How is Ethereum fixing the missing native identity layer of the internet?
The on-chain identity paradigm

Microchain Labs

White Paper

Authored by

Walid Khemiri
Mohamed Ismail Amara

Abstract

This white paper presents the advancements by **Microchain Labs** in addressing the critical challenges of digital identity within blockchain ecosystems. Centralized digital identity systems have long undermined individual privacy, ownership, and autonomy, but Microchain Labs introduces innovative solutions rooted in decentralized principles.

At the core of the lab's approach are **Limited Scope Accounts (LSAs)** and **ZK Session Keys**, which redefine self-custody and account management by leveraging trust-minimized off-chain computation and zero-knowledge proofs (ZKPs). LSAs allow users to customize account capabilities based on specific needs, introducing human-centric, intent-driven behaviors that align with real-world dynamics. ZK Session Keys enhance security and usability by validating user operations off-chain before securely executing them on-chain, reducing costs while maintaining cryptographic security.

Microchain Labs also integrates frameworks like the **ZK Policy Engine**, enabling programmable and scalable account features through off-chain logic validated by cryptographic proofs. This approach bridges the gap between traditional cryptographic wallets and advanced smart contract wallets, offering secure, scalable, and user-friendly self-custody experiences. By focusing on creating Web2-like user experiences within a Web3 infrastructure, Microchain Labs brings decentralized identity closer to mainstream adoption.

Through these innovations, Microchain Labs envisions a future where identity is a public good, managed transparently and securely, free from centralized control. The lab's work is a significant step toward a decentralized internet where users fully own and control their identities, making blockchain technology accessible, human-centric, and scalable for real-world applications.

1. Introduction

From the very beginning, human identification and identity services on the internet have to deal with challenges. At its core, identity is a **fundamental human right**, and yet the infrastructure that supports it has long been fragmented, privatized, and centralized. **Identity services on the internet must evolve to be treated as a public good**, owned and controlled by individuals rather than corporations.

The Current State of Identity Services

The first generation of digital identity systems emerged based on how we build computer programs and services, as well as the trust assumptions inherent to the internet. These systems were predominantly **centralized** and operated by **trusted third-party companies**, leading to two main models of identity management:

1. **Siloed Identity**: Each platform or service creates and manages its own isolated identity solution.
2. **Federated Identity**: Multiple platforms rely on a single identity provider to authenticate users across systems.

To enable interoperability, standards like **SAML**, **OAuth2**, and **OIDC** were introduced to facilitate the exchange of identity information (authentication and authorization) between systems. While these protocols introduced some level of convenience, they did not address the root issue: **control and ownership of digital identity**.

A System at Odds with Human Rights

In the current model, human identity on the internet is owned and controlled by a handful of powerful corporations. These centralized entities wield immense control, with the ability to manage or even erase an individual's digital presence with a single click. This system stands in stark contrast to the principle that **identity is a fundamental human right**—one that should be **owned and controlled by individuals**.

This privatized approach raises critical concerns about privacy, autonomy, and trust. When corporations act in their own interests, they can manipulate, revoke, or exploit digital identities without accountability, leaving individuals vulnerable.

A New Paradigm: Decentralized Identity

The advent of **Bitcoin in 2008** introduced a transformative approach to identity and trust on the internet. For the first time, human identification and authentication were achieved in a **decentralized and trust-minimized manner**. Bitcoin's model allowed individuals to authenticate and perform digital operations without relying on a central authority.

This revolutionary approach laid the foundation for rethinking identity systems, offering a glimpse into a future where digital identity is **self-sovereign, secure, and decentralized**—empowering individuals to take back control of their online presence.

2. Human identification by Bitcoin protocol

The Bitcoin blockchain represents a groundbreaking form of **decentralization** and **trust minimization**, redefining how accounts are identified and authenticated without relying on centralized authorities.

