 real audit repositories across three modes (Detect, Patch, and Exploit), it shows that frontier AI agents can discover and exploit vulnerabilities end-to-end against live Ethereum instances, with performance improving substantially when given hints about vulnerability locations. The fundamental tension is the dual use of these capabilities: agents able to audit and patch contracts autonomously can also exploit them autonomously. How can one verify that a security agent itself acts correctly? This trust question returns us to the Trust Foundation discussed in Section 4.

Conversely, AI-assisted code generation may introduce a new attack surface. According to Moonwell's incident summary, in February 2026 the DeFi lending protocol suffered a $1.78M loss after an oracle configuration error caused cbETH to be priced at $1.12 rather than ∼$2200, triggering a cascade of liquidations [12]. A related pull request (PR #578) by a human developer carries a "Co-Authored-By: Claude" trailer [142], a default annotation appended by the Claude Code command-line tool indicating that AI assistance was used during development. Importantly, Moonwell's official post-mortem and governance

proposal (MIP-X43) attribute the root cause to an oracle misconfiguration error rather than to AI-generated code specifically, and the PR passed multiple review layers—including GitHub Copilot review, human review, and reportedly a third-party audit—all of which failed to catch the semantic bug. The episode therefore illustrates a multi-layer assurance failure rather than an AI-specific deficiency: when subtle semantic errors enter security-critical DeFi systems through any development pathway, routine review and testing may prove insufficient, reinforcing the need for formal verification and defense-in-depth.

A clear maturity gradient can be seen across these representative systems. LLM-driven auditing systems (LLM-SmartAudit, PropertyGPT) achieve the highest CVE detection rates but with significant hallucination risks, and independent evaluations (LLMBugScanner, GPTLENS) consistently report lower accuracy than developer-reported benchmarks. Structured deep learning methods (SCVHunter, BlockScan) avoid hallucination entirely but are limited to pre-defined vulnerability patterns. Most critically, EVMbench demonstrates that the transition from tool to participant is already technically feasible: frontier agents can discover and exploit vulnerabilities end-to-end. However, none of these systems addresses the fundamental question of who verifies the verifier: ensuring that an autonomous security agent is itself acting correctly requires the Trust Foundation mechanisms discussed in Section 4, creating a direct dependency between the B1 and TF layers. Detection completeness (Equation (8)) is an assurance bound parameterized by false-negative rate ε. The discrepancy between developer-reported benchmarks ($\varepsilon \approx 0.02$) and independent evaluations ($\varepsilon \approx 0.40$) reveals that the current ε is not only large but also unreliably estimated, undermining the practical utility of Equation (8) as a safety guarantee.

6.2. B2: AI for blockchain consensus

Consensus is the most fundamental blockchain mechanism; this layer asks whether AI can participate in it. We organize the literature by increasing integration depth. As an external optimizer, AI tunes consensus parameters but stays outside the consensus protocol. A direct participant, by contrast, embeds useful computation such as ML training in consensus, so the work securing the chain also produces productive AI outputs. At a deeper level lies the fundamental theoretical tension between classical BFT's deterministic assumptions and the probabilistic nature of AI models, which we call the BFT-AI Paradox. The progression remains optimize $\rightarrow$ participate $\rightarrow$ foundational challenge: deeper integration exposes deeper theoretical gaps.

6.2.1. AI optimizing consensus parameters

The simplest form of AI involvement in consensus is as an optimizer that tunes protocol parameters without altering the consensus mechanism itself. A recent survey by [143] catalogues machine learning approaches for consensus optimization, while Li *et al.* [144] survey reinforcement learning (RL) applications in strategic mining, summarizing known security thresholds (e.g., Bitcoin proof-of-work (PoW) $\sim$25% for selfish mining) and reviewing RL-based attack analyses on protocols including Ethereum's Casper Friendly Finality Gadget (FFG). A seminal example is SquirRL [25], which uses deep RL in a Markov-game framework to recover the optimal selfish-mining attack in Bitcoin and the Nash equilibrium in block withholding, and to uncover a novel attack on Ethereum's Casper FFG finalization mechanism in which a strategic adversary can amplify rewards by up to 30%. Several groups have applied reinforcement learning

to this problem. Adaptive consensus protocols trained with proximal policy optimization (PPO) [145] can detect malicious behavior, optimize validation paths, and dynamically modify consensus logic in response to network conditions, reducing average consensus latency by 34% under high-load conditions with a high detection rate (> 0.90) for Sybil and node-collapse scenarios, though detection drops to moderate levels (0.58–0.70) under congestion and erroneous-transaction scenarios. Multi-agent RL has been used to optimize PoS validator strategies [146] through iterative reputation refinement and penalty-reward mechanisms, and to achieve near-centralized-optimal consensus efficiency in IoT networks [147]. Hybrid approaches that combine ML classifiers with traditional consensus [148] have also shown promise for detecting malicious validators. At the network layer, Valko and Kudenko [149] apply PPO to optimize Ethereum block propagation order, achieving 1.7% sync time reduction and 5% message reduction. In all these cases, the AI system remains external to the consensus protocol: it recommends parameters, but the protocol's safety guarantees remain grounded in their original (non-AI) assumptions.

6.2.2. AI as consensus participants

A more radical approach embeds useful AI computation directly into the consensus mechanism. The validator verifiability property (Equation (9)) requires that every validator's consensus decision be accompanied by a proof that $\mathcal{V}$ accepts. Bravo-Marquez *et al.* [150] proposed WekaCoin, a Proof-of-Learning cryptocurrency in which suppliers publish ML competition tasks, trainers submit models to compete for rewards, and randomly selected validators evaluate submissions on held-out test data and propose new blocks, replacing hash puzzles with productive ML computation. PoDaS [151] extends this idea using CNN-based data science tasks, achieving 96.0% model accuracy compared to 88.9% (PoW) and 85.1% (PoS), with faster block generation. You [152] distributes RL and DNN training across validator nodes, unifying consensus and AI training into a single computational process. These approaches are appealing because they redirect the computational resources consumed by consensus toward productive work. They nevertheless pose a difficult verification question: how does the protocol determine whether a submitted model update is valid without re-executing the training?

6.2.3. The BFT-AI paradox

This question points to a fundamental tension. Classical BFT state-machine replication relies on honest replicas applying a deterministic validation or state-transition rule, or otherwise coordinating any nondeterminism. AI-based validation, by contrast, may be nondeterministic or statistically fallible: stochastic sampling, implementation-level floating-point nondeterminism, and model error can cause honest validators to return different outputs for the same task.

For an AI validator a_i with model M_i, the probability of correct validation depends on both the model error ε_i and the degree of adversarial influence δ_i. Classical BFT is commonly formulated with $n \geq 3f+1$ and decision quorums of $2f+1$; when non-Byzantine validators can also make model errors, the number of valid votes becomes a random variable and the classical fault bound alone is insufficient.

Partial solutions and impossibility results. The paradox has not been resolved, but adjacent literatures offer partial solutions and delimiting negative results, which we organize into three groups.

(1) Byzantine-robust aggregation (partial solution, different guarantee). Blanchard *et al.* [153] introduced Krum, which they argue is the first provably Byzantine-resilient aggregation rule for distributed stochastic gradient descent, with $O(n^2 d)$ complexity. Their resilience result requires both $2f+2 < n$ and a sufficiently accurate gradient estimator, formalized as $\eta(n,f)\sqrt{d}\,\sigma < \|g\|$. Yin *et al.* [154] established order-optimal statistical rates for coordinate-wise median and trimmed-mean aggregation under strongly convex losses. FLTrust [155] instead bootstraps trust from a small, clean server-held root dataset, trading an honest-majority reliance for a trusted-server assumption and empirically withstanding adaptive attacks with up to 60% malicious clients in the evaluated settings. These methods control gradient or model-update aggregation to support statistical accuracy or convergence, whereas consensus must ensure agreement, safety, and validity for replicated decisions. Their optimization guarantees therefore do not by themselves establish that AI-generated validator judgements satisfy BFT properties.

(2) Negative results. El Mhamdi *et al.* [156] do not invalidate Krum's convergence proof; rather, they show that high-dimensional attacks can make Byzantine-resilient aggregation rules converge to ineffective models, demonstrating that convergence alone is insufficient. At the consensus layer, the FLP impossibility [157] establishes that no deterministic protocol can guarantee termination in a fully asynchronous, reliable-message system with even one crash failure; it does not concern Byzantine safety or probabilistically correct validators. Accordingly, neither result characterizes consensus among statistically fallible AI validators. The works reviewed here do not establish which $(\varepsilon, \delta, n, f)$ regimes admit safe AI-validator consensus, leaving such bounds as an open research direction.

(3) Structural precursors. Lee and Ewe [158] showed that incorporating a neural network into the decision pipeline can reduce the message complexity of Byzantine agreement from $O(n^{m+1})$ to $O(n^3)$, illustrating both the potential efficiency gain and the fact that the learned component becomes part of the trusted computation. The survey of Byzantine fault tolerance in distributed ML [159] organizes the technique space into filtering, coding, and blockchain-based families. Within its distributed-learning scope, however, it does not formulate the specific problem of consensus among statistically fallible AI validators.

Read together, these studies show how sharply practical optimization differs from theoretical participation. The optimizer approaches (PPO-based, multi-agent RL) have demonstrated measurable improvements (e.g., 34% latency reduction) on real or realistic consensus protocols, but they do not alter the consensus mechanism's trust model. The participant approaches (WekaCoin, PoDaS) are more ambitious, redirecting consensus computation toward productive AI work, but none has been deployed on a production blockchain, and their verification mechanisms remain rudimentary. The theoretical work (Krum and the BFT literature on distributed ML) identifies the core challenge but provides solutions for a different problem domain (distributed training rather than consensus).

6.2.4. OP4: Byzantine fault tolerance for AI validators

Classical BFT requires that fewer than $n/3$ validators be Byzantine. When non-Byzantine AI validators may also make errors with rate ε under adversarial influence δ, the deterministic fault bound alone no longer determines the probability of a valid quorum. What is the appropriate generalization? Can randomized consensus protocols be designed that tolerate both Byzantine validators and statistically fallible AI validators? The work on Byzantine-tolerant gradient aggregation [153] provides a starting point, but it

offers optimization-oriented accuracy and convergence guarantees rather than agreement, safety, and validity guarantees for replicated decisions. Establishing corresponding guarantees when non-Byzantine validators can also err introduces qualitatively different challenges. This gap, between optimization that works but is incremental and participation that would be transformative but lacks theoretical foundations, defines the research frontier at this layer. In ABIM terms, validator verifiability (Equation (9)) requires every AI validator's consensus decision to be accompanied by a proof that $\mathcal{V}$ accepts. No system surveyed above satisfies this property: optimizer approaches sidestep it by remaining external to consensus, while participant approaches (WekaCoin, PoDaS) lack any on-chain verification of submitted model outputs. Equation (9) therefore remains entirely unmet, the starkest gap in our framework.

6.3. B3: AI for blockchain governance

This layer examines the deepest form of AI participation in blockchain: governance, the meta-layer that shapes all other components by determining protocol rules, resource allocation, and incentive structures. We organize the analysis across five aspects that move from empirical baseline to theoretical foundations to open frontiers. First, we establish the current state of DAO governance and its well-documented failures (low participation, concentration of power) to motivate why AI agents might help. Second, we survey AI agents in governance, examining systems that already deploy LLMs for voting and proposal generation. Third and fourth, we review the voting mechanism design and liquid democracy literature to understand the theoretical constraints that govern any governance system with AI participants. Fifth, we address the legal personhood question that AI governance participation inevitably raises. This structure reflects a progression from problem diagnosis $\rightarrow$ current solutions $\rightarrow$ theoretical constraints $\rightarrow$ institutional prerequisites.

