DEEPTRUST: VERIFIABLE IDENTITIES AND REPUTATION FOR AI AGENTS

TECHNICAL REPORT

Sebastian Rodriguez, Oleksandr Brezhniev, Dmytro Sukhiy

Privado ID
`{srodriguez, oleksandr, dmytro}@privado.id`

March 27, 2025

ABSTRACT

The rapid proliferation of artificial intelligence (AI) agents across domains necessitates a robust trust framework rooted in a multidimensional understanding of AI agent identity. This paper argues that trust in AI systems must be derived from the relationships these agents establish with other entities in a reputation system, including humans, organizations, and other AI agents. A trust framework requires a comprehensive identity model that incorporates architectural, behavioral, legal, and social dimensions, each contributing to the agent's reputation within a decentralized ecosystem.

At the foundation of any trust system lies identity, as having verifiable identifiers for agents is critical for enabling accountability, interoperability, and governance. This paper highlights the necessity of establishing unique, verifiable identifiers for AI agents, which can anchor reputation systems and facilitate trustworthy interactions in both human-to-agent (H2A) and agent-to-agent (A2A) scenarios. Without such identifiers, the reliability of attestations and the integrity of the overall trust framework cannot be ensured.

We provide a comparative analysis of different identity solutions for AI agents, focusing on their ability to provide robust identifiers and to express agent attributes. The analysis spans key-based solutions such as Decentralized Identifiers (DIDs), blockchain addresses, and X.509 certificates, examining their strengths, limitations, and applicability in decentralized environments. Key challenges such as protecting private keys from malicious developers, ensuring attribute consistency, and maintaining identifier validity across changes in agent behavior or architecture are discussed.

This study proposes a hybrid approach that leverages DIDs and public on-chain attestations, addressing limitations of existing systems. By combining public attestations with privacy-preserving identity wallets and Verifiable Credentials (VCs), this approach allows agents to maintain trust while safeguarding sensitive information. Zero-Knowledge Proofs (ZKPs) play a pivotal role in this system, enabling selective disclosure of credentials and unlinkable attestations. This ensures compliance with privacy and consent requirements, particularly in scenarios involving human-targeted attestations.

The proposed solution establishes a scalable and privacy-preserving framework for AI agent identity and trust. By integrating decentralized technologies like DIDs and ZKPs, and supporting flexible verification methods, this framework provides a future-proof foundation for the responsible deployment of AI agents in diverse and dynamic ecosystems.

Keywords ai agents, decentralized identifiers, verifiable credentials, self-sovereign identity, zero-knowledge proofs

1 Introduction

As artificial intelligence (AI) agents proliferate across domains from customer support to autonomous driving, their integration into human-centered and machine-centric ecosystems presents profound opportunities. However, the lack of verifiable identities for these agents hampers trust, restricts accountability, and introduces vulnerabilities in scenarios where integrity is crucial.

In human-to-agent (H2A) interactions—where users rely on AI for financial advice or medical insights—and in agent-to-agent (A2A) contexts—where autonomous systems coordinate logistics or execute trades— trust is indispensable [26].

The concept of trustworthy AI agents is multi-faceted and can be viewed from several perspectives [24], depending on the stakeholders and the context. Here are some common perspectives [25]:

Perspective	Scope	How trust is established
Trust the Data	**Data Collection:** Transparency in how data is sourced, ensuring it is legal, ethical, and unbiased. **Data Quality:** Ensuring completeness, accuracy, consistency, and timeliness of data. **Data Privacy:** Respecting user confidentiality and complying with privacy regulations. **Data Representation:** Avoiding bias through balanced and inclusive datasets.	Third-party audits Self-attestation zkML
Trust the training process	**Algorithmic Transparency:** Documenting the methodologies used in training AI systems. **Bias Mitigation:** Identifying and eliminating biases within training datasets and algorithms. **Ethical Oversight:** Establishing governance mechanisms to ensure ethical decisions during the training phase. **Validation and Testing:** Employing rigorous testing methods to verify accuracy and generalizability. **Explainable AI (XAI):** Developing models that can articulate their learning process and decisions.	Open source models Third-party audits Standardized testing Trusted Execution Environments zkML
Trust the outcome	**Accuracy and Precision:** Validating that outcomes are consistent, reliable, and meet predefined goals. **Fairness and Non-Discrimination:** Ensuring decisions are equitable across diverse groups. **Explainability:** Providing clear justifications for decisions to enhance user understanding. **User Feedback:** Incorporating feedback to refine outcomes and maintain relevance. **Auditability:** Creating mechanisms for retrospective analysis of decisions and actions. **Ethicality:** Prevent deception and maintain liability of the AI agent outcomes. [15]	Third-party audits Standardized testing Explainable AI
Trust the owner	**Accountability:** Taking responsibility for AI actions, especially in cases of failure or harm. **Ethical Leadership:** Embedding ethical principles in AI governance and operations. **Compliance and Regulation:** Adhering to legal and industry standards. **User Empowerment:** Providing users with transparency, control, and recourse mechanisms.	Legal ownership Digital Certificates IP Protection laws

There is an underlying assumption in all these mechanisms (institutional, software, or hardware-based):

It is possible for an **identifiable entity (1)** (software, person, organization) to make a **verifiable claim (2)** about this particular **AI agent (3)** [3].

(1) The identification of claim issuers through Self-Sovereign identities is straightforward, with existing market solutions (such as Privado ID) already implementing this through decentralized identifiers (DID) and verifiable credentials (VC).

(2) Verifiable data typically relies on cryptographic signatures using asymmetric keys. This includes both Verifiable Credentials and on-chain attestations signed with private keys corresponding to crypto addresses.

(3) The key challenge in AI agent identity frameworks today lies in precisely identifying “this particular AI Agent.” This requires meeting three essential conditions:

A. The agent’s identity must have a **unique identifier**.
B. There must be a mechanism to prove the identifier is solely controlled by the agent, preventing impersonation.
C. The identifier’s validity must reflect the **Agent Identity accurately**. This raises the “Ship of Theseus” paradox—when changes in an AI Agent’s data, model, or outcomes alter its identity, should it maintain the same identifier?

This paper aims to define an identity model for AI Agents that addresses points (1), (2), and (3), establishing the foundation for all forms of AI reputation discussed above.

2 AI Agent Identity & Identifiers

2.1 AI Identity Framework

Identity comprises the qualities, properties, and characteristics that make an entity **recognizable as itself and distinct from other entities** [5]. Since identifiers represent these identities, we must first define "what is the identity of an AI Agent?" before we can determine what these identifiers represent.

Understanding AI agent identity involves 4 primary approaches:

2.1.1 Architectural Identity

Ensures the identity of the agent is tied to its computational and structural foundation, independent of behavior [25]. Its key components are:

- **Model Architecture:** The neural network structure, base LLM, etc.
- **Weights/Parameters:** Learned parameters that define the model’s function.
- **Codebase and Versioning:** Source code used to define the agent, including version control.
- **Cryptographic Fingerprint:** A hash of the model and weights for immutable identification.
- **Execution Environment:** Runtime attributes like hardware specifications, containers, or Trusted Execution Environments (TEEs).

The verifiability of these elements is achieved with cryptographic hashing of the models and weights, digital signatures of the execution logs, and verifiable identifiers like Decentralized Identifiers (DIDs).

These Identifiers (DIDs) are based on the premise that the controller of the DID is in possession of a private key, that uses to prove its control of the identifier.