Decentralized State Management

At its core, Bitcoin's ledger maintains the state of user accounts and processes state transitions in a **censorship-resistant, decentralized, and trust-minimized manner**. These state transitions are computed based on user-intended operations, ensuring that no single entity can alter or obstruct the process. This decentralization extends across multiple critical components, including:

- **Consensus Mechanisms**
- **Block Building and Verification**
- **Data Availability**
- **Mitigation of MEV (Miner Extractable Value)**
- **Physical Decentralization of the network**
- **Protocol Rules**

By decentralizing these layers, Bitcoin achieves a robust, trust-minimized infrastructure that empowers individuals and resists manipulation or censorship. It is a trust-minimized state machine on the Internet secured via crypto economic security.

Decentralized Identification and Authentication

Bitcoin introduced a revolutionary approach to **identifying accounts** and **authenticating transactions**. This model provides the foundation for a new era of digital identity, allowing users to operate without the need for centralized third parties. Here's how it works:

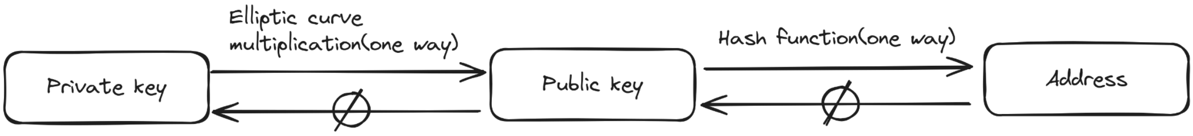
- **Identification**: Bitcoin identifies users through a blockchain address based on **Public Key Cryptography (PKC)**.
- **Authentication**: Users authenticate their transactions via **digital signatures**, ensuring security and integrity.

This decentralized system ensures that identification and authentication are completely independent of centralized entities, enabling users to maintain full control over their assets and digital presence.

How Bitcoin Addresses Work

Bitcoin addresses are generated using **Public Key Cryptography**, playing a critical role in uniquely identifying user accounts. The process works as follows:

1. **Generate a Private Key**: A 256-bit random number is created as the private key. This key must remain secret.
2. **Derive the Public Key**: Using an elliptic curve cryptography algorithm, the public key is derived from the private key. The public key is openly shared with the world.
3. **Generate the Address**: A hashing function is applied to the public key to produce the unique Bitcoin blockchain address.



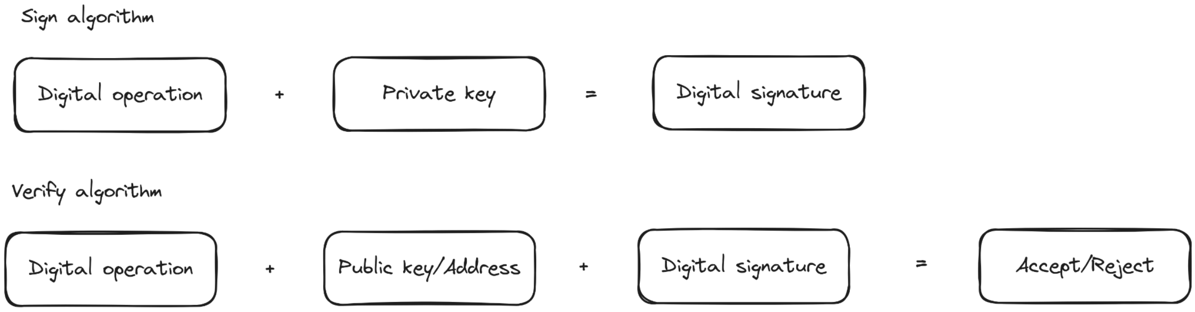
Key and address generation Algorithm

Example of Bitcoin address:

- 1GrkYWRfTVaj5996gitTz3yWFAjtE9cZMz
- 1Bir32WNYixVaE282o6krQFXH9DN57FeMZ

Digital signatures are used to authenticate user transactions on the ledger. They have three main services:

- Authentication: Make sure the message was created by a known sender.
- Non-repudiation: The sender cannot deny sending the message.
- Data integrity: The message wasn't altered in transit.