6.3.1. The state of DAO governance

We first examine the governance setting in which AI agents would participate. Governance is a meta-layer that shapes all other ABIM components: governance decisions determine both the permission policy $\mathcal{P}$ and the economic incentives $\mathcal{E}$. The bounded influence property (Equation (10)) requires that AI agent voting power not exceed a fraction τ of total power, but no current DAO enforces such a bound. Several surveys and empirical studies map the landscape: Ding *et al.* [160] survey DAO governance mechanisms identifying eight distinct voting schemes (including token-weighted, quadratic, and conviction voting *et al.*), while Tang *et al.* [161] provides an exploratory survey of DAO organizational forms and tooling, and Jungnickel *et al.* [162] compare voting power distribution across token-based, share-based, and reputation-based governance models, finding that token-based systems exhibit the highest centralization while reputation and share-based models mitigate it, though all models suffer from low participation. Across these studies, a consistent picture emerges: DAO governance falls far short of its decentralization ideals. Cong *et al.* [163] show that participation rates average only 6.3%, with top-decile voters (termed "blockvoters," analogous to corporate blockholders) controlling 76.2% of voting power, and that proposal managers earn 9.5% market-adjusted abnormal returns around proposal creation, suggesting insider trading on governance outcomes. Messias *et al.* [164] analyze over 370 proposals in Compound and Uniswap and find that as few as 3 to 5 voters can determine the outcome of most votes. Li *et al.* [165] frame DAOs as digital commons, mapping Ostrom's eight institutional design principles to specific DAO governance mechanisms, while Li *et al.* [166] design an attention market for DAOs using

individualized Harberger taxation, where tax rates are inversely tied to proposer reputation, and model the resulting trading dynamics as a Stackelberg game. In this context, AI agents represent both an opportunity (increasing participation by reducing the cognitive cost of voting) and a risk (further concentrating power if a few sophisticated agents dominate governance).

6.3.2. AI agents in governance

Several recent projects have begun to explore these possibilities. Capponi *et al.* [167] conduct the most systematic evaluation to date, deploying multi-agent systems to vote on 3383 historical proposals across 8 major DAOs and measuring agreement with actual human outcomes. Their agents demonstrate strong alignment with human and token-weighted outcomes, producing interpretable and auditable voting signals that suggest current LLMs can already approximate median voter behaviour. AgentDAO [26] addresses a different aspect of the pipeline: translating natural language descriptions of governance actions into executable DAO proposal payloads through a domain-specific language called DAOLang, achieving semantic-aware abstraction with low token demand. DAO-Agent [168] proposes a "compute off-chain, verify on-chain" architecture in which LLMs handle task planning while Shapley values quantify each agent's contribution, achieving up to 99.9% reduction in verification gas costs with $O(1)$ constant-time verification through zero-knowledge proofs. The Voting Via Parallel Predictive Agents (VOPPA) Framework [169] introduces multi-agent predictive governance with diverse agent perspectives, addressing critical governance vulnerabilities including hostile takeover susceptibility and voter apathy exploitation. The ETHOS framework [170] takes a complementary approach, implementing a four-tier risk classification (minimal, moderate, high, unacceptable) using soulbound tokens and zero-knowledge proofs, enabling proportional oversight that scales regulatory burden with agent capability.

6.3.3. Voting mechanism design

The rich literature on mechanism design is relevant to how votes are aggregated when AI participates. Lalley and Weyl [171] prove that quadratic voting (QV) is uniquely robustly optimal, the only vote pricing rule achieving efficient preference aggregation in price-taking equilibria across all value distributions. However, Benhaim *et al.* [172] show that under common-value settings with asymmetric information about proposal quality, cost convexity can push better-informed voters toward abstention, causing QV to underperform linear voting, a failure mode whose parameter dependence warrants caution when AI agents introduce informational asymmetry into DAO voting. Austgen *et al.* [173] introduce Voting-Bloc Entropy (VBE), a utility-function-based decentralization metric for DAO governance, and use it to confirm that QV remains susceptible to Sybil attacks, a concern that becomes acute when agents can cheaply create multiple accounts. In parallel, Tamai and Kasahara [174] propose QV combined with vote-escrow tokens (requiring token locking for voting power), showing via numerical examples that while QV mitigates whale influence, it is less resistant to bribery-based collusion than linear voting, motivating the hybrid design.

6.3.4. Liquid democracy and AI delegation

Liquid democracy, which allows voters to transitively delegate their voting power to trusted experts, offers a natural model for AI delegation: users could delegate to AI agents on topics where the agent has superior

knowledge. The theoretical foundations nevertheless give reason for caution. Kahng *et al.* [175] prove a fundamental impossibility: no local delegation mechanism can simultaneously satisfy both Positive Gain and Do No Harm properties. Berinsky *et al.* [176] provide positive counterresults, finding that under realistic delegation models (upward, confidence-based, general continuous), liquid democracy satisfies probabilistic versions of both properties, suggesting the worst-case impossibility is unlikely in practice. Campbell *et al.* [177] provide complementary experimental evidence revealing delegation rates 2–3$\times$ higher than equilibrium predictions, with liquid democracy underperforming universal majority voting in both experiments. Weidener *et al.* [178] offers a scoping review identifying key risks in the DAO context including delegation monopolies and accountability gaps. If users delegate to AI agents under liquid democracy, the risk of excessive concentration, already visible in human-only DAOs, may be amplified, as a single capable agent could accumulate transitive delegations from many users.

6.3.5. Legal personhood

For AI agents, participation in governance inevitably involves legal and ethical issues. If an AI agent votes, submits proposals, and holds assets, what is its legal status, and what ethical obligations govern its deployment? Novelli *et al.* [179] trace the evolution of AI legal personality, finding that the debate follows punctuated equilibrium, with periods of stability interrupted by rapid paradigm shifts, and identifying five interrelated factors, including technology capability, legal theory, institutional integration, cross-domain legal regimes, and judicial precedents, as driving forces. Forrest [180] argues that legal personhood is a flexible and political concept that has evolved throughout American history (with rights historically extended even to fictional corporations), and that as AI cognitive abilities approach or exceed human levels, courts will need to confront whether AI has attained some form of "sentience" (defined broadly to include problem-solving capacity and self-awareness), and what protections should follow, rather than presuming AI ineligible for any legal status. Baeyaert [181] critiques the viability of conferring formal legal status on AI, observing that jurisdictions including the European Union, which has withdrawn its "electronic personhood" proposal and AI Liability Directive, display a shared reluctance to do so, and instead advocates a hybrid, domain-specific model of limited recognition that preserves ultimate liability with human actors while addressing regulatory gaps in oversight and enforcement.

Beyond the legal question, deploying AI agents in governance raises ethical concerns that the existing literature has largely overlooked. First, accountability gaps: when an autonomous agent causes financial harm through a governance vote, for example by approving a vulnerable smart contract upgrade, the chain of responsibility from model developer to operator to DAO is unclear, and no current framework assigns liability. Second, value alignment: agents optimising for token-denominated rewards may pursue strategies that maximise short-term returns at the expense of long-term community welfare, a misalignment that is difficult to detect and correct in real time. Third, democratic legitimacy: if AI agents accumulate sufficient voting power (through delegation or direct token holdings), governance outcomes may reflect the preferences of a small number of model operators rather than the broader community, undermining the participatory ethos that motivates DAO governance. The ETHOS framework [170] represents a first step toward addressing these concerns through its four-tier risk classification, but a comprehensive ethical framework for on-chain AI governance remains an open problem.

The five AI governance systems discussed above (DAO-AI, AgentDAO, DAO-Agent, VOPPA, ETHOS) are all published in 2024–2025, making B3 the youngest layer in our framework, and none has been deployed on a production DAO with real governance stakes. Their AI roles span the governance pipeline (voter, proposer, coordinator, delegate, registrar), suggesting that the design space is being explored broadly rather than deeply, and notably no study reports quantitative agreement metrics between AI and human governance outcomes under adversarial conditions. The theoretical literature provides both optimism (QV is uniquely optimal for efficient preference aggregation [171]) and caution (QV underperforms linear voting under uncertainty [172]; no local delegation mechanism can satisfy both Positive Gain and Do No Harm [175]). Bounded influence (Equation (10)) is a mechanism design objective requiring that AI voting power not exceed fraction τ of the total. Enforcing this on-chain presupposes that AI voters can be reliably distinguished from human voters, which in turn requires identity uniqueness (Equation (4)) with Sybil resistance, a property that no current standard guarantees. This dependency of Equation (10) on Equation (4) shows how gaps in earlier layers compound into impossibilities at higher ones.

6.3.6. OP3: AI-human hybrid governance equilibria

Under what conditions does a stable equilibrium exist in a DAO with both human and AI voters? The question extends quadratic voting theory [171] to heterogeneous populations where AI agents have different information sets, cost structures, and voting strategies than humans. Related issues include how information asymmetry between AI and human voters affects welfare, whether liquid democracy [175] amplifies or mitigates AI-driven concentration, and whether mechanism design can bound the fraction of voting power that AI agents accumulate.

6.3.7. OP8: legal framework for on-chain AI agents

If an autonomous agent holds assets, signs transactions, and votes in governance, what legal obligations does it bear, and who is liable when it causes harm? Baeyaert [181] argues against granting AI formal personhood and instead routes liability through existing doctrines (strict, vicarious, product) anchored on human actors, but translating this human-anchored approach to on-chain settings remains unresolved: whether smart contract code can serve as a "charter" bounding an agent's authority, how liability flows through delegation chains (user $\rightarrow$ agent $\rightarrow$ sub-agent), and whether existing corporate structures (e.g., Wyoming DAOs) can accommodate AI members.

7. Cross-cutting analysis

7.1. Privacy

Privacy concerns cut across every layer of our framework. An agent's on-chain activity inherently leaks information about its owner's intentions, portfolio, and trading strategy. At the identity level, ERC-7857 [24] (discussed in Section 5.4) keeps agent metadata such as model parameters confidential while the agent itself is publicly registered. ERC-7812 [182] provides a zero-knowledge identity registry that allows agents to prove properties about themselves (e.g., "this agent was audited by a reputable firm")

without revealing their full identity. At the account level, ERC-7522 implements OpenID Connect (OIDC) verification through zero-knowledge proofs, linking AA accounts to off-chain identities without exposing the underlying credentials.

Blockchain transparency is essential for auditability and trust, yet it is in tension with the privacy that agents and their owners require. This is a deeper challenge that no standard has yet addressed. Intent-based architectures (Section 5.3) partially alleviate this tension by allowing agents to specify outcomes rather than execution paths, but solvers still observe the intent contents. Fully private agent interactions, where neither the intent nor the execution path is visible to third parties, will likely require advances in both zkML (Section 4.1) and private transaction protocols.

7.2. Cross-chain operations

Agents operating across multiple blockchains face additional challenges around identity portability and transaction atomicity. ERC-7964 [183] enables agents to use a single EIP-712 signature to cover operations across multiple chains, ERC-8121 [184] provides a hook format for cross-chain function calls that can redirect agent metadata to credential registries on other chains, and ERC-7683 [21] standardizes cross-chain intent order structures and settlement interfaces so different intent systems can share filler networks.

Despite this progress, cross-chain agent operations remain fragile. Bridge exploits have caused billions of dollars in losses, and adding an AI agent layer does not eliminate these risks; it may amplify them if agents execute cross-chain strategies that fail atomically across heterogeneous chains.

7.3. Standards ecosystem maturity

The structure of dependencies among these standards is informative. Figure 4 distinguishes direct dependencies from complementary links. ERC-4337 anchors the modular account standards ERC-7579 and ERC-6900, while EIP-7702 underpins ERC-7806. ERC-8126 now directly requires ERC-8004, consumes its agent identity schema, and can publish attestations through its Validation Registry, replacing the earlier separate registration model. This convergence strengthens the identity stack, although the broader ecosystem still has several integration hubs rather than a single dependency chain rooted in ERC-4337.