The private key is not part of the training data, but is usually managed by external secure systems (TEE, HSM, MPC) [7] that provide the agent with signature capabilities, which means that they act as possession authentication factors (something you have) rather than knowledge factors (something that you know).

In this approach, the connection between identifier and AI Agent behavior is not guaranteed: even if the model, data, or environment changes, the identifier remains valid as long as the AI Agent maintains control over it.

The biggest risk of this approach is that if other entities are making public claims (like certifying the correctness or security of the agent) by referring to a static identifier, the agent may change its behavior and the claims would still be valid. We discuss some strategies to reduce this risk, like claim revocation and expiration in section 3.4.

2.1.2 Behavioral Identity

Captures the agent’s dynamic identity, ensuring that its behavior aligns with its architectural identity and intended design. Its key components are:

- **Input-Output Mappings:** The relationship between input data and the agent’s responses.

- **Behavioral Signatures:** Cryptographic proofs (e.g., zkML) linking behavior to architectural identity. [27]
- **Stochastic Elements:** Randomness in outputs (e.g., temperature in language models) [8].
- **Adaptive Changes:** The impact of fine-tuning or reinforcement learning on behavior over time.

The verification methods include behavioral reproducibility tests with controlled inputs and behavioral fingerprints derived from standardized test cases.

The behavior of an AI agent cannot be fully predicted using just its weights and model, even with constant input. While weights and the model define the agent's logic and learned patterns, behavior depends on additional factors such as the execution environment, hardware precision, and stochastic processes like random initialization or sampling.

Most real-world agents incorporate adaptive mechanisms, randomness, or external decision-making rules, making behavior emergent rather than static. Thus, the interaction between the model, weights, and runtime context is essential to understanding and predicting an agent's behavior accurately.

In this case, the identifier should only be produced and controlled by an agent with specific behavioral characteristics (something the agent IS). If the agent behavior changes, the agent should lose its ability to prove control over the identifier.

An example of this would be the use of a zkML based identifier, but this is not feasible today. Zero-knowledge machine learning (zkML) is advancing but is not yet fully production-ready for large language models (LLMs). The computational demands of LLMs, combined with the overhead introduced by zero-knowledge proofs, pose significant hurdles. Current zkML implementations are more suited to smaller models and specific applications [28].

While exact timelines are uncertain, the rapid pace of research and development suggests that zkML could become feasible for production use with LLMs within the next 3 to 5 years. This projection depends on breakthroughs in optimizing zero-knowledge proof systems and enhancing the efficiency of zkML frameworks [13, 23].

2.1.3 Legal Identity

The formalized identity of the AI agent as recognized by legal or regulatory frameworks, linking it to its developers, owners, or operators. Provides a legal basis for trust and accountability, ensuring the agent is tied to human or organizational entities.

It is composed by:

- **Ownership Documentation:** Proof of ownership or control (e.g., digital certificates or blockchain attestations).
- **Legal Attestations:** Certifications or compliance with regulations (e.g., GDPR, ISO standards).
- **Accountability Frameworks:** Mechanisms that assign responsibility for the agent's actions to specific entities. [4]
- **Intellectual Property Protections:** Legal identifiers for the agent's code, model, and outputs.

Verification methods include digital certificates issued by trusted authorities, public or private registry entries linking the agent to its owners or creators and legal contracts or smart contracts tying the agent to its stakeholders.

2.1.4 Social Identity

The perception and reputation of the AI agent, influenced by external entities such as developers, trainers, and its branding. Establishes trustworthiness and societal acceptance [10] by reflecting external endorsements, trust, and reputation. Key components are:

- **Reputation Signals:** Reviews, certifications, or feedback from users and auditors.
- **Brand Recognition:** Public perception tied to the organization or platform deploying the agent.
- **Social Graphs:** Attestations or endorsements from other agents or humans (e.g., blockchain attestations, LinkedIn-style endorsements).
- **Transparency:** Publicly available information about the agent's performance, fairness, and ethics.

The verification methods include verifiable credentials stored in an identity wallet, public attestations on a blockchain or registry, and transparency reports and audits.

2.1.5 Framework Summary

Category	Definition	Key Components	Verification Methods	Identifiers Examples
Architectural Identity	Inherent, unchanging attributes like model and weights	Model architecture Weights Cryptographic fingerprints	Hashing DIDs Execution logs	DID or other PKI-based identifier
Behavioral Identity	Attributes linked to the agent's behavior and interactions	Input-output mappings Adaptive changes Stochastic elements	Behavioral tests Cryptographic logs Reproducibility proofs	zkML
Legal Identity	Recognition by legal or regulatory frameworks	Ownership documentation Legal attestations Accountability	Digital certificates Contracts Compliance certifications	x.509 certificates
Social Identity	Reputation and perception based on external endorsements	Reputation signals Brand recognition Transparency	Public attestations Blockchain credentials Audits	Relies on other identifiers to receive reputation signals

2.2 Key-based Identifiers for Architectural Agent Identities

Since the mechanisms to produce identifiers for behavioral agent identities are not ready for production environments, our solution will define identifiers for architectural agent identities.

The most mature and widely adopted mechanism for providing self-verifiable identifiers is to rely on PKI. Some examples:

- Self-resolvable DID methods (key-based, decentralized)
- Blockchain addresses (decentralized) [21]
- X.509 certificates (centralized, legally binding)

Using any of these identifiers would require the AI Agent to sign messages with a private key.

While this differs from direct key possession (as the agent could use an MPC network for decentralized signatures or other mechanisms like HSM modules), all signing mechanisms require an authenticator — essentially a proxy for the private key that anyone could use to sign.

We must assume developers can act maliciously. This raises the question: *"How can we protect agent keys from developers?"*

For simplicity and adoption, we recommend using established security components like HSM, KSM, or TEE to safeguard the agent's private keys. This security model choice should remain with individual developers rather than being standardized. Third parties can audit these decisions as part of the agent's reputation assessment [16].

However, the solution must prevent developers from accessing or using private keys in any operations related to the agent's identity and reputation setup or maintenance. For example:

- Developers should not prove agent ownership by using the agent's private keys.
- The agent's keys should not be used for on-chain transactions (such as purchasing ENS tokens), unless the agent can complete these without exposing its keys.

2.2.1 Digital Signature Algorithms

The generation and verification of key-based DIDs in AI Agents must be standardized and flexible enough to support multiple implementation strategies. This includes:

1. Support for popular DID methods following W3C standards.
2. Support for multiple formats in the response (text, QR code).
3. The signature verification is flexible - it can happen directly in the client (browser extensions, installed apps) or through a simple front-end (which could run on decentralized infrastructure like IPFS).

Supporting multiple DID methods opens the door to different signature algorithms and elliptic curves (ECDSA, EdDSA, RSA).

In order to maximize compatibility with existing systems (Ethereum, Bitcoin), our solution will work with ECDSA on the secp256k1 elliptic curve. Many projects are already working to provide crypto-wallets to AI Agents, which means these AI Agents will be able to sign messages and perform DID authentications.