Digital Signature Algorithm

In the Bitcoin identity model, humans prove ownership and control of a Bitcoin account through the private key. When a user intends to perform a transaction, they sign it using their private key, generating a **digital signature**. The blockchain ledger verifies this signature before executing the transaction and updates the state based on the transaction data. This trust-minimized process ensures both security and transparency.

Wallets: The Gateway to Digital Sovereignty

Wallets are software clients that allow users to manage their identifiers, authenticate their identity, and issue blockchain transactions. Key functions of wallets include:

- Managing **seed phrases, private keys, and addresses**.
- Signing blockchain transactions.
- Propagating transactions to the blockchain network.

A crucial feature of wallets is **freedom of choice**: users can select any wallet provider and switch between them with just a few clicks, without requiring permission from a centralized trusted third party. This flexibility underscores the decentralized nature of the Bitcoin identity model.

Sovereignty, Digital Ownership, and Self-custody,

Bitcoin's identity model empowers individuals with **sovereignty over their identity on the internet**. Users own their private keys, granting them complete control over their digital presence and assets. This self-sovereign approach is critical for maintaining true ownership of values and assets on the internet.

Without a trust-minimized identity model like Bitcoin’s, even a decentralized ledger cannot guarantee true ownership. This principle is captured in the popular phrase: **“Not your keys, not your coins.”** By owning their private keys, individuals retain full ownership and control over their digital assets.

Bitcoin’s identity model has introduced a new behavioral paradigm in the digital world: **self-custody**. This concept enables individuals to manage and safeguard their assets independently, without relying on trusted third parties. With self-custody, humans take full responsibility for managing their private keys and seed phrases, ensuring direct ownership of their assets.

Challenges in User Experience

While self-custody represents true digital ownership, its current user experience (UX) remains a significant barrier to adoption. Many cryptocurrency holders rely on centralized custodians, such as exchanges (CEXs), to safeguard their assets. This reliance stems from the **complexity and user-unfriendliness** of managing private keys and seed phrases.

Another challenge lies in making **on-chain protocols more accessible** for everyday users while preserving self-custody principles. Striking a balance between usability and security remains a critical area of innovation for the broader adoption of trust-minimized identity and self-custody.

3. Ethereum: Identity Protocols

Different projects in the Ethereum ecosystem are working to provide identity services for humans via on-chain protocols as a public good on the internet independent of centralized trusted third parties.

Ethereum naming service(ENS)

The **Ethereum naming service (ENS)** is a decentralized naming protocol built on the Ethereum network. Ethereum protocol identifies humans via addresses that are not human-friendly and are very complex to remember. ENS allows users to map long, complex Ethereum addresses to simple, memorable ones, enhancing user experience and facilitating wider adoption of blockchain technologies. It maps machine-readable identifiers such as Ethereum addresses to human-readable names like 'alice.eth'.

Ethereum Attestation Service(EAS)

The **Ethereum Attestation Service (EAS)** is an open, decentralized protocol that enables users and applications to create, verify, and manage **on-chain attestations** on the Ethereum blockchain. Attestations are verifiable claims or statements about a person, entity, or event—for example, proof of identity, certifications, achievements, or transaction records. EAS provides a standardized framework for generating these attestations, which can be publicly accessible, transparent, and cryptographically secured, making them highly reliable and tamper-proof.

By leveraging EAS, users and developers can create a wide range of verifiable claims that can be used across different dApps, enabling **trust-minimized** interactions without relying on centralized authorities. This service supports use cases in **identity verification, decentralized reputation systems, credential management,** and more, paving the way for a more secure, interoperable, and trust-minimized ecosystem.

KYC

KYC DAO is a decentralized compliance protocol designed specifically for Web3, addressing the increasing need for **trust and regulatory compliance** in the crypto ecosystem. As blockchain technology matures and gains mainstream attention, establishing trust is becoming a competitive advantage and a foundational aspect of Web3's future.