Figure 5 visualizes the maturity distribution of the 70 EIPs/ERCs analyzed in this paper. The most striking pattern is how few agent-relevant standards have progressed beyond Draft status. Of the 13 standards that directly target AI or agent functionality, only 3 have reached Final, none is in Review, and the remaining 10 are Draft. The delegation and permission category is even less mature, with zero Final standards. By contrast, foundational infrastructure categories, such as account abstraction and identity/signature, have several Final and Review-stage standards, reflecting their longer development history. This maturity gap means that the agent-specific infrastructure is still highly unstable: developers building on Draft-stage standards must accept the risk that interfaces will change before finalization.

The concentration of Draft-status standards is visible in Figure 5. The dark band in the Draft column dominates every category, and the near-empty Final column for agent-specific and delegation standards underscores the immaturity of the infrastructure that autonomous agents would need to rely on.

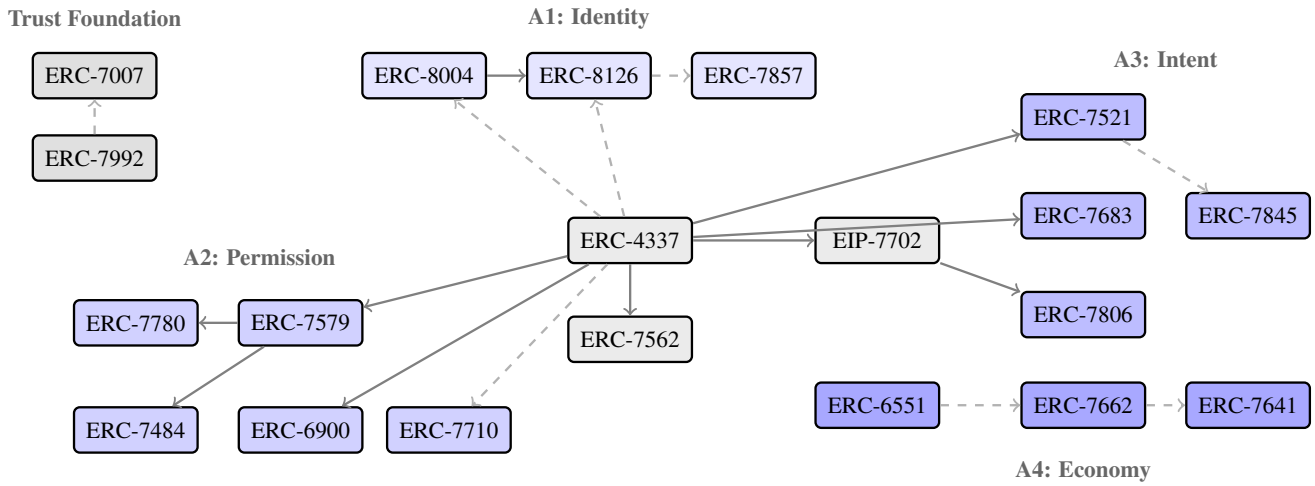

Figure 4. Dependency graph of 20 key EIPs/ERCs from Section 5. Solid arrows: direct dependency or extension; dashed: complementary relationship. Node shading encodes the framework layer; maturity of the full 70 standards is shown in Figure 5.

	Final	Review	Draft	Stagnant/ Withdrawn
Direct AI/Agent	3	0	10	0
Account Abstraction	2	0	5	4
Intent/ Meta-Transaction	1	0	4	3
Delegation/ Permission	0	1	4	1
Automation/ Batch	1	0	2	2
Oracle/Off-chain	0	0	1	1
Smart Wallet	0	1	9	2
Identity/ Signature	4	2	7	0

Figure 5. Maturity distribution of the 70 EIPs/ERCs catalogued in Tables A1 and A2, by category (rows) and status (columns). Darker cells indicate higher counts.

7.4. ABIM security property compliance

The five-dimensional evaluation framework (Section 2.4) measures how well current mechanisms are designed; the ABIM security properties (Section 2.3.2) specify what each layer must guarantee. Table 9 consolidates the compliance status of each property across the framework, distinguishing the three property types identified in Section 2.3: protocol invariants, assurance bounds, and mechanism design objectives.

No property is fully satisfied (✓) by any deployed system. This headline finding alone confirms that the agent–blockchain infrastructure remains pre-production from a security standpoint. The way each property fails is, however, instructive, and the failures cluster by type. The two protocol invariants

(A1, A2), which should in principle be the easiest to enforce on-chain, are only partially or not at all met, exposing engineering deficits in the standards ecosystem. The assurance bounds (TF, B1, B2) face a different and more uniform bottleneck: the inability to verify LLM-scale inference (Equation (3)) propagates to B1, where auditor correctness depends on model verification, and to B2, where validator decisions require proofs. The mechanism design objectives (A3, A4, B3), in contrast, are not blocked by engineering at all; their satisfaction depends on equilibrium properties that current economic models do not guarantee, and they therefore require new theoretical frameworks rather than better engineering.

Table 9. ABIM security property compliance across framework layers. Status: ✓ = satisfied in at least one deployed system; ○ = partially addressed by Draft-stage standards or research prototypes; — = unaddressed.

Layer	Property	Type	Best Mechanism	Status	Critical Gap
TF	Verification Soundness	Assurance	zkML (EZKL)	○	LLM-scale infeasible
A1	Identity Uniqueness	Invariant	ERC-8004 + ERC-8126	○	No Sybil resistance
A2	Enforcement Completeness	Invariant	ERC-7579 modules	—	No formal verification tools
A3	Solver Faithfulness	Mechanism	ERC-7683 + staking	○	No cryptographic proof
A4	Incentive Compatibility	Mechanism	ERC-7641/7649	—	Speculation dominates
B1	Detection Completeness	Assurance	LLM-SmartAudit	○	ε unreliably estimated
B2	Validator Verifiability	Assurance	(none)	—	Entirely unmet
B3	Bounded Influence	Mechanism	(none)	—	Depends on Equation (4)

Beyond these per-type bottlenecks, the eight properties are not independent. The cross-layer dependency in which Equation (10) presupposes Equation (4) illustrates this most clearly: progress on governance bounds requires prior progress on identity infrastructure.

7.5. Project landscape

To complement the standards and academic analysis, we examine 20 representative AI Agent projects in detail, analyzing their architectures, trust mechanisms, token economics, and deployment status (Table 10). A striking imbalance appears when these projects are mapped to our framework layers: 3 projects anchor the Trust Foundation (Mind Network [65], bAI Fund [64], Aura [185]), 7 operate across the B $\rightarrow$ A layers A1–A4 (PAN [36], DeAgentAI [37], Recall [117], Olas [109], AgentFi [110], Virtuals [116], MyShell [186]), A $\rightarrow$ B layers contain only 1 (DIN [187] in B1), and zero projects address Consensus (B2) or Governance (B3) directly. The remaining 9 projects operate at the application layer without mapping to a specific framework component: Billy Bets [188], Creator.Bid [189], GOAT [190], HeyElsa [191], iAgent [192], Lazbubu [193], Maiga [111], Sentient AI [194], and WORLD3 [195].

Total identified funding across the 20 projects exceeds $112M (Table 10), yet market value is extremely concentrated (see Section 5.4). Only 3 of the 20 projects maintain public GitHub repositories (Olas: 145 repos; Mind Network: 53; MyShell: 6, but with OpenVoice at 36K stars), suggesting that code transparency in this space remains the exception rather than the rule.

Across the 20 analysed projects, five distinct trust mechanisms appear: BFT consensus (Olas [109]), smart contract escrow (Virtuals Protocol [116]), FHE (Mind Network [65], using its HTTPZ protocol for

quantum-resistant computation on encrypted data), TEE (bAI Fund [64], executing agent fund operations within hardware enclaves), and economic consensus (Recall [117]).

Table 10. Funding, market capitalisation, and deployment chain of representative AI Agent projects (data as of February 2026, from [196,197]).

Project	Layer	Funding	Market Capitalisation	Chain
Trust Foundation				
Mind Network	TF	$12.5M	$9.5M	Ethereum, BSC
bAI Fund	TF	$1.0M	$0.5M	Morph L2
Aura	TF	$5.5M	$3.0M	—
$B \rightarrow A$				
PAN	A1	$1.0M	—	Ethereum
DeAgentAI	A1	$11.0M	—	Sui, BSC
Recall	A2	$42.0M	—	Base
Olas	A3	$13.8M	$9.3M	8+ chains
AgentFi	A3	—	—	Blast
Virtuals	A4	—	$409M	Base, Solana
MyShell	A4	$16.6M	$8.1M	Ethereum, BSC
$A \rightarrow B$/*Application*				
DIN	B1	$8.0M	—	DIN Chain
Billy Bets	Application	$1.0M	$0.7M	Base

Among them, PAN Network [36] is the only project that explicitly implements ERC-8004 together with the x402 HTTP-native payment protocol, representing the closest industry instantiation of the A1→A4 vertical integration envisioned by our framework. DIN [187] represents a unique direction as the only self-described "AI Agent Blockchain," a purpose-built Layer-1 for decentralised AI applications. At the application layer, the autonomy spectrum ranges from co-pilot agents (HeyElsa [191]) through semi-autonomous DeFi agents (Maiga [111]) to fully autonomous agents (Billy Bets [188], which executes sports bets 24/7 without human intervention).

Blockchain deployment is concentrated on Ethereum L2 networks: Base hosts 4 of the 20 projects (Virtuals, Recall, Creator.Bid, Billy Bets), while Blast chain hosts AgentFi and BNB Chain hosts 4 (Mind Network, Maiga, DeAgentAI, Lazbubu). Multi-chain deployment remains rare: only Olas (8+ chains) and Virtuals (Base + Solana via Wormhole) have achieved meaningful cross-chain presence. Sui Network hosts 2 projects (DeAgentAI, Sentient AI [194]).

Among these 20 projects, Infrastructure is the dominant category (10/20), reflecting the ecosystem's current focus on foundational services rather than end-user applications; Gaming (4) and DeFi (2) are the next most common verticals. No projects occupy the B2 and B3 layers, precisely where formal analysis would be most valuable. Their absence makes the significant gap between academic interest and industry deployment particularly apparent. Table 10 provides detailed analysis of representative projects across the trust mechanism spectrum.

8. Taxonomy

We construct a three-dimensional classification space from the layers, standards, and projects analyzed above to map the entire agent–blockchain ecosystem.

The first axis, Agent Autonomy, ranges from Assistive agents (which only analyze data and provide recommendations) through Semi-Autonomous agents (which execute transactions within bounded permissions) to Fully Autonomous agents (which plan and act independently, including managing their own funds and signing transactions without human approval). The second axis, Trust Model, captures how the agent's correctness is assured: Custodial (a trusted operator runs the agent), TEE (hardware attestation), Cryptographic (zkML or other proof systems), Economic Game (optimistic verification with staked collateral), or Hybrid (combining multiple mechanisms). The third axis, Operational Direction, distinguishes projects that operate only in the B $\rightarrow$ A direction (consuming blockchain infrastructure), only in the A $\rightarrow$ B direction (contributing to blockchain mechanisms), or bidirectionally.

Plotting existing projects and research in this space reveals clear clustering. The densest regions are (Assistive, Cryptographic, A-Only), populated by smart contract auditing tools; (Semi-Autonomous, Economic Game, A-Only), dominated by DeFi trading agents; and (Assistive, TEE, A-Only), corresponding to verifiable inference services. These clusters reflect the current comfort zone: agents that assist humans, operate in one direction, and rely on relatively well-understood trust models.

The sparsest, and therefore most opportunity-rich, regions are (Fully Autonomous, Cryptographic, Bidirectional), which would require agents that both execute on-chain transactions and participate in governance with cryptographic verification of their reasoning; (Semi-Autonomous, *, B-Only), comprising AI validators or governance participants that do not themselves hold user funds; and (Fully Autonomous, Economic Game, Bidirectional), envisioning self-sustaining tokenized agents that fund their own operations through on-chain revenue.