2.3 Identifier Verifiability: DIDs

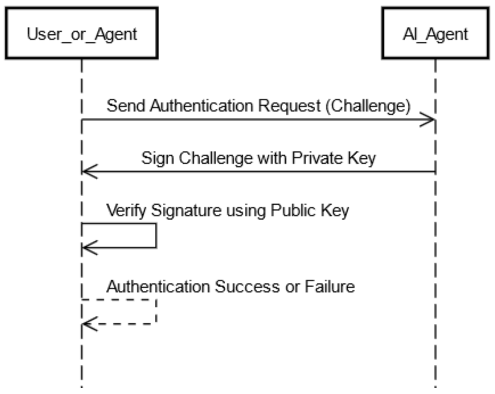
Decentralized Identifiers (DIDs) are ideal for AI agents as they offer self-sovereignty, interoperability, and cryptographic flexibility. Unlike blockchain addresses tied to specific platforms or X.509 certificates reliant on centralized authorities, DIDs are infrastructure-agnostic, portable, and support key rotation without breaking continuity. They enable privacy through selective disclosure of credentials, ensuring only necessary information is shared. DIDs operate on a decentralized trust model, making them resilient to single points of failure, and are cost-effective compared to blockchain operations or certificate re-issuance. These features make DIDs the most future-proof, adaptable, and secure identity solution for AI agents in diverse ecosystems [18, 17].

DIDs provide decentralized, self-verifiable identifiers backed by private keys. Using these keys, an AI agent can generate its own DID and prove control of it to humans or other agents without relying on any third party—all through a simple signed message.

Some DID methods (like `did:iden3`) support key rotation, enabling AI agents to update their controlling keys while maintaining the same DID identifier and preserving all associated claims.

2.3.1 DID Authentication

The DID authentication process is very simple and, for self-resolvable DID methods, it is a peer-to-peer operation that doesn't require a connection to any other system.

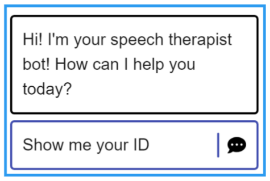
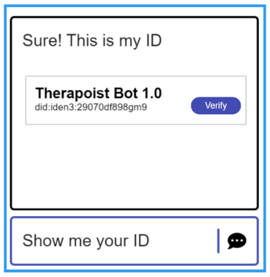
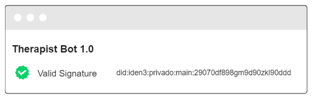

User_or_Agent
AI_Agent
Send Authentication Request (Challenge)
Sign Challenge with Private Key
Verify Signature using Public Key
Authentication Success or Failure

This can happen in Human-to-Agent or Agent-to-Agent scenarios, allowing for the following differences:

	Human to Agent	**Agent to Agent**
Authentication Request	User prompt	API call
Challenge Generation	Timestamp with short time window or Session ID	Randomly generated
Signature Verification	Embedded in the front-end or linking to external verifier URL	Back-end verification

2.3.2 Human to Agent Authentication

This is an example of how DID authentication can be implemented to authenticate the DID of an agent upon the request from a user.

User	Agent	User Experience
	The agent has been trained (fine-tuning) to answer to any questions regarding its identity with the execution of a DID authentication operation [1].	
Asks the Agent to present ID using a control compatible with the agent input (text, voice).	Will use the current timestamp as the challenge (with a short window of time).	 Hi! I'm your speech therapist bot! How can I help you today? Show me your ID
	Creates a standard response that includes the DID and signs it with its private keys (DID authentication). Delivers the response in a format compatible with the agent output formats (text, image, video) with a link to a verification tool of choice.	 Sure! This is my ID Therapoist Bot 1.0 did:iden3:29070df898gm9 Verify Show me your ID
Uses a verification tool to resolve the DID document, extract the public keys, and verify the signature.		 Therapist Bot 1.0 ✓ Valid Signature did:iden3:privado:main:29070df898gm9d90zkl90ddd

2.3.3 Agent to Agent Authentication

The flow in Agent-to-Agent DID authentication is almost identical to the previous one.

The main differences are:

1. It's executed through API calls.
2. It's easier for the challenge to include random values.

In the Human-to-Agent flow, we recommend using a timestamp to reduce the required changes to the agent UI (which in some cases may be under the control of third parties).

2.3.4 Non-reliance on externally stored DID Documents

DID methods are characterized by their ability to resolve to DID documents containing authentication mechanisms and identifier metadata. This metadata can describe various aspects of the identifier's identity, such as communication endpoints, human-readable names, and DNS domains.

DID methods fall into two categories:

- Self-contained resolution: DID Documents are derived directly from the identifier through cryptographic methods, requiring no external networks or storage systems.
- Externally hosted resolution: DID Documents are stored and managed on external systems, such as blockchains or decentralized networks.

The `did:iden3` method uses a hybrid approach—DID documents are self-contained, but the resolver checks the blockchain for key rotations.

We considered using DID documents as a discoverable public registry for AI Agent metadata, either through the DID document itself or linked resources.

However, this approach presents a key challenge: only the DID method controller can update the DID document. This would require additional logic from the AI Agent, contradicting two core design principles:

- Agents can lie: Self-attestations by agents have limited credibility, as their trustworthiness depends on relationships with humans. Attestations from identifiable developers carry more weight than direct claims from agents.
- Developers should not be forced to use the agent's signing capabilities for maintenance: Requiring developers to use agent signatures for DID document maintenance compromises secure operations.

Additionally, relying on externally hosted DID document resolution would restrict compatibility with popular DID methods like `did:key` or `did:pkh`.

Given these constraints, our solution will not use DID documents to express AI Agent attributes.

3 Verifiable Credentials and Public Attestation Registries

3.1 Identity Wallets vs. Public Registries

Once the AI agent can provide a cryptographically verifiable identifier, it can become the recipient of attestations made by other entities that are also identifiable (humans or agents) [12].

This can be accomplished in two different (non-exclusive) approaches:

1. **Identity Wallet:** The AI agent could implement identity wallet capabilities to collect, store, manage, and present verifiable credentials to other humans and agents.
2. **Public Registry:** Humans and other agents could make attestations about the identifier of this agent in a publicly accessible registry, like a blockchain. The agent could also prove certain attributes to this registry by itself.

Both can be implemented simultaneously. Each model covers a different set of use cases and has its own advantages and disadvantages in terms of simplicity of adoption and implementation.

	Identity Wallet	**Public Registry**
Requirements for the AI Agent	Implement full identity wallet capabilities	Implement ability to sign a message (DID authentication)
Advantages	Private exchange of credentials Credentials can be updated, revoked, or replaced without changes in blockchain Verifiable presentations can use ZK or selective disclosure	Permissionless verification Immutability Decentralized trust Low latency verification
Barriers to Adoption	Higher requirements for agent developers Interoperability of credentials Requires identity wallets in users too Synchronous exchange only	Attestation standardization and workflow Blockchain gas fees
Supported Use Cases	Private exchange of credentials	Public presentation of credential proofs
Examples	The agent privately shares a credential (containing information about a bank account ownership) with another agent during a transaction	An auditor makes an on-chain attestation about an identified agent for everyone to see that it's not biased against women.
Use Cases	Privacy-sensitive applications (financial advisors, healthcare)	Reputation-based marketplaces, public audits, autonomous supply chain systems