KYC DAO enables **Web2-compliant identity verification** in a way that aligns with Web3 principles. At the core of this protocol are **kycNFTs**—soulbound, multi-chain NFTs minted by trusted KYC DAO issuers. These NFTs act as composable proofs of compliance, allowing users to verify their identity in a secure, privacy-preserving manner without revealing personal information on-chain. This framework provides the foundational layer for building a **trusted ecosystem**, linking pseudonymous identities in Web3 to verifiable compliance.

KYC DAO’s approach allows users to carry a single, reusable proof of KYC status across multiple decentralized applications, streamlining compliance for DAOs, DeFi, and NFT marketplaces. The protocol emphasizes **privacy** by keeping all personally identifiable information (PII) off-chain, preserving user anonymity while meeting regulatory standards.

Proof of personhood

As AI continues to advance, distinguishing human content from AI-generated content becomes essential. **Proof of Personhood (PoP)** provides a way to verify an individual's humanness and uniqueness, establishing trust in online interactions and protecting against issues like **Sybil attacks** (where one user creates multiple fake accounts) and **AI-generated disinformation**.

Why Proof of Personhood Matters

PoP helps prevent abuse by verifying users as real, unique humans. It limits Sybil attacks and enables platforms to filter for authentic human content, reducing the spread of AI-generated disinformation. Achieving PoP at scale would enhance digital trust and security.

Worldcoin’s Solution: World ID

Worldcoin is pioneering PoP with **World ID**, an open identity protocol that verifies uniqueness and humanness while preserving privacy:

- **The Orb**: A biometric device that generates an **iris code** to ensure each World ID is unique and cannot be duplicated.
- **Privacy with Zero-Knowledge Proofs (ZKP)**: World ID allows users to prove they are real humans without revealing personal data. ZKPs ensure privacy by preventing third parties from accessing or tracking identity data.

World ID acts as a digital passport, letting users prove their human identity across platforms securely and privately, establishing a foundation for PoP on a global scale. Worldcoin's approach to PoP offers a robust solution to the challenges of trust and authenticity in the digital age.

POAP, short for Proof of Attendance Protocol

POAP (Proof of Attendance Protocol) is a decentralized protocol that enables individuals to create and collect digital badges or "attendance tokens" as proof of participation in events, both virtual and in-person. Each POAP token is an NFT that is unique to the event and verifiable on the blockchain, creating a permanent record of attendance. These tokens serve as digital mementos that can represent a wide range of experiences, from conferences and workshops to online meetups and gaming events. POAPs offer a novel way to reward engagement, foster community, and build digital identity by accumulating a history of interactions and shared experiences across different platforms.

Gitcoin Passport

Gitcoin Passport is a decentralized identity system designed to verify a user's trustworthiness and credibility within Web3 communities. By aggregating multiple identity "stamps" or verifications from various sources, Gitcoin Passport establishes a user's unique identity and resistance to Sybil attacks (multiple fake identities). This digital passport is particularly valuable for projects seeking fair distribution in community incentives, grants, or governance voting by ensuring participants are legitimate and unique. Gitcoin Passport allows users to prove their authenticity across different platforms in a privacy-preserving manner, fostering trust and transparency within decentralized ecosystems.

Safe: The Leader in Ethereum Multi-Sig Wallets

Multi-signature (multi-sig) wallets enhance security by requiring approval from multiple parties before a transaction can be executed. Transactions are only authorized when a predefined number of signers confirm the operation. This mechanism ensures that no single entity can unilaterally control the account, offering a higher level of trust and security for shared accounts.

Safe (formerly Gnosis Safe) is the leading multi-sig wallet in the Ethereum ecosystem, offering unparalleled security and flexibility for managing smart accounts. A key innovation of Safe is its **modular architecture**, which allows developers to attach additional modules to Safe accounts. These modules are essentially smart contracts that can add customizable, programmable logic to the account. Examples include:

- **New Authentication Schemes**: Modules can enable advanced signers, such as hardware wallets, biometric signers, or other novel authentication methods, to issue transactions.
- **Account Recovery Mechanisms**: Safe supports adding recovery workflows, ensuring users can regain access to their account if a key is lost.

This modular design enables Safe accounts to adapt to various user needs, providing flexibility beyond traditional multi-sig setups.