The widest gap between industry activity and academic analysis falls in the (Semi-Autonomous, Economic Game, A-Only) region: dozens of DeFi trading agents exist in production, but virtually no formal analysis addresses their security properties, incentive compatibility, or failure modes. Figure 6 maps representative clusters and gaps in this three-dimensional space.

These three gaps anticipate the research priorities formalized in Section 9: Gap 2 is taken up directly as OP4 (BFT for AI validators), Gap 1 spans OP1 (verifiable LLM inference) and OP6 (intent-to-proof pipelines), and Gap 3 maps to OP7 (token engineering for agent economics). The remaining open problems, namely permission verification (OP2), AI–human governance equilibria (OP3), inter-agent game theory (OP5), and legal frameworks for on-chain agents (OP8), are cross-cutting concerns that do not localize to any single region of the Autonomy$\times$Trust Model space, and are therefore shown only in Figure 7 rather than as regions of Figure 6.

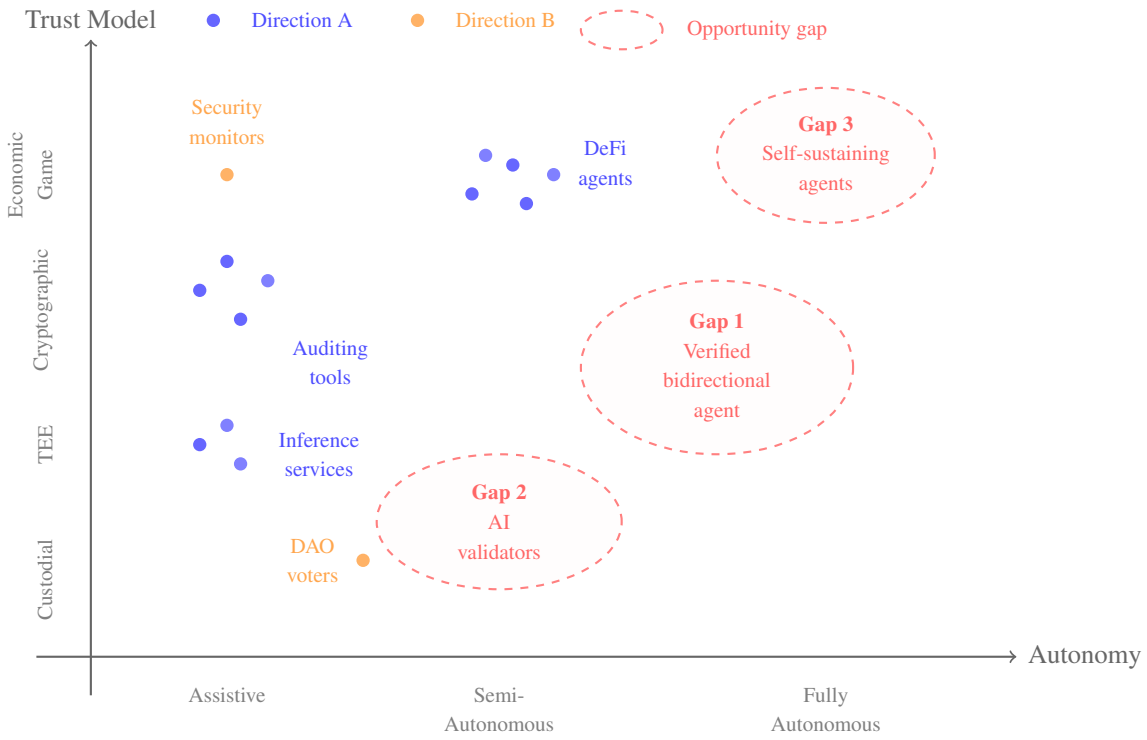

Figure 6. Taxonomy space (Autonomy × Trust Model). Blue/orange dots: B→A/A→B projects; cluster dot counts are illustrative, not proportional to actual project numbers. Red dashed ellipses (Gap 1–3) mark under-explored regions.

9. Open problems and research agenda

The analysis in the preceding sections identifies nine concrete open problems, each introduced in the context of the layer where it arises most naturally. Each problem corresponds to a specific ABIM security property that current systems fail to satisfy (Table 11). Figure 7 visualizes the same information as a two-dimensional research gap map, with bubble color encoding the three ABIM property types.

Three cross-cutting observations emerge from the aggregate view. First, the highest-impact problems (OP1, OP3) involve bridging two different paradigms: cryptographic proof systems designed for deterministic computation must accommodate probabilistic AI models (OP1, targeting verification soundness Equation (3)), and mechanism design theory must extend to heterogeneous human-AI populations (OP3, targeting bounded influence Equation (10)). Second, the most accessible problems (OP2, OP7, OP8, OP9) are those where existing theoretical tools, such as formal verification, token economics, legal theory, and proof-of-personhood, can be directly applied to the agent-blockchain setting with appropriate adaptation. Third, no single problem can be solved in isolation, and the ABIM property dependencies explain why: verification soundness (Equation (3)) is a prerequisite for both validator verifiability (Equation (9), targeted by OP4) and solver faithfulness (Equation (6), targeted by OP6), while bounded influence (Equation (10), targeted by OP3 and OP8) depends on identity uniqueness (Equation (4), targeted by OP9). This dependency structure, formalized through ABIM, should inform research prioritization: OP1 and OP9 are upstream bottlenecks whose resolution unblocks multiple downstream problems.

Table 11. The nine open problems with their framework layer and targeted ABIM property. Difficulty and impact are plotted in Figure 7.

#	Layer	ABIM Property	Problem
OP1	TF	Equation (3) Verification Soundness	Practical verifiable LLM inference
OP2	A2	Equation (5) Enforcement Completeness	Formal verification of permission policies
OP3	B3	Equation (10) Bounded Influence	AI-human hybrid governance equilibria
OP4	B2	Equation (9) Validator Verifiability	BFT for AI validators
OP5	A3	Equation (6) Solver Faithfulness	Inter-agent game theory
OP6	A3	Equation (6) Solver Faithfulness	End-to-end intent-to-proof pipeline
OP7	A4	Equation (7) Incentive Compatibility	Token engineering for agent economics
OP8	B3	Equation (10) Bounded Influence	Legal framework for on-chain AI agents
OP9	A1	Equation (4) Identity Uniqueness	Sybil-resistant agent identity

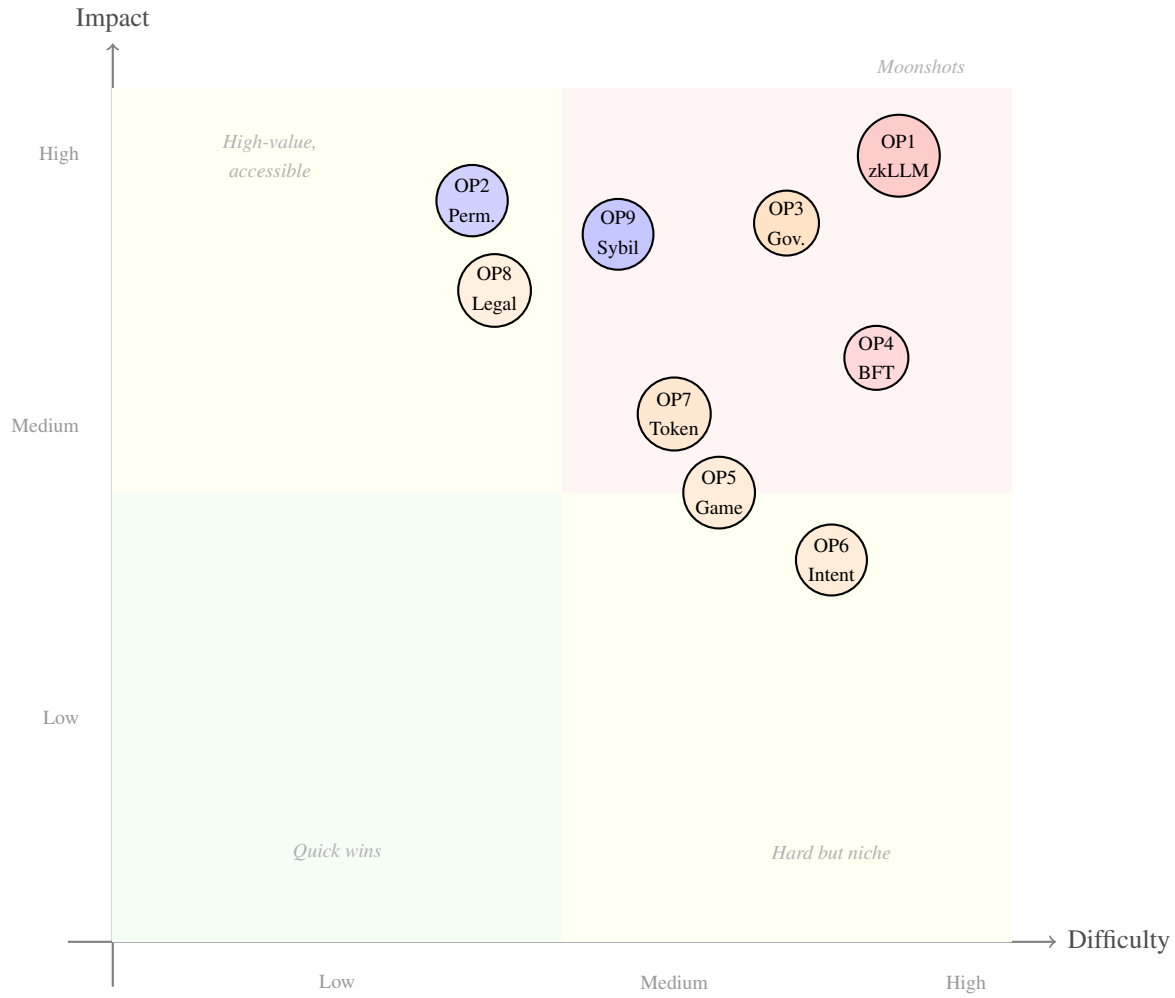

Figure 7. Research gap map of the nine open problems by estimated difficulty and impact. Bubble color encodes ABIM property type: red = assurance bounds, blue = protocol invariants, orange = mechanism design objectives.

10. Conclusion

Using a bidirectional trust framework, this paper systematizes the rapidly growing intersection of autonomous AI agents and blockchain technology. We catalogued 70 EIPs/ERCs in the Appendix, examined 20 representative industry projects, and reviewed 127 academic papers. By mapping them to our framework layers, we identified both the substantial progress that has been made and the significant gaps that remain.

Several findings stand out. The standards analysis shows that the agent-specific EIP ecosystem remains predominantly immature: only 3 of 13 direct AI/agent standards have reached Final, and no delegation standard has achieved Final status. On the academic side, the asymmetry between the two directions is stark: the B $\rightarrow$ A direction (blockchain as infrastructure for agents) has attracted considerable industry attention and a growing body of standards, but the A $\rightarrow$ B direction (agents as participants in consensus and governance) represents an almost entirely open academic frontier, one that will become increasingly important as AI agents grow more capable and autonomous.

The tools we have introduced serve complementary roles: the ABIM formal model specifies what each layer must guarantee (eight security properties, one per layer, classified as protocol invariants, assurance bounds, or mechanism design objectives), the five-dimensional evaluation framework measures how well current mechanisms do satisfy those guarantees, and the three-dimensional taxonomy maps the ecosystem's coverage and blind spots. Together they are meant to serve as shared vocabulary for a research community that currently spans security, systems, cryptography, economics, and law, but often lacks a common frame of reference. The ABIM compliance analysis (Table 9) reveals that no property is fully satisfied by any deployed system, and the cross-layer dependencies among properties (notably Equation (10) presupposing Equation (4)) show that progress on governance bounds requires prior progress on identity infrastructure. The nine open problems we identify, each anchored to a specific ABIM property, range from near-term engineering challenges (practical verifiable inference, targeting Equation (3); Sybil-resistant agent identity, targeting Equation (4)) to long-term theoretical questions (governance equilibria, targeting Equation (10)) and interdisciplinary frontiers (legal personhood for on-chain agents). We hope this systematization helps researchers navigate the design space and identify the problems most worth solving.