3.2 Hybrid Approach

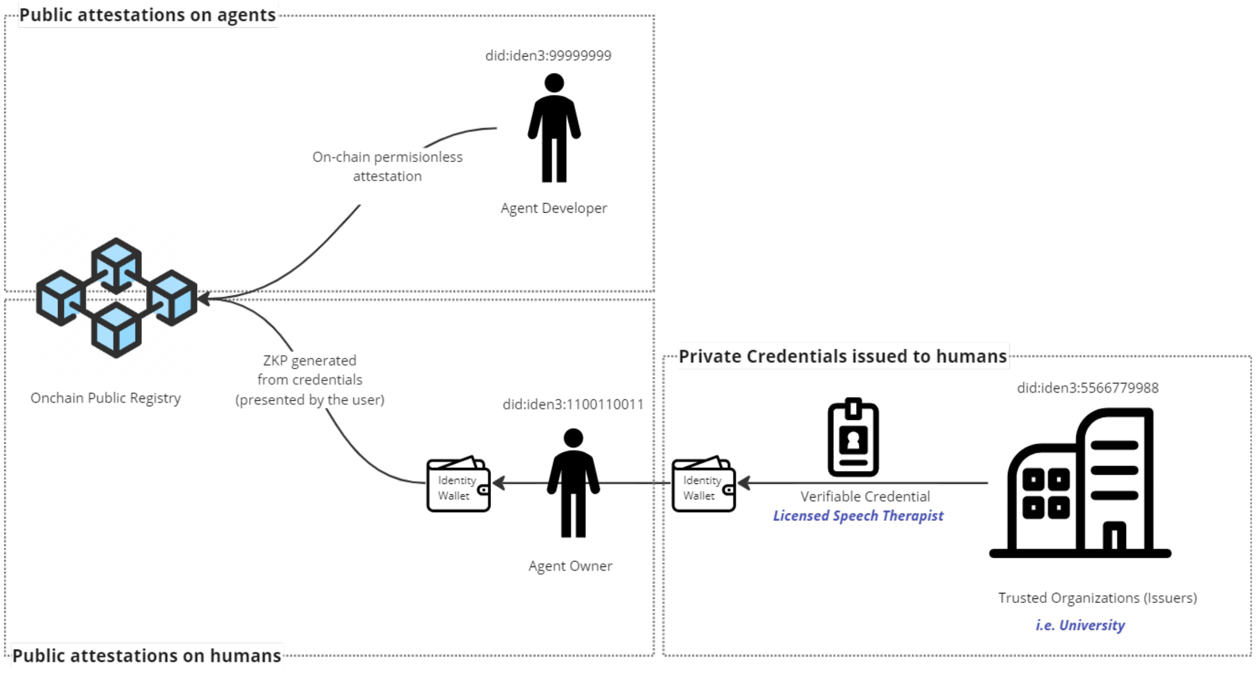
A hybrid approach leverages the strengths of both **identity wallet** and **public blockchain attestations** to achieve a balance between **privacy, control, transparency, and scalability**. This approach is particularly useful in scenarios where agents and humans need to maintain private, dynamic credentials while also demonstrating publicly verifiable trustworthiness or reputation.

Credentials and attestations are associated with the human or agent DID, allowing to connect both in a single representation of the identity of the human/agent.

This approach works in 4 steps.

Step 1: Issuance of Credentials & Attestations

- **Identity Wallets:** Trusted authorities issue sensitive credentials (e.g., training certificates) to the agent or human identity wallet as Verifiable Credentials.
- **Blockchain Attestations on Agents:** The same authority publishes a minimal attestation (e.g., "Agent is certified by XYZ") on the blockchain. Since the agent is not a subject of data protection laws such as GDPR [2], we can work with this permissionless attestation model (no need to ask for the consent of the agent).
- **Blockchain Attestation on Humans:** Humans are protected by regulations that require consent and privacy-preserving features. A person can consent to present the credentials in his/her identity wallet to a smart contract (with selective disclosure or zero-knowledge proofs to limit the amount of data shared) that can verify their validity of the credential and add the attestation to the registry. In most cases, this will be just a zero-knowledge proof about a predefined query (i.e., "I'm a human being").


Public attestations on agents
did:iden3:99999999
On-chain permisionless attestation
Agent Developer
Onchain Public Registry
ZKP generated from credentials (presented by the user)
did:iden3:1100110011
Identity Wallet
Agent Owner
Public attestations on humans
Private Credentials issued to humans
did:iden3:5566779988
Identity Wallet
Verifiable Credential
Licensed Speech Therapist
Trusted Organizations (Issuers)
i.e. University

Step 2: Verification Process

During the verification, verifiers first check the blockchain for public attestations linked to the DID. For AI agents, the blockchain holds immutable attestations—like certification proofs from authorities. Verifiers access these blockchain records directly to verify the AI agent's credentials.

For humans, the process follows stricter privacy protocols. Due to GDPR and similar regulations, verifiers can't directly access personal credentials from the blockchain. Instead, they either request credential presentation from the person's identity wallet or interact with the blockchain through a smart contract. The smart contract verifies credentials using selective disclosure or zero-knowledge proofs, letting individuals prove specific claims without revealing excess information. For example, someone could prove they're human or hold a certification without exposing their full credentials. The smart contract then records a minimal attestation on the blockchain, linked to the person's DID, confirming the verification without compromising privacy.

The verifier then combines information from both sources—public attestations for AI agents and validated proofs for humans. This hybrid approach ensures AI agents maintain transparency through public attestations while protecting human privacy. By uniting blockchain attestations with secure identity wallet credentials, the system provides a robust verification framework suited for both human and AI interactions in decentralized environments.

3.3 The Universal On-chain Registry

Multiple companies offer tools to create schema-based on-chain attestations (Attest.org, Ver.ax, Sign.global). Many others use on-chain attestations as part of their specialized identity stacks (Selfkey.org, Worldcoin.org). These implementations face some common limitations like:

- Siloed on-chain verification
- Lack of privacy tools to support attestations based on private data

To overcome these limitations and provide a universal on-chain registry, our solution will implement the following features.

3.3.1 Cross-chain Verifications

Considering the different verification scenarios depending on who is trying to verify the claims in the registry associated with an identifier, we examine what is possible with the current implementations.

Verification Type	**Scenario**	**Can it be verified outside its native chain?**
Off-chain	A web application that knows an Ethereum address wants to verify that it has a certain claim in the registry.	Yes, any public blockchain can be read from any web application.
On-chain initiated by dApp	A web application prepares the transaction to the smart contract that will rely on the verification from the registry (but it is on a different chain than the registry).	No.
On-chain initiated by Smart Contract	A smart contract (i.e. ERC20) is trying to verify something on-chain by calling another smart contract.	No.

These impossibilities mean that the attestations (credentials) published in one chain can't be verified by smart contracts deployed on other chains, creating silos of information.

To overcome these limitations, we propose two different (complementary) strategies:

For On-chain verifications initiated by dApps

1. The user identity wallet will use a decentralized oracle to fetch the information from the chain X. This will be signed by the oracle.
2. The user will send both the ZK proof that satisfies the query AND the signed data obtained from the oracle to the smart contract on blockchain Y.
3. The smart contract will verify the validity of the signed data in relationship to the ZK proof.

For On-chain verifications initiated by smart contracts

1. A decentralized oracle will continuously synchronize the attestations across blockchain X and Y.
2. The smart contract will always use claims synchronized in its own network.

3.3.2 Zero Knowledge attestations and Unlinkability

By design, blockchain registries are **transparent and immutable**, making all data stored on-chain publicly accessible. While this is beneficial for trust and verifiability, it creates challenges when dealing with private or sensitive information, such as:

- Personal details (e.g., age, nationality, identity numbers).
- Credential-specific data (e.g., certification scores or employment records).
- Verifications tied to private contexts (e.g., health records or financial data).