Sign-In with Ethereum(SIWE)

Sign-In with Ethereum(SIWE) defines a standard for using Ethereum-based identity by off-chain services like web2 platforms.

Sign-In with Ethereum describes how Ethereum accounts are used to authenticate within off-chain services by signing a standard message format parameterized by scope, session details, and security mechanisms (e.g., a nonce).

4. Ethereum: Account Abstraction(AA)

Account abstraction(AA)

On the Ethereum blockchain, humans typically use **Externally Owned Accounts (EOAs)** to interact with on-chain protocols like stablecoins or decentralized exchanges. However, EOAs are **static and limited**, controlling account ownership solely through private keys. This rigid model makes it difficult to program more complex ownership logic, posing a significant barrier to **scalable blockchain adoption** in society.

The Importance of Trust-Minimized Account Management

Trust-minimized account ownership and management are critical because they directly impact the concept of **self-custody**—the foundation of owning your coins, tokens, and identity. This principle is encapsulated in the popular phrase: **“Not your keys, not your coins.”** Ethereum defines two account types:

- **Externally Owned Accounts (EOAs):** Controlled by private keys.
- **Smart Contract Accounts (SCAs):** Controlled by code.

While SCAs offer more flexibility, EOAs dominate today’s ecosystem, limiting the scope of what users can do with their accounts.

What is Account Abstraction (AA)?

In computer science, **abstraction** involves separating an idea from its implementation, offering generality and power through a simple interface while hiding underlying complexity. **Account Abstraction (AA)** applies this concept to Ethereum accounts, providing a unified interface for managing accounts while abstracting the complexities of ownership, control, and management.

With AA, account management is decoupled from key management. Instead of proving account ownership through a private key, AA allows users to prove ownership through **programmable logic**. By leveraging smart contracts, AA enables users to program any logic governing the ownership and management of their accounts.

The Power of Account Abstraction

AA transforms Ethereum accounts by enabling **programmable self-custody**. With AA, humans can:

- Manage their identity and accounts through trust-minimized, on-chain programs.
- Deploy and execute smart contracts on a decentralized infrastructure (Ethereum) that prioritizes **censorship resistance**.

In this model:

- Humans are identified by an **Ethereum address**.
- A **smart contract** attached to the address by which human implements authentication mechanisms and additional logic, such as authorization or interactions with on-chain protocols (e.g., trading on Uniswap, managing stablecoins, voting on governance proposals).

The Shift to Programmable Ownership

AA represents a paradigm shift: from **static private key ownership** to **programmable, flexible account control**. By unlocking new capabilities for account management, AA makes blockchain technology more accessible and scalable, empowering individuals to fully own and manage their digital presence in a decentralized, trust-minimized ecosystem.

Account Abstraction via ERC-4337

ERC-4337 introduces **Account Abstraction (AA)** without requiring any changes to the core Ethereum protocol by leveraging an **alternative mempool**. This innovative approach enables the benefits of AA to be implemented seamlessly alongside existing Ethereum infrastructure, preserving decentralization and compatibility.

Key Components of ERC-4337

1. **Bundler**
    - The Bundler aggregates user operations from the alternative mempool into a single transaction. This process reduces the overhead of individual transactions while enabling smart accounts to execute operations efficiently.
2. **Paymaster**
    - The Paymaster introduces flexibility in transaction fee payments, allowing fees to be paid in tokens other than ETH or even sponsored by third parties.

This is crucial for improving **user experience** by removing barriers like the need to hold ETH for gas.

3. **Entrypoint**
   - The Entrypoint smart contract acts as the central hub for validating and executing user operations. It ensures that all operations conform to predefined rules and logic, maintaining security and trustlessness in the system.
4. **Embedded Wallet**
   - ERC-4337 introduces wallets as programmable entities directly embedded into the Ethereum ecosystem. These wallets operate as **smart contract accounts (SCAs)**, enabling advanced features like account recovery, multi-sig setups, and programmable logic for managing assets and identity.