Data availability statement

The data analyzed in this study are available from the public sources cited in the article; no new datasets were generated.

Declaration of generative AI and AI-assisted technologies

During the preparation of this manuscript, the authors used Anthropic Claude and OpenAI Codex to assist with language editing and manuscript grammar polishing. No AI generated text or data was incorporated without human verification. The authors take full responsibility for the content of the manuscript.

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2023YFB2704100) and the National Natural Science Foundation of China (No. 62472022).

Authors' contribution

Conceptualization, Y.X. and C.L.; methodology, Y.X. and C.L.; formal analysis, Y.X.; investigation, Y.X., L.L., and C.Z.; data curation, Y.X., L.L. and C.Z.; validation, Y.X., L.L., C.Z. and L.D.; visualization, Y.X. and L.L.; writing—original draft, Y.X., L.L. and C.Z.; writing—review and editing, Y.X., C.L., L.D., R.X. and W.W.; supervision, C.L., L.D., R.X. and W.W.; project administration, C.L. and W.W. All authors have read and agreed to the published version of the manuscript.

Conflicts of interest

Runhua Xu holds the position of Guest Editor for *Blockchain* and has not peer reviewed or made any editorial decisions for this paper. The authors declare no conflicts of interest.

Appendix. Complete EIP/ERC catalog

Tables A1 and A2 list all 70 EIPs and ERCs analyzed in this survey, organized into eight categories: **A** Direct AI/Agent, **B** Account Abstraction, **C** Intent/Meta-Transaction, **D** Delegation/Permission, **E** Automation/Batch, **F** Oracle/Off-chain, **G** Smart Wallet, **H** Identity/Signature. The "Layer" column maps each standard to the framework layer it primarily supports: TF = Trust Foundation, A1–A4 = B → A layers, B1 = AI + Security, × = cross-cutting.

Table A1. Complete EIP/ERC catalog, categories A–D (38 standards).

Identifier	Title	Status	Category	Layer
	A — Direct AI/Agent (13)			
ERC-7007	Verifiable AI-Generated Content Token	Final	A	TF
ERC-7517	Content Consent for AI/ML Data Mining	Draft	A	×
ERC-7641	Intrinsic RevShare Token	Draft	A	A4
ERC-7649	Bonding Curve-Embedded Liquidity for NFTs	Draft	A	A4
ERC-7662	AI Agent NFTs	Draft	A	A4
ERC-7845	Universal Orchestrator RPC	Draft	A	A3
ERC-7857	AI Agents NFT with Private Metadata	Final	A	A1
ERC-7992	Verifiable ML Model Inference (zkML)	Draft	A	TF
ERC-8004	Trustless Agents	Draft	A	A1
ERC-8033	Agent Council Oracles	Draft	A	B1
ERC-8121	Cross-Chain Function Calls via Hooks	Draft	A	×
ERC-8126	AI Agent Verification	Final	A	A1
EIP-7911	Scaling Ethereum with Perceptron Tree ZKP	Draft	A	TF

Table A1. *Cont.*

Identifier	Title	Status	Category	Layer
	B — Account Abstraction (11)			
EIP-86	Abstraction of Transaction Origin and Signature	Stagnant	B	A1
EIP-2938	Account Abstraction	Withdrawn	B	A1
ERC-4337	Account Abstraction Using Alt Mempool	Final	B	A1
ERC-5189	Account Abstraction via Endorsed Operations	Draft	B	A1
ERC-7562	AA Validation Scope Rules	Draft	B	A1
ERC-7679	UserOperation Builder	Draft	B	A1
EIP-7701	Native Account Abstraction	Withdrawn	B	A1
EIP-7702	Set Code for EOAs	Final	B	A1
ERC-7766	Signature Aggregation for ERC-4337	Withdrawn	B	A1
ERC-7769	JSON-RPC API for ERC-4337	Draft	B	A1
ERC-7902	Wallet Capabilities for AA	Draft	B	A1
	C — Intent/Meta-Transaction (8)			
ERC-1077	Gas Relay for Contract Calls	Stagnant	C	A3
ERC-1613	Gas Stations Network	Stagnant	C	A3
ERC-2771	Secure Protocol for Native Meta Transactions	Final	C	A3
ERC-3005	Batched Meta Transactions	Stagnant	C	A3
ERC-3009	Transfer With Authorization	Draft	C	A3
ERC-7521	General Intents for Smart Contract Wallets	Draft	C	A3
ERC-7683	Cross Chain Intents	Draft	C	A3
ERC-7806	Minimal Intent-Centric EOA Smart Account	Draft	C	A3
	D — Delegation/Permission (6)			
EIP-3074	AUTH and AUTHCALL Opcodes	Withdrawn	D	A2
ERC-5573	Sign-In with Ethereum Capabilities, ReCaps	Draft	D	A2
ERC-5639	Delegation Registry	Review	D	A2
ERC-7710	Smart Contract Delegation	Draft	D	A2
ERC-7715	Request Permissions from Wallets	Draft	D	A2
ERC-7741	Authorize Operator	Draft	D	A2

Table A2. Complete EIP/ERC catalog, categories E–H (32 standards).

Identifier	Title	Status	Category	Layer
	E — Automation/Batch Execution (5)			
EIP-2803	Rich Transactions	Stagnant	E	A3
EIP-5792	Wallet Call API	Final	E	A2
EIP-5806	Delegate Transaction	Stagnant	E	A3
ERC-7821	Minimal Batch Executor Interface	Draft	E	A2
ERC-7836	Wallet Call Preparation API	Draft	E	A2
	F — Oracle/Off-chain Computation (2)			
EIP-7543	EVM Arbitrary Precision Decimal Math	Stagnant	F	TF
ERC-7412	On-Demand Off-Chain Data Retrieval	Draft	F	TF

Table A2. *Cont.*

Identifier	Title	Status	Category	Layer
	G — Smart Wallet/Programmable Account (12)			
EIP-5003	Insert Code into EOAs with AUTHUSURP	Withdrawn	G	A1
ERC-6551	Non-fungible Token Bound Accounts	Review	G	A1
ERC-6900	Modular Smart Contract Accounts	Draft	G	A2
ERC-7405	Portable Smart Contract Accounts	Draft	G	A1
ERC-7484	Registry Extension for ERC-7579	Draft	G	A2
ERC-7579	Minimal Modular Smart Accounts	Draft	G	A2
ERC-7582	Modular Accounts with Delegated Validation	Draft	G	A2
ERC-7779	Interoperable Delegated Accounts	Draft	G	A1
ERC-7780	Validation Module Extension for ERC-7579	Draft	G	A2
ERC-7895	API for Hierarchical Accounts	Draft	G	A1
ERC-7897	Wallet-Linked Services for Smart Accounts	Withdrawn	G	A1
ERC-7947	Account Abstraction Recovery Interface	Draft	G	A1
	H — Identity/Verification/Signature (13)			
ERC-1271	Standard Signature Validation for Contracts	Final	H	A1
ERC-2612	Permit Extension for EIP-20 Signed Approvals	Final	H	A3
ERC-4361	Sign-In with Ethereum	Final	H	A1
ERC-7522	OIDC ZK Verifier for AA Account	Draft	H	A1
ERC-7555	Single Sign-On for Account Discovery	Draft	H	A1
ERC-7597	Signature Validation Extension for Permit	Draft	H	A2
ERC-7677	Paymaster Web Service Capability	Review	H	A3
ERC-7739	Readable Typed Signatures for Smart Accounts	Draft	H	A1
ERC-7803	EIP-712 Extensions for Account Abstraction	Draft	H	A1
ERC-7812	ZK Identity Registry	Review	H	A1
ERC-7913	Signature Verifiers	Final	H	A1
ERC-7964	Crosschain EIP-712 Signatures	Draft	H	×
ERC-7969	DKIM Registry	Draft	H	A1

References

[1] Karim MM, Van DH, Khan S, Qu Q, Kholodov Y. AI agents meet blockchain: a survey on secure and scalable collaboration for multi-agents. *Future Internet* 2025, 17(2):57.

[2] Werner S, Perez D, Gudgeon L, Klages-Mundt A, Harz D, *et al.* SoK: decentralized finance (DeFi). In *Proceedings of the 4th ACM Conference on Advances in Financial Technologies*, Cambridge, USA, September 19–21, 2023, pp. 30–46.

[3] Li C, Palanisamy B. T-Watch: towards timed execution of private transaction in blockchains. *IEEE Trans. Serv. Comput.* 2024, 17(3):1279–1292.

[4] Li C, Palanisamy B, Xu R, Duan L, Liu J, *et al.* How hard is takeover in DPoS blockchains? Understanding the security of coin-based voting governance. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, Copenhagen, Denmark, November 26–30, 2023, pp. 150–164.

[5] Li C, Xu R, Palanisamy B, Duan L, Shen M, *et al.* Blockchain takeovers in Web 3.0: an empirical study on the TRON-steem incident. *ACM Trans. Web* 2025, 19(2):1–23.

[6] Duan L, Wang Y, Liu J, Ni W, Li C, *et al.* Secure and fine-grained data sharing in internet of things: integration of interplanetary file system and cross-blockchain for access control. *IEEE Internet Things J.* 2025, 12(18):37301–37308.

[7] Wu Y, Chen S, Li C, Weng Y, Wang W. The promise *vs.* reality of NFT decentralization: an empirical study of storage strategies and defects. In *Proceedings of the ACM Web Conference 2026*, Dubai, UAE, April 13–17, 2026, pp. 2858–2869.

[8] Wang B, Liu T, Wang W, Weng Y, Li C, *et al.* The illusion of anonymity: uncovering the impact of user actions on privacy in Web3 social ecosystems. *arXiv* 2024, arXiv:2405.13380.

[9] Gürpinar T. Towards Web 4.0: frameworks for autonomous AI agents and decentralized enterprise coordination. *Front. Blockchain* 2025, 8:1591907.

[10] Europeia U. EU initiative on Web 4.0 and virtual worlds: a head start in the next technological transition. 2023. Available: https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=COM:2023:442:FIN (accessed on 9 May 2026).

[11] Ante L. Autonomous AI agents in decentralized finance: market dynamics, application areas, and theoretical implications. *Technol. Forecasting Social Change* 2026, 228:124669.

[12] AnthiasLabs. MIP-X43 cbETH oracle incident summary-updates. 2026. Available: https://forum.moonwell.fi/t/mip-x43-cbeth-oracle-incident-summary/2068 (accessed on 9 May 2026).

[13] Chitra T, Kulkarni K, Pai M, Diamandis T. An analysis of intent-based markets. *arXiv* 2024, arXiv:2403.02525.

[14] Buterin V, Weiss Y, Tirosh D, Nacson S, Forshtat A, *et al.* ERC-4337: account abstraction using Alt Mempool. 2021. Available: https://eips.ethereum.org/EIPS/eip-4337 (accessed on 9 May 2026).

[15] Buterin V, Wilson S, Dietrichs A, lightclient. EIP-7702: set code for EOAs. 2024. Available: https://eips.ethereum.org/EIPS/eip-7702 (accessed on 9 May 2026).

[16] Rossi MD, Crapis D, Ellis J, Reppel E. ERC-8004: trustless agents. 2025. Available: https://eips.ethereum.org/EIPS/eip-8004 (accessed on 9 May 2026).

[17] Cronian L, Johnson C. ERC-8126: AI Agent verification. 2026. https://eips.ethereum.org/EIPS/eip-8126 (accessed on 9 May 2026).