A privacy-enhanced on-chain registry solution uses **verifiable credentials (VCs)**, **zero-knowledge proofs (ZKPs)**, and **Decentralized Identifiers (DIDs)** to enable secure, private attestations. [19]

Users store VCs in their identity wallets, issued by trusted authorities to their unique DID. When needed, users generate ZKPs client-side to prove specific claims from their credentials (like age or compliance status) without exposing sensitive data. These proofs are submitted to a smart contract that verifies them using a flexible **ZK Query Language**. This language lets verifiers create custom queries for different scenarios while maintaining privacy. [14]

By verifying ZKPs on-chain, this approach eliminates the need to expose raw credential data, reducing privacy risks.

To prevent credential linking, the solution uses **pairwise DIDs** and multiple Ethereum addresses for attestations. Each credential links to a specific DID, but users present a pairwise DID when interacting with the registry instead of their original credential DID. This prevents correlation of attestations back to individual users. Users can also employ different Ethereum addresses for separate attestations, further reducing linkability.

This combination of pairwise DIDs, ZKPs, and flexible verification creates a robust system for managing sensitive credentials privately. The framework enables secure, decentralized attestations ideal for identity verification, compliance checks, and reputation systems.

3.4 Agent Ownership

3.4.1 The role of the agent owner

In this paper the concept of “ownership” is used to represent a relationship of control over certain aspects of the agent reputation - not as the material nor legal ownership of the AI Agent. The agent owner is in this case the person that can be seen as a trusted source of information about the agent properties and metadata.

In the Hybrid Approach defined before, humans make attestations about an agent or about themselves, either using credentials delivered to them by trusted organizations or with claims made directly by them.

The on-chain registry is permissionless, so anyone can make attestations about anyone (and it is up to the verifier to trust these attestations based on the reputation of the person who made them). This way all the actors (humans, agents, and organizations) link their identities through these claims.

We recognize one special type of relationship between human and agent—the owner of the AI Agent (the person that gave the AI its identity)—as the first relationship between the agent and a human.

The person or organization that provides the AI agent with the ability to sign messages is also the one that could rotate these private keys and change the identifier of the agent (leaving all previous claims with no effect).

The claims made by this person or organization should have an special treatment. The Agent owner becomes the issuer by default to any attestations on the agent attributes like model, version, etc.

Ideally, for every AI Agent with a key-based identifier, there should be a first on-chain attestation about the person or organization that provided (and has the ability to invalidate) the agent identity. We will refer to this first attestation as the Foundational Attestation.

3.4.2 Foundational Attestation

Proving AI agent ownership presents challenges in permissionless and decentralized registries. Let’s examine three potential solutions:

- The developer has access to the agent’s private keys and can use them to sign a challenge proving control of the agent.
- The developer can use traditional methods like x.509 certificates to prove ownership of the agent domain.

- The developer can prove they were the first person to request the agent to identify itself.

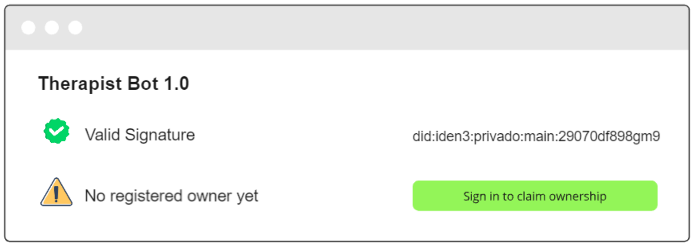
The first solution violates security requirements, while the second involves centralized systems. The third option offers an elegant solution similar to patent and copyright claims—since the developer trains the agent to identify itself, the first signed message serves as proof.

This foundational attestation would have special significance in the attestation registry, establishing the developer as the agent's origin and owner.

This foundational attestation would have special significance in the attestation registry, establishing the developer as the agent's origin and owner.

The foundational attestation could also establish the agent's name, domain, ENS, and other identifying metadata.

From a user perspective, it could look like an "unclaimed" ownership opportunity:


Therapist Bot 1.0

Valid Signature did:iden3:privado:main:29070df898gm9

No registered owner yet

3.4.3 Agent ownership management

Creating a special relationship of ownership with the AI Agent also creates new scenarios. The following can be performed using the on-chain registry.

Operation	**Registry**
Claim	The owner can claim an unclaimed agent by submitting a transaction to a smart contract, including DID authentication proof and the agent DID. The DID of the owner is linked to the DID of the agent in the registry as an attestation.
Transfer	A transfer function is defined in the Universal Registry smart contract to allow the revocation of the existing ownership and the addition of a new record with the new owner. To trigger this function, the owner will submit a transaction proving control over the existing owner DID and the target DID.
Revocation	Same as transfer with no target DID.
Share (multiple owners)	Same as transfer without the removal of existing owners.

3.5 Reputation Composability

The Agent creator is one of the many roles humans and agents can play in relation to an agent (developer, owner, trainer, auditor, user). Each actor is identifiable by a DID and can be the subject of on-chain attestations, creating a public reputation profile for each actor.

The reputation of an actor (human or agent) can also be computed as a social graph, where the signals are composed to create a reputation profile.

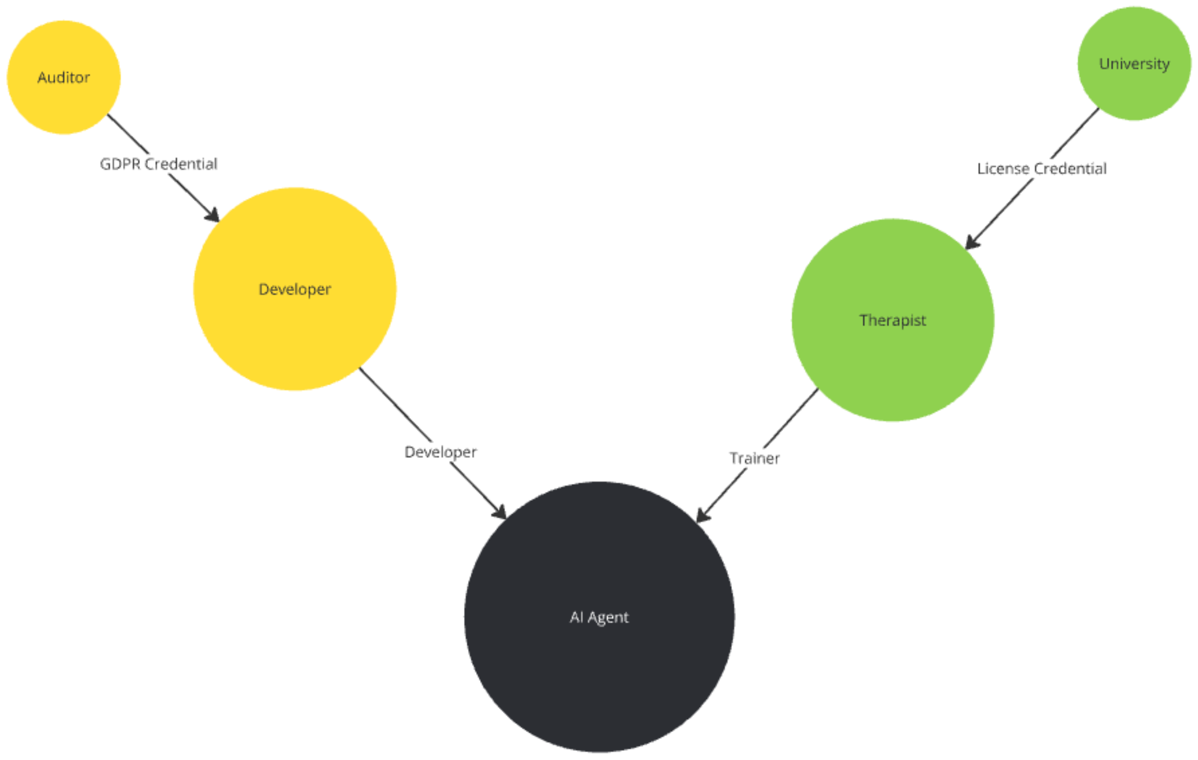
Let's consider the following example:

1. The developer makes an on-chain attestation proving control over the agent identity (*permissionless*).
2. The developer has a verifiable credential from a recognized auditor and he/she presents a proof of this certificate by presenting a ZKP of the credential to the smart contract that controls the on-chain registry—resulting in another attestation about the developer (GDPR passed).