Why ERC-4337 Matters

By implementing AA through an **alternative mempool**, ERC-4337 avoids modifying the Ethereum protocol, ensuring smooth integration with existing infrastructure. This approach unlocks powerful features for **smart accounts** and **self-custody wallets**, including gas abstraction, enhanced security, and programmable ownership.

ERC-4337 represents a significant milestone in making blockchain more **accessible and user-friendly**, paving the way for scalable, trust-minimized account management on Ethereum.

Account Abstraction via EIP-7702

EIP-7702 introduces **Account Abstraction (AA)** by extending the capabilities of Externally Owned Accounts (EOAs) to include programmable logic, creating a hybrid model that combines the simplicity of EOAs with the flexibility of Smart Contract Accounts (SCAs). This innovation allows EOAs to retain their core functionality while enabling **smart account features** without fundamentally altering Ethereum's architecture.

Key Features of EIP-7702

1. **Programmable EOAs**
   - EIP-7702 enhances EOAs by enabling them to incorporate **smart contract-like logic**, allowing for advanced features such as transaction batching, account recovery, and fine-grained permissions while maintaining compatibility with existing tools and infrastructure.
2. **Hybrid Accounts**
   - EIP-7702 creates a new category of accounts called **Hybrid Contract Accounts (HCAs)**. These accounts combine the simplicity of EOAs with the programmability of SCAs, offering users the best of both worlds—flexibility and ease of use.
3. **Backward Compatibility**
   - One of the key benefits of EIP-7702 is its compatibility with existing Ethereum wallets and dApps. This ensures that users can adopt advanced features

without needing entirely new infrastructure, simplifying the transition to more sophisticated account management.

4. **Decoupled Authentication and Ownership**
    - By separating account ownership from key management, EIP-7702 allows users to implement **custom authentication mechanisms**. This includes session keys, multi-sig setups, and other programmable authentication schemes that enhance security and usability.

Why EIP-7702 Matters

EIP-7702 bridges the gap between EOAs and SCAs, enabling **programmable self-custody** without requiring users to switch entirely to smart contract wallets. This evolution empowers developers to build **more intuitive and flexible applications**, while users gain access to advanced features like gas abstraction, session keys, and programmable permissions—all while maintaining a familiar user experience.

Native account abstraction:

Native Account Abstraction (RIP-7560)

Native Account Abstraction (RIP-7560) is a long-term proposal to enshrine **ERC-4337** directly into the Ethereum protocol, requiring core changes for a more streamlined and efficient implementation. Unlike ERC-4337, this native approach removes reliance on intermediary steps, allowing contracts to authorize transactions and pay gas fees directly.

Key features include:

- A new transaction type for efficient processing.
- Non-sequential nonce support for advanced use cases.
- Backward compatibility with ERC-4337 for a seamless transition.

By addressing limitations like high gas costs and limited node participation, RIP-7560 aims to make Ethereum interactions more scalable, efficient, and user-friendly. The proposal is championed by Ethereum co-founder **Vitalik Buterin** and leading developers, marking it as a major step toward Ethereum's evolution.

EIP-7701: Native Account Abstraction with EOF

EIP-7701 is a proposal to implement **Native Account Abstraction (AA)** by leveraging the **EVM Object Format (EOF)**to enhance account functionality. Unlike previous approaches such as ERC-4337 or RIP-7560, EIP-7701 introduces a cleaner and more efficient method for separating validation and execution logic within smart contract accounts by utilizing **distinct code sections** introduced by EOF.

Key Features

1. **EOF Integration**:
    - Uses EOF’s "code sections" to distinguish between validation and execution logic, avoiding reliance on method selectors (as in RIP-7560), which can lead to technical debt.
2. **Native AA Implementation**:
    - Smart Contract Accounts explicitly define roles such as sender, paymaster, or deployer using entry points marked by EOF.
    - Introduces a structured mechanism to validate and execute transactions within native AA contexts.
3. **SSZ Data Encoding**:
    - Protocol uses SSZ encoding for transaction data, enabling efficient validation and decision-making for smart contract accounts.
4. **Backward Compatibility**:
    - Maintains compatibility with EIP-3540 while introducing enhancements specific to native account abstraction.

zkSync Era: Scaling Ethereum with ZK Rollups

zkSync Era is a **Layer 2 solution** that leverages **Zero Knowledge Rollups (ZK Rollups)** to scale Ethereum, enabling faster and cheaper transactions while maintaining security. It bundles multiple transactions off-chain, verifies them with cryptographic proofs on-chain, and inherits Ethereum's decentralization and trust model.