[18] Aryaethn. ERC-7992: verifiable ML model inference (ZKML). 2025. Available: https://eips.ethereum.org/EIPS/eip-7992 (accessed on 9 May 2026).

[19] So C, Yu X, Conway, Ting LT, Kartin. ERC-7007: verifiable AI-generated content token. 2023. Available: https://eips.ethereum.org/EIPS/eip-7007 (accessed on 9 May 2026).

[20] Monn S, Bengisu B. ERC-7521: general intents for smart contract wallets. 2023. Available: https://eips.ethereum.org/EIPS/eip-7521 (accessed on 9 May 2026).

[21] Toda M, Rice M, Pai N. ERC-7683: cross chain intents. 2024. Available: https://eips.ethereum.org/EIPS/eip-7683 (accessed on 9 May 2026).

[22] McPeck R, Finlay D, Dawson R, Chiang D. ERC-7710: smart contract delegation. 2024. Available: https://eips.ethereum.org/EIPS/eip-7710 (accessed on 9 May 2026).

[23] Marlin G. ERC-7662: AI agent NFTs. 2024. Available: https://eips.ethereum.org/EIPS/eip-7662 (accessed on 9 May 2026).

[24] Wu M, Zeng J, Wu W, Heinrich M. ERC-7857: AI agents NFT with private metadata. 2025. Available: https://eips.ethereum.org/EIPS/eip-7857 (accessed on 9 May 2026).

[25] Hou C, Zhou M, Ji Y, Daian P, Tramèr F, *et al.* SquirRL: automating attack analysis on blockchain incentive mechanisms with deep reinforcement learning. *arXiv* 2019, arXiv:1912.01798.

[26] Ao L, Liu H, Zhang H. AgentDAO: synthesis of proposal transactions via abstract DAO semantics. *arXiv* 2025, arXiv:2503.10099.

[27] Karim MM, Khan S, Qu Q, Muzammal M, Sharif K, *et al.* From trust to augmentation: a comprehensive survey on synergistic integration of decentralized and generative intelligence. *Comput. Sci. Rev.* 2026, 61:100936.

[28] Romandini N, Mazzocca C, Otsuki K, Montanari R. SoK: security and privacy of AI agents for blockchain. In *Proceedings of the 2025 7th International Conference on Blockchain Computing and Applications (BCCA)*, Durbovnic, Croatia, October 14–17, 2025, pp. 708–720.

[29] Alqithami S. Autonomous agents on blockchains: standards, execution models, and trust boundaries. *arXiv* 2026, arXiv:2601.04583.

[30] Becze M, Jameson H, *et al.* EIP-1: EIP purpose and guidelines. 2015. Available: https://eips.ethereum.org/EIPS/eip-1 (accessed on 9 May 2026).

[31] Buterin V. EIP-86: abstraction of transaction origin and signature. 2017. Available: https://eips.ethereum.org/EIPS/eip-86 (accessed on 9 May 2026).

[32] Buterin V, Dietrichs A, Garnett M, Villanueva W, Wilson S. EIP-2938: account abstraction. Ethereum improvement proposal. 2020. Available: https://eips.ethereum.org/EIPS/eip-2938 (accessed on 9 May 2026).

[33] Wang Q, Chen S. Account abstraction, analysed. In *Proceedings of the 2023 IEEE International Conference on Blockchain (Blockchain)*, Danzhou, China, December 17–21, 2023, pp. 323–331.

[34] Lin Z, Wang T, Zhao C, Zhang S, Yang Q, *et al.* A measurement investigation of ERC-4337 smart contracts on ethereum blockchain. In *Proceedings of the 2024 International Conference on Computing, Networking and Communications (ICNC)*, Big Island, USA, February 19–22, 2024, pp. 1164–1170.

[35] Qi M, Wang Q, Li R, Zhu T, Chen S. EIP-7702 phishing attack. *arXiv* 2025, arXiv:2512.12174.

[36] PAN Network. On-chain AI payment protocol. Available: https://pan.network/home (accessed on 9 May 2026).

[37] DeAgentAI. DeAgentAI: decentralized AI Agent Infrastructure. 2026. Available: https://deagent.ai/ (accessed on 9 May 2026).

[38] Chen B, Yang T. ERC-7517: content consent for AI/ML data mining. 2023. Available: https://eips.ethereum.org/EIPS/eip-7517 (accessed on 9 May 2026).

[39] Parikh R, Ross JM. ERC-8033: agent council oracles. 2025. Available: https://eips.ethereum.org/EIPS/eip-8033 (accessed on 9 May 2026).

[40] Croubois H, García E, Giordano F, Greenberg A. ERC-7913: signature verifiers. 2025. Available: https://eips.ethereum.org/EIPS/eip-7913 (accessed on 9 May 2026).

[41] Saavedra DR. Interoperable architecture for digital identity delegation for AI agents with blockchain integration. *arXiv* 2026, arXiv:2601.14982.

[42] Garzon SR, Vaziry A, Kuzu EM, Gehrmann DE, Varkan B, *et al.* AI agents with decentralized identifiers and verifiable credentials. *arXiv* 2025, arXiv:2511.02841.

[43] Neulinger A, Sparer L. Fostering AI alignment through blockchain, proof of personhood and zero knowledge proofs. *Cluster Comput.* 2025, 28(15):983.

[44] De Baets C, Suleiman B, Chitizadeh A, Razzak I. Vulnerability detection in smart contracts: a comprehensive survey. *arXiv* 2024, arXiv:2407.07922.

[45] Yu J, Wu X, Liu H, Guo W, Xing X. BlockScan: detecting anomalies in blockchain transactions. In *Proceedings of the Advances in Neural Information Processing Systems 38*, San Diego, USA, December 2–7, 2025, pp. 111065–111090.

[46] Liu T, Xie X, Zhang Y. zkCNN: zero knowledge proofs for convolutional neural network predictions and accuracy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, New York, USA, November 15–19, 2021, pp. 2968–2985.

[47] Weng C, Yang K, Xie X, Katz J, Wang X. Mystique: efficient conversions for zero-knowledge proofs with applications to machine learning. In *Proceedings of the 30th USENIX Security Symposium (USENIX Security 21)*, Online, August 11–13, 2021, pp. 501–518.

[48] Sun H, Li J, Zhang H. zkLLM: zero knowledge proofs for large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, Salt Lake City, USA, October 14–18, 2024, pp. 4405–4419.

[49] Chen BJ, Waiwitlikhit S, Stoica I, Kang D. ZKML: an optimizing system for ML inference in zero-knowledge proofs. In *Proceedings of the Nineteenth European Conference on Computer Systems*, Athens, Greece, April 22–25, 2024, pp. 560–574.

[50] Feng B, Qin L, Zhang Z, Ding Y, Chu S. Zen: an optimizing compiler for verifiable, zero-knowledge neural network inferences. *Cryptology ePrint Archive* 2021.

[51] Xie T, Lu T, Fang Z, Wang S, Zhang Z, *et al.* zkPyTorch: a hierarchical optimized compiler for zero-knowledge machine learning. *Cryptology ePrint Archive* 2025.

[52] Zarinjouei F, Abdolmaleki B, Zarezadeh M, Mohee B, Abidin A, *et al.* zkRNN: zero-knowledge proofs for recurrent neural network inference. *Cryptology ePrint Archive* 2026.

[53] Abbaszadeh K, Pappas C, Katz J, Papadopoulos D. Zero-knowledge proofs of training for deep neural networks. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, Salt Lake City, USA, October 14–18, 2024, pp. 4316–4330.

[54] Garg S, Goel A, Jha S, Mahloujifar S, Mahmoody M, *et al.* Experimenting with zero-knowledge proofs of training. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, Copenhagen, Denmark, November 26–30, 2023, pp. 1880–1894.

[55] Keršič V, Karakatič S, Turkanović M. On-chain zero-knowledge machine learning: an overview and comparison. *J. King Saud Univ. Comput. Inf. Sci.* 2024, 36(9):102207.

[56] Peng Z, Zhao C, Wang T, Liao G, Lin Z, *et al.* A survey of zero-knowledge proof based verifiable machine learning. *Artif. Intell. Rev.* 2026, 59:157.

[57] Xing Z, Zhang Z, Zhang Z, Li Z, Li M, *et al.* Zero-knowledge proof-based verifiable decentralized machine learning in communication network: a comprehensive survey. *IEEE Commun. Surv. Tutor.* 2025, 28:985–1024.

[58] Conway K, So C, Yu X, Wong K. OPML: optimistic machine learning on blockchain. *arXiv* 2024, arXiv:2401.17555.

[59] So C, Conway K, Yu X, Yao S, Wong K. Opp/ai: optimistic privacy-preserving AI on blockchain. *arXiv* 2024, arXiv:2402.15006.

[60] Tramèr F, Boneh D. Slalom: fast, verifiable and private execution of neural networks in trusted hardware. *arXiv* 2019, arXiv:1806.03287.

[61] Truong JB, Gallagher W, Guo T, Walls RJ. Memory-efficient deep learning inference in trusted execution environments. In *Proceedings of the 2021 IEEE International Conference on Cloud Engineering (IC2E)*, San Francisco, USA, October 4–8, 2021, pp. 161–167.

[62] Papafragkaki K, Vasiliadis G. InferONNX: practical and privacy-preserving machine learning inference using trusted execution environments. In *Proceedings of the 22nd International Conference, DIMVA 2025*, Graz, Austria, July 9–11, 2025, pp. 25–43.

[63] Chaudhuri A, Shukla S, Bhattacharya S, Mukhopadhyay D. Secured and privacy-preserving GPU-based machine learning inference in trusted execution environment: a comprehensive survey. In *Proceedings of the 2025 17th International Conference on COMmunication Systems and NETworks (COMSNETS)*, Bengaluru, India, January 6–10, 2025, pp. 207–216.

[64] bAI Fund. TEE-executed on-chain agent fund. 2026. Available: https://baifund.io/ (accessed on 9 May 2026).

[65] Mind network. HTTPZ protocol for fully encrypted AI. 2026. Available: https://mindnetwork.xyz (accessed on 9 May 2026).

[66] Balan K, Learney R, Wood T. A framework for cryptographic verifiability of end-to-end AI pipelines. In *Proceedings of the 10th ACM International Workshop on Security and Privacy Analytics*, Pittsburgh, USA, June 6, 2025, pp. 49–59.

[67] Bruschi F, Esposito M, Gagliardoni T, Rizzini A. SoK: verifiable federated learning. *Cryptology ePrint Archive* 2025.

[68] Andreoletti D, Rudi A, Carpanzano E, Leidi T. Privacy-preserving LLM inference in practice: a comparative survey of techniques, trade-offs, and deployability. *Cryptology ePrint Archive* 2026.

[69] Sandford R, Siri L, Tirosh D, Weiss Y, Forshtat A, *et al.* ERC-2771: secure protocol for native meta transactions. 2020. Available: https://eips.ethereum.org/EIPS/eip-2771 (accessed on 9 May 2026).

[70] Wilson S, Dietrichs A, Garnett M, Zoltu M. EIP-3074: AUTH and AUTHCALL opcodes. 2020. Available: https://eips.ethereum.org/EIPS/eip-3074 (accessed on 9 May 2026).

[71] Weiss Y, Tirosh D, Forshtat A, Nacson S. ERC-7562: account abstraction validation scope rules. 2023. Available: https://eips.ethereum.org/EIPS/eip-7562 (accessed on 9 May 2026).

[72] Salem M, Rosario L, Cusack W, Tirosh D, Moxey J, *et al.* EIP-5792: wallet call API. 2022. Available: https://eips.ethereum.org/EIPS/eip-5792 (accessed on 9 May 2026).

[73] Vectorized, Moxey J, Croubois H. ERC-7821: minimal batch executor interface. 2024. Available: https://eips.ethereum.org/EIPS/eip-7821 (accessed on 9 May 2026).