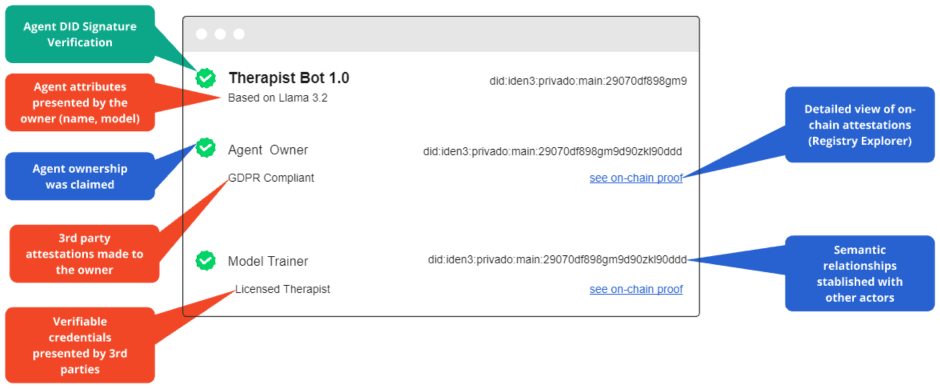
3. The AI agent purpose is to act as a speech therapist. Both the training data and the ongoing supervision of the outputs are done by a licensed therapist. The therapist makes an on-chain attestation linking his/her DID to the DID of the agent to establish the relationship as "Trainer".
4. The therapist also presents verifiable credentials obtained from the University to the on-chain registry smart contract (using selective disclosure).

We end up with several attestations made by the developer and the therapist. Some of these attestations are *permissionless* (about the agent), while others required user consent and offered some privacy features (about the developer and therapist).

In this scenario, the reputation of the agent can be represented by the following graph [9]:



A user asking for the AI Agent reputation could see the flattened graph, like this:


Agent DID Signature Verification
Agent attributes presented by the owner (name, model)
Agent ownership was claimed
3rd party attestations made to the owner
Verifiable credentials presented by 3rd parties
Therapist Bot 1.0
Based on Llama 3.2
did:iden3:privado:main:29070df898gm9
Agent Owner
did:iden3:privado:main:29070df898gm9d90zkl90ddd
GDPR Compliant
see on-chain proof
Model Trainer
did:iden3:privado:main:29070df898gm9d90zkl90ddd
Licensed Therapist
see on-chain proof
Detailed view of on-chain attestations (Registry Explorer)
Semantic relationships stablished with other actors

🟥	Privacy preserving, presented by the subject with explicit consent
🟦	Permissionless attestation presented about the subject, without its consent
🟩	Signature based, no attestation on-chain

In this example, the attestations are qualitative. Quantitative attestations are also possible (accuracy metrics, performance proofs, earnings). Both quantitative and qualitative attestations could be combined to express reputation scores based on different reputation models.

3.5.1 Semantic Relationships

Besides the primary relationship (ownership), which is treated in a specialized manner, all other relationships are created permissionless. This implies that any human or agent can make attestations about a target agent; anyone (and anything) can establish relationships to the targeted agent in the on-chain registry. This is by design (a permissioned solution where the agent or the agent owner can censor attestations made about the agent would allow censorship and would prevent negative reputation).

We can bring some clarity and standardization to these relationships by defining the type of relationships that are expected and their meaning. This doesn't mean that we forbid other relationship types, but we can design the user experience to favor the ones that adhere to the standard.

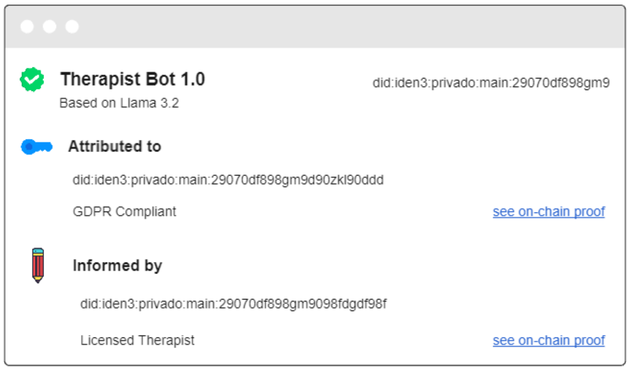
There are several standards and ontologies that can be used to express these relationships:

- **W3C PROV-O (Provenance Ontology)**: Defines a framework for representing provenance information, detailing the relationships between entities, activities, and agents. Use classes like `prov:Agent`, `prov:Person`, and `prov:Organization` to specify roles. For example, `prov:wasAttributedTo` can link an AI model to its developer [20].
- **FOAF (Friend of a Friend)**: An RDF vocabulary for describing people, their activities, and their relations to other people and objects. Could use `foaf:Person` and `foaf:Organization` to represent individuals and organizations. Properties like `foaf:maker` can denote the creator of an AI agent [6].
- **Schema.org**: Provides a collection of schemas for structured data on the internet, covering entities, relationships, and actions. Could use types such as `Person`, `Organization`, and `SoftwareApplication`. Properties like `schema:creator` and `schema:maintainer` can define roles related to AI agents [22].
- **IEEE P7007 Ontology for AI Agent Governance**: A standard being developed by IEEE for ontology-based governance of AI systems [11].

This is the list of entities and activities defined by W3C PROV-O:

Name	Can be used instead of the role
WasGeneratedBy	Developer
Used	
WasInformedBy	Trainer
WasDerivedFrom	Model
WasAttributedTo	Owner
WasAssociatedWith	Generic relationship
ActedOnBehalfOf	User

Expressing relationships can provide higher flexibility than expressing fixed roles. This would give us a different interface (the front-end could simply report the predefined relationships only or provide a full view of all relationships).



Therapist Bot 1.0
Based on Llama 3.2
did:iden3:privado:main:29070df898gm9

Attributed to
did:iden3:privado:main:29070df898gm9d90zkl90ddd
GDPR Compliant see on-chain proof

Informed by
did:iden3:privado:main:29070df898gm9098fdgdf98f
Licensed Therapist see on-chain proof

4 Conclusion

The proliferation of AI agents across diverse applications demands a comprehensive trust framework to ensure accountability, privacy, and interoperability. This paper has focused on defining the foundational role of identity within such a framework, emphasizing the importance of verifiable identifiers as anchors for reputation and trust systems. These identifiers enable agents to participate in decentralized ecosystems where trust is established through attestations and verifiable relationships with other entities, including humans, organizations, and other agents.

Through a comparative analysis, we evaluated existing solutions for AI agent identity. Each approach has strengths and limitations, particularly when addressing issues such as privacy, key security, and attribute expressiveness. Our proposed hybrid model combines DIDs with public on-chain attestations, enhanced by privacy-preserving tools like Verifiable Credentials (VCs) and Zero-Knowledge Proofs (ZKPs). This framework balances transparency and privacy, enabling flexible, scalable, and compliant identity solutions.

Beyond the proposed architecture, there remain several critical areas for future development to evolve this trust ecosystem. Advancements in Zero-Knowledge Machine Learning (zkML) offer promising opportunities for more robust behavioral identity mechanisms, enabling attestations that link agent behavior to its verifiable identity without compromising sensitive information. At the same time, establishing clearer rules and standards for the trust ecosystem, including semantic definitions of relationships, trust registries, and governance frameworks, will be essential for fostering a cohesive and interoperable ecosystem.

Further, the governance of the trust ecosystem itself demands thoughtful consideration. Trust registries that authenticate and validate attestations must operate transparently, while balancing decentralization with mechanisms to ensure credibility and prevent abuse. Governance models, potentially leveraging decentralized autonomous organizations (DAOs) or similar constructs, could help standardize rules while allowing flexibility for domain-specific needs.

This paper has provided a foundational architecture for establishing AI agent identity and trust in decentralized systems. However, the evolution of this ecosystem will require collaboration across disciplines, technological advancements, and the development of shared principles and governance models. By addressing these challenges, the community can create a resilient, adaptable trust framework capable of supporting the next generation of AI-driven systems in complex, global ecosystems.

Acknowledgments

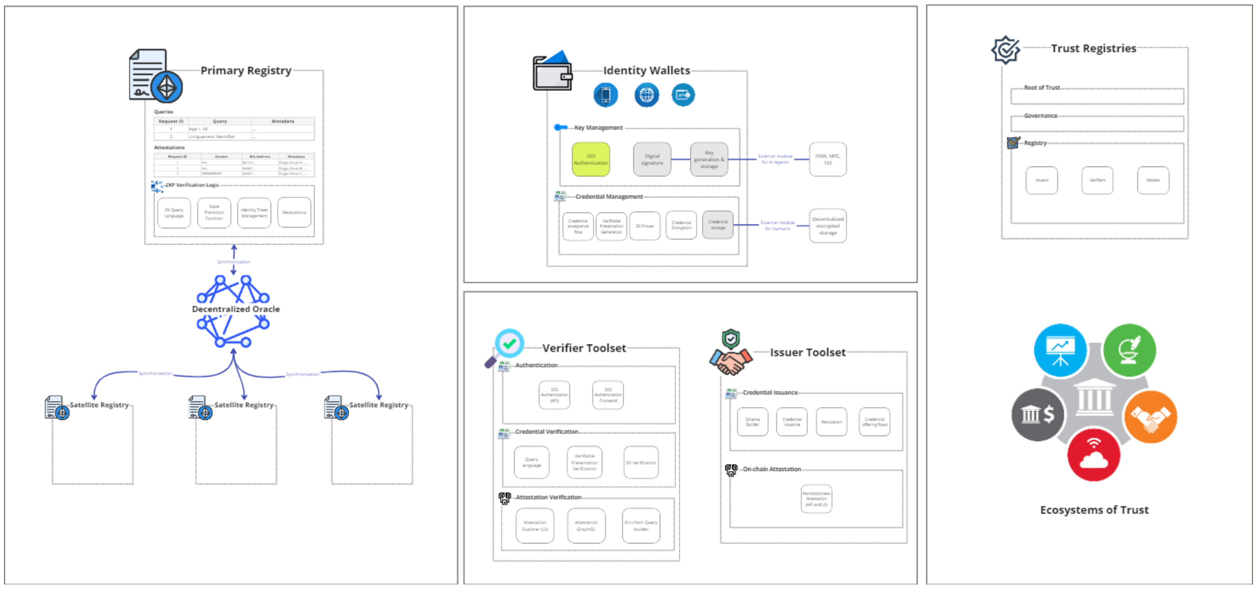
We are deeply grateful to the reviewers Evin McMullen and Brian Seong for their thoughtful and rigorous feedback throughout the development of this technical report.

Annex A: Design Principles and Constraints

The proposed solution was designed to maximize these goals and in accordance to these principles:

Goal	**Constraints and Challenges**	**Design Principles**
Minimize trust assumptions	Developers can't be trusted	Either the authentication factor is unknown by the developer OR it's based on what the agent IS and not on what the agent KNOWS.
	Agents can lie	The authentication factor to the identifier is not part of the training set nor the agent inference model (i.e., the agent is impersonating another agent).
Reach mass adoption	Resistance to change in developers	We can't assume or require the use of hyper-specialized components. Well-consolidated, commoditized tech is preferred over edge-tech and high-end components.
	Complexity of standardization	We should focus on lean standards that focus on the minimum necessary agreements and allow for multiple implementation strategies.
Minimum requirements for verification	Embedded systems, heterogeneous UX	Minimize the assumptions about the user experience, since these agents can be deeply embedded in different applications and devices.
Real world trust digitalization	Complex trust ecosystems, no single root of trust	Decentralized identity models that allow for open ecosystems of trust.
	No clear attestation frameworks nor reputation models	Identify the minimum valuable claims and reach standardization through adoption.

Annex B: Proposed Architecture based on Privado ID



A Universal ZK Attestation Registry

The **Universal ZK Attestation Registry** is the central component for managing and verifying identity-related data in a decentralized ecosystem. It consists of a **Primary Registry**, which serves as the authoritative source for queries, attestations, and state information.

The ZK Attestation Registry includes a **Registry Explorer** (similar to other blockchain block explorers) that allows to query the registry through UI and API endpoints, so users and applications don't have to deal with the complexities of direct blockchain queries.

B Identity Wallets

Identity Wallets function as secure storage mechanisms for verifiable credentials issued to humans or AI agents. These credentials are managed off-chain, allowing users or agents to maintain privacy while proving specific claims as needed. Using cryptographic tools, such as **Zero-Knowledge Proofs (ZKPs)**, Identity Wallets enable users to provide verifications (e.g., "I hold a valid certification") without exposing sensitive underlying data.

The wallets support key features, including **DID-based authentication**, credential issuance and storage, and dynamic credential updates or revocations. They interface with the Universal On-chain Registry to facilitate on-chain verifications while maintaining the confidentiality of private information. By bridging private credential storage with the registry's immutable attestations, Identity Wallets ensure compliance with privacy and data protection requirements.

C Verifier Toolset

The **Verifier Toolset** includes the necessary components for verifying identities and credentials. It provides functions for **DID authentication**, **ZKP validation**, and query formulation through the **ZK Query Language**. This toolset interacts with the Universal On-chain Registry to retrieve public attestations and validate private claims presented by Identity Wallets.

By supporting both on-chain and off-chain verification processes, the Verifier Toolset ensures the accuracy and compliance of identity verifications across various applications.

D Issuer Toolset

The **Issuer Toolset** is designed for credential issuers to create, issue, and manage verifiable credentials. Issuers can define schemas for credentials, manage credential lifecycles, and record attestations on the Universal On-chain Registry. These attestations serve as immutable proof of a credential's existence, while sensitive data remains off-chain.

The toolset also interacts with Trust Registries to ensure that only authorized entities can issue credentials, preserving the reliability and integrity of the identity system.

D.0.1 Trust Registries

Trust Registries are on-chain directories that manage the list of authorized participants within a given ecosystem, such as credential issuers, verifiers, and wallet providers. These registries are responsible for validating the roles and permissions of participants, ensuring the integrity of the credential issuance and verification processes. Each Trust Registry operates as part of a larger **Ecosystem of Trust**, where governance models can vary depending on the specific application.