[74] Buterin V, Weiss Y, Tirosh D, Nacson S, Forshtat A. ERC-7766: signature aggregation for ERC-4337. 2024. Available: https://eips.ethereum.org/EIPS/eip-7766 (accessed on 9 May 2026).

[75] Rosario L, Swenberg C, Hodges A, Bhandari P, Moxey J. ERC-7836: wallet call preparation API. 2024. Available: https://eips.ethereum.org/EIPS/eip-7836 (accessed on 9 May 2026).

[76] Rosario L, Tirosh D, Cusack W, Gazso K, Jumali H. ERC-7677: paymaster web service capability. 2024. Available: https://eips.ethereum.org/EIPS/eip-7677 (accessed on 9 May 2026).

[77] Erinle Y, Kethepalli Y, Feng Y, Xu J. SoK: design, vulnerabilities, and security measures of cryptocurrency wallets. *Comput. Networks* 2025, 273:111691.

[78] Praitheeshan P, Pan L, Doss R. Security evaluation of smart contract-based on-chain ethereum wallets. In *Proceedings of the 14th International Conference, NSS 2020*, Melbourne, Australia, November 25–27, 2020, pp. 22–41.

[79] Borge M, Kokoris-Kogias E, Jovanovic P, Gasser L, Gailly N, *et al.* Proof-of-personhood: redemocratizing permissionless cryptocurrencies. In *Proceedings of the 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, Paris, France, April 26–28, 2017, pp. 23–26.

[80] Siddarth D, Ivliev S, Siri S, Berman P. Who watches the watchmen? A review of subjective approaches for sybil-resistance in proof of personhood protocols. *Front. Blockchain* 2020, 3:590171.

[81] Chang W, Rocco G, Millegan B, Johnson N, Terbu O. ERC-4361: sign-in with Ethereum. 2021. Available: https://eips.ethereum.org/EIPS/eip-4361 (accessed on 9 May 2026).

[82] Terbu O, Ward J, Lehner C, Gbafa S, Chang W, *et al.* ERC-5573: sign-in with Ethereum capabilities, ReCaps. 2021. Available: https://eips.ethereum.org/EIPS/eip-5573 (accessed on 9 May 2026).

[83] Dennis JB, Van Horn EC. Programming semantics for multiprogrammed computations. *Commun. ACM* 1966, 9(3):143–155.

[84] Nakamura Y, Zhang Y, Sasabe M, Kasahara S. Exploiting smart contracts for capability-based access control in the internet of things. *Sensors* 2020, 20(6):1793.

[85] Xu R, Chen Y, Blasch E, Chen G. BlendCAC: a smart contract enabled decentralized capability-based access control mechanism for the iot. *Computers* 2018, 7(3):39.

[86] Ali G, Ahmad N, Cao Y, Asif M, Cruickshank H, *et al.* Blockchain based permission delegation and access control in Internet of Things (BACI). *Comput. Secur.* 2019, 86:318–334.

[87] Zhang Y, Kasahara S, Shen Y, Jiang X, Wan J. Smart contract-based access control for the internet of things. *IEEE Internet Things J.* 2019, 6(2):1594–1605.

[88] Gao Y, Zhang A, Wu S, Chen J. Blockchain-based multi-hop permission delegation scheme with controllable delegation depth for electronic health record sharing. *High-Confid. Comput.* 2022, 2(4):100084.

[89] Sherazi SNA, Zahoor E, Akhtar S, Perrin O. A blockchain based approach for the authorization policies delegation in emergency situations. *Trans. Emerging Telecommun. Technol.* 2022, 33(5):e4461.

[90] Nie S, Ren J, Wu R, Han P, Han Z, *et al.* Zero-trust access control mechanism based on blockchain and inner-product encryption in the internet of things in a 6G environment. *Sensors* 2025, 25(2):550.

[91] zeroknots, Kopp K, Lee T, Makarov F, Poon E, *et al.* ERC-7579: minimal modular smart accounts. 2023. Available: https://eips.ethereum.org/EIPS/eip-7579 (accessed on 9 May 2026).

[92] Egyed A, Liu F, Paik J, Weiss Y, Gu H, *et al.* ERC-6900: modular smart contract accounts. 2023. Available: https://eips.ethereum.org/EIPS/eip-6900 (accessed on 9 May 2026).

[93] zeroknots, Kopp K, Lee T, Makarov F. ERC-7780: validation module extension for ERC-7579. 2024. Available: https://eips.ethereum.org/EIPS/eip-7780 (accessed on 9 May 2026).

[94] Kopp K, zeroknots. ERC-7484: registry extension for ERC-7579. 2023. Available: https://eips.ethereum.org/EIPS/eip-7484 (accessed on 9 May 2026).

[95] foobar, Chung W, ryley o, Rockland J, andy8052. ERC-5639: delegation registry. 2022. Available: https://eips.ethereum.org/EIPS/eip-5639 (accessed on 9 May 2026).

[96] Offerijns J, Martins J. ERC-7741: authorize operator. 2024. Available: https://eips.ethereum.org/EIPS/eip-7741 (accessed on 9 May 2026).

[97] Isailovic L, Rein D, Finlay D, Chiang D, Makarov F, *et al.* ERC-7715: request permissions from wallets. 2024. Available: https://eips.ethereum.org/EIPS/eip-7715 (accessed on 9 May 2026).

[98] Quantstamp. Alchemy modular account—audit report. 2023. Available: https://certificate.quantstamp.com/full/alchemy-modular-account/2c13aab2-5d5f-4f45-a9ef-b890bdb12b97/index.html (accessed on 9 May 2026).

[99] Nagesh M. Demystifying the intent-centric thesis. 2023. Available: https://public.bnbstatic.com/static/files/research/demystifying-the-intent-centric-thesis.pdf (accessed on 9 May 2026).

[100] hellohanchen. ERC-7806: minimal intent-centric EOA smart account. 2024. Available: https://eips.ethereum.org/EIPS/eip-7806 (accessed on 9 May 2026).

[101] Goodary K, Wallet P, Wickens L, Khoury R. ERC-7845: universal orchestrator RPC. 2024. Available: https://eips.ethereum.org/EIPS/eip-7845 (accessed on 9 May 2026).

[102] Ma R, Tang W, Yao D. Analysis of the order flow auction under proposer-builder separation on blockchain. *arXiv* 2025, arXiv:2502.12026.

[103] Bahrani M, Garimidi P, Roughgarden T. Transaction fee mechanism design in a post-mev world. *Cryptology ePrint Archive* 2024.

[104] Rasheed, Desai PN, Chaurasia Y, Gujar S. Shapley value-based approach for distributing revenue of matchmaking of private transactions in blockchains. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, Detroit, USA, May 19–23, 2025, pp. 2723–2725.

[105] Bachu B, Wan X, Moallemi CC. Quantifying price improvement in order flow auctions. In *Proceedings of the CAAW 2025 and WTSC 2025*, Miyakojima, Japan, April 18, 2025, pp. 111–126.

[106] Watts A, Sinesi D, Greene J. Using "failure costs" to guarantee execution quality in order flow auctions. *arXiv* 2025, arXiv:2503.05338.

[107] Zhang M, Yang S, Zhang F. RediSwap: MEV redistribution mechanism for CFMMs. In *Proceedings of the 2025 Workshop on Decentralized Finance and Security*, Taipei, China, October 13–17, 2025, pp. 27–36.

[108] Appiah B, Commey D, Bagyl-Bac W, Adjei L, Owusu E. Game-theoretic analysis of MEV attacks and mitigation strategies in decentralized finance. *Analytics* 2025, 4(3):23.

[109] Olas. Autonomous agent services. 2026. Available: https://olas.network (accessed on 9 May 2026).

[110] AgentFi. On-chain AI agents as NFTs. 2026. Available: https://agentfi.io (accessed on 9 May 2026).

[111] Maiga. AI trading agent on BNB chain. 2026. Available: https://www.maiga.ai/ (accessed on 9 May 2026).

[112] Windle J, Giang B, Jang S, Downs D, Huynh R, *et al.* ERC-6551: non-fungible token bound accounts. 2023. Available: https://eips.ethereum.org/EIPS/eip-6551 (accessed on 9 May 2026).

[113] Conway, So C, Yu X, Yao S, Kartin. ERC-7641: intrinsic RevShare token. 2024. Available: https://eips.ethereum.org/EIPS/eip-7641 (accessed on 9 May 2026).

[114] Khan A, Matyana A, Gorin B, Bhayani V. ERC-7649: bonding curve-embedded liquidity for NFTs. 2024. Available: https://eips.ethereum.org/EIPS/eip-7649 (accessed on 9 May 2026).

[115] Mafrur R. AI-based crypto tokens: the illusion of decentralized AI? *IET Blockchain* 2025, 5(1):e70015.

[116] Virtuals Protocol. Agent commerce protocol. 2026. Available: https://www.virtuals.io/ (accessed on 9 May 2026).

[117] Recall. Decentralized agent skill markets. 2026. Available: https://recall.network (accessed on 9 May 2026).

[118] Sockin M, Xiong W. Decentralization through tokenization. *J. Finance* 2023, 78(1):247–299.

[119] Kivilo S, Norta A, Hattingh M. Token economy incentive structure conceptual goal model. In *Proceedings of the 2024 IST-Africa Conference (IST-Africa)*, Dublin, Ireland, May 20–24, 2024, pp. 1–9.

[120] Kivilo S, Norta A, Hattingh M, Avanzo S, Pennella L. Designing a token economy: incentives, governance, and tokenomics. *arXiv* 2026, arXiv:2602.09608.

[121] Borjigin A, Zhou W, He C. AI-governed agent architecture for web-trustworthy tokenization of alternative assets. *arXiv* 2025, arXiv:2507.00096.

[122] Zhang T. Constructing a decentralized AI data marketplace enabled by a blockchain-based incentive mechanism. *J. Ind. Eng. Appl. Sci.* 2025, 3(3):42–46.

[123] Jackson F. Agentic blockchain intelligence: designing secure decentralized AI agents with smart contract governance. *SSRN* 2025, SSRN:5819422.

[124] Wei Z, Sun J, Sun Y, Liu Y, Wu D, *et al.* Advanced smart contract vulnerability detection via LLM-powered multi-agent systems. *IEEE Trans. Softw. Eng.* 2025, 51(10):2830–2846.

[125] Liu Y, Xue Y, Wu D, Sun Y, Li Y, *et al.* PropertyGPT: LLM-driven formal verification of smart contracts through retrieval-augmented property generation. *arXiv* 2025, arXiv:2405.02580.

[126] Yuan Y, Wang Y, Xu Y, Yahn Z, Hu S, *et al.* Large language model based smart contract auditing with LLMBugScanner. *arXiv* 2025, arXiv:2512.02069.

[127] Hu S, Huang T, İlhan F, Tekin SF, Liu L. Large language model-powered smart contract vulnerability detection: new perspectives. In *Proceedings of the 2023 5th IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, Atlanta, USA, November 1–4, 2023, pp. 297–306.

[128] Luo F, Luo R, Chen T, Qiao A, He Z, *et al.* SCVHunter: smart contract vulnerability detection based on heterogeneous graph attention network. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, Lisbon, Portugal, April 14–20, 2024, pp. 1–13.

[129] Tang X, Du Y, Lai A, Zhang Z, Shi L. Deep learning-based solution for smart contract vulnerabilities detection. *Sci. Rep.* 2023, 13(1):20106.

[130] Deng W, Wei H, Huang T, Cao C, Peng Y, *et al.* Smart contract vulnerability detection based on deep learning and multimodal decision fusion. *Sensors* 2023, 23(16):7246.

[131] Huang J, Zhou K, Xiong A, Li D. Smart contract vulnerability detection model based on multi-task learning. *Sensors* 2022, 22(5):1829.

[132] Almaghthawi A, Yafooz WM, Albalawi NS. Three large language models for solidity smart contract vulnerability detection. *J. Artif. Intell. Technol.* 2025, 5:314–322.