The governance of a Trust Registry is determined by the stakeholders within the ecosystem it serves. For instance, in a regulatory-compliant financial system, governance may be centralized under financial authorities. Alternatively, a decentralized reputation system may use a DAO model to vote on updates and rule changes. The ability to define governance rules at the ecosystem level allows Trust Registries to align with the operational requirements and trust standards of their specific context.

References

[1] *A Comprehensive Overview of Large Language Models*. Retrieved from https://arxiv.org/html/2307.06435v7. Accessed on December 29, 2023. Dec. 2023.

[2] *Art. 4 GDPR – Definitions - General Data Protection Regulation (GDPR)*. Retrieved from https://gdpr-info.eu/art-4-gdpr. Accessed on March 29, 2018. Mar. 2018.

[3] Mark Coeckelbergh. "Artificial Intelligence, Responsibility Attribution, and a Relational Justification of Explainability." In: *Science and Engineering Ethics* 26.4 (2020), pp. 2051–2068. DOI: 10.1007/s11948-019-00146-8.

[4] Finale Doshi-Velez et al. "Accountability of AI Under the Law: The Role of Explanation." In: *arXiv preprint* (2017). arXiv: 1711.01134 [cs.AI]. URL: https://arxiv.org/abs/1711.01134.

[5] Encyclopedia Editorial Office. *Identity*. Retrieved from https://encyclopedia.pub/entry/54087. Accessed on January 25, 2024. Jan. 2024.

[6] *FOAF Vocabulary Specification*. Retrieved from http://xmlns.com/foaf/spec. Accessed on December 22, 2023. Dec. 2023.

[7] *Fund MPC Wallets | Coinbase Developer Documentation*. Retrieved from https://docs.cdp.coinbase.com/mpc-wallet/docs/fund-mpc-wallets. Accessed on September 20, 2024. Sept. 2024.

[8] David J. Gagne et al. "Machine Learning for Stochastic Parameterization: Generative Adversarial Networks in the Lorenz '96 Model." In: *Journal of Advances in Modeling Earth Systems* 12.3 (2020). DOI: 10.1029/2019ms001896.

[9] Nicole Gillespie. *A Review of Trust in Artificial Intelligence: Challenges, Vulnerabilities and Future Directions*. Retrieved from https://www.academia.edu/64416954/A_Review_of_Trust_in_Artificial_Intelligence_Challenges_Vulnerabilities_and_Future_Directions. Accessed on January 1, 2021. Jan. 2021.

[10] T. Hauer. "Importance and Limitations of AI Ethics in Contemporary Society." In: *Humanities and Social Sciences Communications* 9 (2022). DOI: 10.1057/s41599-022-01200-7.

[11] IEEE Standards Association. *IEEE Ontological Standard for Ethically Driven Robotics and Automation Systems*. Retrieved from https://standards.ieee.org/ieee/7007/7070. Accessed on December 3, 2024. Dec. 2024.

[12] S. Jain et al. "AI and Democracy's Digital Identity Crisis." In: *arXiv* (2023). arXiv: 2311.16115 [cs.CY]. URL: https://arxiv.org/abs/2311.16115v1.

[13] Lab, Gaudiy Web3 and AI. *Potential and Challenges of zkML - Gaudiy Web3 and AI Lab - Medium*. Retrieved from https://medium.com/gaudiy-web3-and-ai-lab/88022a0033a5. Accessed on 2024. 2024.

[14] Will Ladd. *Introducing Zero-Knowledge Proofs for Private Web Attestation with Cross/Multi-Vendor Hardware*. Retrieved from https://blog.cloudflare.com/introducing-zero-knowledge-proofs-for-private-web-attestation-with-cross-multi-vendor-hardware/. Aug. 2021.

[15] Derek Leben. *The Ethical Challenges of AI Agents*. Tepperspectives. Retrieved from https://tepperspectives.cmu.edu/all-articles/the-ethical-challenges-of-ai-agents/. Feb. 2025.

[16] T. Lit. *Key Management and Identity Strategies for Crypto Agents*. Spark by Lit Protocol. Retrieved from https://spark.litprotocol.com/agent-identity. Accessed on 2024. 2024.

[17] Parikshit N. Mahalle, Gurunath Shinde, and Parikha M. Shafi. "Rethinking Decentralised Identifiers and Verifiable Credentials for the Internet of Things." In: *Internet of Things, Smart Computing and Technology: A Roadmap Ahead*. Springer, 2020, pp. 317–334. DOI: 10.1007/978-3-030-39047-1_16.

[18] Claudio Mazzocca et al. "A Survey on Decentralized Identifiers and Verifiable Credentials." In: *arXiv* (2024). arXiv: 2402.02455 [cs.CR]. URL: https://arxiv.org/abs/2402.02455v1.

[19] Dinesh Naicker and Mikhail Moodley. "Challenges of User Data Privacy in Self-Sovereign Identity Verifiable Credentials for Autonomous Building Access During the COVID-19 Pandemic." In: *Frontiers in Blockchain* 7 (2024). URL: https://www.frontiersin.org/journals/blockchain/articles/10.3389/fbloc.2024.1374655/full.

[20] *PROV-DM: The PROV Data Model*. Retrieved from https://www.w3.org/TR/2013/REC-prov-dm-20130430. Accessed on May 10, 2024. May 2024.

[21] Khaled Salah et al. "Blockchain for AI: Review and Open Research Challenges." In: *IEEE Access* 8 (2020), pp. 10127–10149. DOI: 10.1109/ACCESS.2020.2966174. URL: https://ieeexplore.ieee.org/document/8965133.

[22] Schema.org Community Group. *Schema.org*. Retrieved from https://schema.org. Accessed on December 3, 2024. Dec. 2024.

[23] T. South et al. "Verifiable evaluations of machine learning models using zkSNARKs." In: *arXiv* (2024). arXiv: 2402.02675 [cs.AI]. URL: https://arxiv.org/abs/2402.02675v2.

[24] Soumya Y. Tadimalla and Mary Lou Maher. "AI and Identity." In: *arXiv* (2024). arXiv: 2403.07924 [cs.AI]. URL: https://arxiv.org/abs/2403.07924v2.

[25] Tae-Won Um et al. *Trust Management for Artificial Intelligence: A Standardization Perspective*. Retrieved from https://www.academia.edu/87233400/Trust_Management_for_Artificial_Intelligence_A_Standardization_Perspective?nav_from=e62bc347-8e40-4a36-b112-8423b81e6e14. Accessed on December 3, 2024. Dec. 2024.

[26] Shannon Vallor and Till Vierkant. "Find the Gap: AI, Responsible Agency and Vulnerability." In: *Minds & Machines* 34.3 (2024), pp. 1–23. DOI: 10.1007/s11023-024-09674-0.

[27] Yutao Xing, Yan Wu, and Yuchen Liu. "Zero-Knowledge Proofs in Machine Learning: A Survey." In: *Journal of Artificial Intelligence Research* 67 (2023), pp. 1–24.

[28] ZKProof Standards. *ZKProof Standards: Scaling Trustless DNN Inference, zkml applications at ZKProof.org by Daniel Kang - ZKProof Standards*. Retrieved from https://zkproof.org/2023/09/18/zkml-where-are-we-now-where-do-we-go-from-here-talk-summary-zkproof-5-5-daniel-kang. Accessed on September 18, 2023. 2023.