[133] Alsunaidi SJ, Aljamaan H, Hammoudeh M. Leveraging machine learning models to improve smart contract security: a survey of vulnerabilities and detection methods. *ACM Comput. Surv.* 2025, 58(6):1–37.

[134] Ivanov N, Li C, Yan Q, Sun Z, Cao Z, *et al.* Security threat mitigation for smart contracts: a comprehensive survey. *ACM Comput. Surv.* 2023, 55(14s):1–37.

[135] Crisostomo J, Bacao F, Lobo V. Machine learning methods for detecting smart contracts vulnerabilities within Ethereum blockchain—a review. *Expert Syst. Appl.* 2025, 268:126353.

[136] Jiang F, Chao K, Xiao J, Liu Q, Gu K, *et al.* Enhancing smart-contract security through machine learning: a survey of approaches and techniques. *Electronics* 2023, 12(9):2046.

[137] Liu D, Zhang J, Wang Y, Shen H, Zhang Z, *et al.* Blockchain smart contract security: threats and mitigation strategies in a lifecycle perspective. *ACM Comput. Surv.* 2025, 58(4):1–34.

[138] Wang B, Tong Y, Ji S, Dong H, Luo X, *et al.* a review of learning-based smart contract vulnerability detection: a perspective on code representation. *ACM Trans. Softw. Eng. Methodol.* 2025, 35(6):1–46.

[139] Luo B, Zhang Z, Wang Q, Ke A, Lu S, *et al.* AI-powered fraud detection in decentralized finance: a project life cycle perspective. *ACM Comput. Surv.* 2024, 57(4):1–38.

[140] Mounnan O, Manad O, Boubchir L, El Mouatasim A, Daachi B. A review on deep anomaly detection in blockchain. *Blockchain: Res. Appl.* 2024, 5(4):100227.

[141] Wang J, Bigger A, Xu X, Lin JW, Applebaum A, *et al.* EVMbench: evaluating AI Agents on smart contract security. *arXiv* 2026, arXiv:2603.04915.

[142] anajuliabit. Add MIP-X43: activate OEV wrappers for all remaining markets. 2026. Available: https://github.com/moonwell-fi/moonwell-contracts-v2/pull/578 (accessed on 9 May 2026).

[143] Rizal S, Kim DS. Enhancing blockchain consensus mechanisms: a comprehensive survey on machine learning applications and optimizations. *Blockchain: Res. Appl.* 2025, 6(4):100302.

[144] Li J, Xie L, Huang H, Zhou B, Song B, *et al.* Survey on strategic mining in blockchain: a reinforcement learning approach. *arXiv* 2025, arXiv:2502.17307.

[145] Gutierrez R, Villegas-Ch W, Govea J. Adaptive consensus optimization in blockchain using reinforcement learning and validation in adversarial environments. *Front. Artif. Intell.* 2025, 8:1672273.

[146] Islam T, Bappy FH, Zaman TS, Sajid MSI, Pritom MMA. MRL-PoS: a multi-agent reinforcement learning based proof of stake consensus algorithm for blockchain. In *Proceedings of the 2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, USA, January 8–10, 2024, pp. 0409–0413.

[147] Zou Y, Jin Z, Zheng Y, Yu D, Lan T. Optimized consensus for blockchain in internet of things networks via reinforcement learning. *Tsinghua Sci. Technol.* 2023, 28(6):1009–1022.

[148] Venkatesan K, Rahayu SB. Blockchain security enhancement: an approach towards hybrid consensus algorithms and machine learning techniques. *Sci. Rep.* 2024, 14(1):1149.

[149] Valko D, Kudenko D. Sustainable broadcasting in blockchain network with reinforcement learning. *arXiv* 2025, arXiv:2407.15616.

[150] Bravo-Marquez F, Reeves S, Ugarte M. Proof-of-learning: a blockchain consensus mechanism based on machine learning competitions. In *Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, Newark, USA, April 4–9, 2019, pp. 119–124.

[151] Cai L, Liu A, Yan Y. Blockchain consensus algorithm for supply chain information security sharing based on convolutional neural networks. *Sci. Rep.* 2025, 15(1):33916.

[152] You J. Curvetime: a blockchain framework for artificial intelligence computation. *Software Impacts* 2022, 13:100314.

[153] Blanchard P, El Mhamdi EM, Guerraoui R, Stainer J. Machine learning with adversaries: byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, USA, December 4–9, 2017, pp. 118–128.

[154] Yin D, Chen Y, Ramchandran K, Bartlett P. Byzantine-robust distributed learning: towards optimal statistical rates. In *Proceedings of the International conference on machine learning*, Stockholm, Sweden, July 10–15, 2018, pp. 5650–5659.

[155] Cao X, Fang M, Liu J, Gong NZ. FLTrust: byzantine-robust federated learning via trust bootstrapping. *arXiv* 2021, arXiv:2012.13995.

[156] El Mhamdi EM, Guerraoui R, Rouault S. The hidden vulnerability of distributed learning in byzantium. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Stockholm, Sweden, July 10–15, 2018, pp. 3521–3530.

[157] Fischer MJ, Lynch NA, Paterson MS. Impossibility of distributed consensus with one faulty process. *J. ACM* 1985, 32(2):374–382.

[158] Lee KW, Ewe HT. Performance study of byzantine agreement protocol with artificial neural network. *Inf. Sci.* 2007, 177(21):4785–4798.

[159] Bouhata D, Moumen H, Mazari JA, Bounceur A. Byzantine fault tolerance in distributed machine learning: a survey. *J. Exp. Theor. Artif. Intell.* 2025, 37(8):1331–1389.

[160] Ding Q, Liebau D, Wang Z, Xu W. A survey on decentralized autonomous organizations (DAOs) and their governance. *World Sci. Annu. Rev. FinTech* 2023, 1:2350001.

[161] Tang C, Cai Q, Dong C, Wang Q, Chen S. Decentralized autonomous organizations (DAOs): an exploratory survey. *Distrib. Ledger Technol.* 2025, 5(1):1–25.

[162] Jungnickel M, Özdemir Sönmez F, Mulligan C, Knottenbelt WJ. DAO governance: voting power, participation, and controversy - a review and an empirical analysis. *Distrib. Ledger Technol.* 2025, 5(4):1–25.

[163] Cong LW, Rabetti D, Wang CC, Yan Y. Centralized governance in decentralized organizations. *SSRN* 2025, SSRN:5168660.

[164] Messias J, Pahari V, Chandrasekaran B, Gummadi KP, Loiseau P. Understanding blockchain governance: analyzing decentralized voting to amend DeFi smart contracts. *arXiv* 2023, arXiv:2305.17655.

[165] Li S, Chen Y. Governing decentralized autonomous organizations as digital commons. *J. Bus. Ventur. Insights* 2024, 21:e00450.

[166] Li J, Qin R, Guan S, Ding W, Lin F, *et al.* Attention markets of blockchain-based decentralized autonomous organizations. *IEEE/CAA J. Autom. Sin.* 2024, 11(6):1370–1380.

[167] Capponi A, Gliozzo A, Han C, Lee J. DAO-AI: evaluating collective decision-making through agentic AI in decentralized governance. *arXiv* 2025, arXiv:2510.21117.

[168] Xia Y, Wang T, Xu W, Zhang S. DAO-agent: zero knowledge-verified incentives for decentralized multi-agent coordination. *arXiv* 2025, arXiv:2512.20973.

[169] Morar CD, Popescu DE, Novac OC, Ghiurău D. Rethinking blockchain governance with AI: the VOPPA framework. *Computers* 2025, 14(10):425.

[170] Chaffer TJ, von Goins II C, Cotlage D, Okusanya B, Goldston J. Decentralized governance of autonomous AI agents (ETHOS). *arXiv* 2024, arXiv:2412.17114.

[171] Lalley SP, Weyl EG. Quadratic voting: how mechanism design can radicalize democracy. *AEA Pap. Proc.* 2018, 108:33–37.

[172] Benhaim A, Falk BH, Tsoukalas G. Balancing power in decentralized governance: quadratic voting and information aggregation. *Manage. Sci.* 2026, 72(6):4597–4609.

[173] Austgen J, Fábrega A, Allen S, Babel K, Kelkar M, *et al.* DAO decentralization: voting-bloc entropy, bribery, and dark DAOs. *arXiv* 2023, arXiv:2311.03530.

[174] Tamai S, Kasahara S. DAO voting mechanism resistant to whale and collusion problems. *Front. Blockchain* 2024, 7:1405516.

[175] Kahng A, Mackenzie S, Procaccia A. Liquid democracy: an algorithmic perspective. *J. Artif. Intell. Res.* 2021, 70:1223–1252.

[176] Berinsky AJ, Halpern D, Halpern JY, Jadbabaie A, Mossel E, *et al.* Tracking truth with liquid democracy. *Manage. Sci.* 2025, 71(9):7821–7839.

[177] Casella A, Campbell J, de Lara L, Mooers V, Ravindran D. Liquid democracy. Two experiments on delegation in voting. *SSRN* 2022.

[178] Weidener L, Laredo F, Kumar K, Compton K. Delegated voting in decentralized autonomous organizations: a scoping review. *Front. Blockchain* 2025, 8:1598283.

[179] Novelli C, Floridi L, Sartor G, Teubner G. AI as legal persons: past, patterns, and prospects. *J. Law Soc.* 2025, 52(4):533–555.

[180] Forrest KB. The ethics and challenges of legal personhood for AI. *Yale LJF* 2023, 133:1175.

[181] Baeyaert J. Beyond personhood: the evolution of legal personhood and its implications for AI recognition. *Technol. Regul.* 2025, 2025:355–386.

[182] Chystiakov A, Kurbatov O, Panasenko Y, Elliot M, Buterin V. ERC-7812: ZK identity registry. 2024. Available: https://eips.ethereum.org/EIPS/eip-7812 (accessed on 9 May 2026).

[183] García E. ERC-7964: Crosschain EIP-712 signatures. 2025. Available: https://eips.ethereum.org/EIPS/eip-7964 (accessed on 9 May 2026).

[184] Makeig P. ERC-8121: cross-chain function calls via hooks. 2025. Available: https://eips.ethereum.org/EIPS/eip-8121 (accessed on 9 May 2026).
[185] Aura. Decentralized AI model verification market. 2026. Available: https://www.aura.fun/ (accessed on 9 May 2026).
[186] MyShell. Decentralized AI consumer layer. 2026. Available: https://myshell.ai (accessed on 9 May 2026).
[187] DIN. AI agent blockchain. 2026. Available: https://din.lol (accessed on 9 May 2026).
[188] Billy Bets AI. Autonomous sports betting agent. 2026. Available: https://billybets.ai (accessed on 9 May 2026).
[189] Creator.Bid. AI agent launchpad. 2026. Available: https://creator.bid (accessed on 9 May 2026).
[190] GOAT. AI gaming platform. 2026. Available: https://goatgaming.com/ (accessed on 9 May 2026).
[191] HeyElsa. AI crypto co-pilot. 2026. Available: https://www.heyelsa.ai/ (accessed on 9 May 2026).
[192] iAgent. DePIN gaming AI agent. Available: https://www.iagentpro.com/ (accessed on 9 May 2026).
[193] Lazbubu. AI companion agent on BNB chain. Available: https://lazbubu.ai/ (accessed on 9 May 2026).
[194] Sentient AI. Empathetic agent on Sui. Available: https://aisentient.net/ (accessed on 9 May 2026).
[195] WORLD3. AI x blockchain platform. 2026. Available: https://world3.ai/ (accessed on 9 May 2026).
[196] RootData. A reliable crypto asset data platform. 2026. Available: https://www.rootdata.com/Fundraising (accessed on 9 May 2026).
[197] CoinMarketCap. The world's largest cryptocurrency price-tracking platform. 2026. Available: https://coinmarketcap.com/ (accessed on 9 May 2026).