With **EVM compatibility** and native support for **Account Abstraction (AA)**, zkSync Era allows developers to build flexible, user-friendly dApps, while reducing gas costs and enhancing scalability. It’s a key step toward making Ethereum more accessible and efficient for real-world use cases like DeFi, gaming, and NFTs.

Account abstraction Ethereum standards

There are many major players in the Ethereum account abstraction ecosystem that are building different components and products: smart wallet by base, ZeroDev, Biconomy, Rhinestone, Safe, Alchemy, Candide, WalletConnect(reown), Stackup, Magic, Etherspot, Pimlico, OKX, etc.

The Ethereum community is working on different standards to bring account abstraction to the next level.

- **EIP-3047**: Introduces improvements for enabling seamless user interactions by expanding account capabilities with minimal overhead, focusing on enhancing Ethereum account functionality.
- **ERC-4337**: Implements Account Abstraction (AA) via an alternative mempool, enabling programmable smart accounts without requiring changes to Ethereum's core protocol.
- **EIP-5792**: Proposes advanced mechanisms for on-chain account recovery, enhancing security and usability for smart contract and user accounts.

- **ERC-6492**: Enables trustless off-chain wallet interactions by introducing account abstraction features that simplify account management and transactions.
- **ERC-6900**: Introduced by Alchemy, this standard focuses on modularity for account management, allowing for customizable and upgradeable smart account features.
- **EIP-7702**: An alternative to EIP-3047, EIP-7702 enables hybrid accounts by introducing programmability to Externally Owned Accounts (EOAs) without altering their simplicity.
- **EIP-7701**: Leverages EOF (EVM Object Format) to enable native account abstraction, separating validation and execution logic for cleaner and more efficient smart accounts.
- **ERC-7579**: Defines a modular framework for account extensions, allowing the addition of programmable features like session keys or gasless transactions.
- **ERC-7484**: Introduces enhanced multi-signature account functionality with advanced validation schemes and modular transaction authorization.
- **ERC-7677**: Focuses on interoperability for smart contract wallets, standardizing account interaction protocols across dApps and chains.
- **ERC-7679**: Enables advanced gas abstraction, allowing transaction fees to be paid in tokens or sponsored by third parties.
- **ERC-7682**: Proposes mechanisms for decentralized account delegation, enabling secure sharing of account functionality without compromising ownership.
- **ERC-7715**: Implements session keys for programmable temporary account permissions, improving UX and security for high-frequency interactions.
- **ERC-7739**: Enhances the security and recovery features of smart accounts by introducing a layered validation system.
- **ERC-7555**: Standardizes gas-optimized multi-transaction bundling for smart accounts, improving efficiency for complex operations.
- **ERC-7562**: Introduces token-based account abstraction features, integrating asset management into programmable smart accounts.
- **RIP-7212**: Aims to standardize recovery mechanisms across Ethereum accounts, focusing on trustless and decentralized recovery solutions.
- **RIP-7711**: Expands account abstraction capabilities by introducing enhanced on-chain validation logic for smart accounts.
- **RIP-7560**: Proposes native account abstraction enshrined in Ethereum's protocol, streamlining account functionality for efficiency and scalability.

Signers and keys management

A **signer** is an abstract object used to authenticate humans on the internet. It can implement various authentication mechanisms, such as ownership of private keys, providing flexibility and security. In the context of smart contract wallets, different types of signers are utilized, each with unique trade-offs. Examples include:

- **Passkeys**: A passwordless authentication standard built on top of WebAuthn, often paired with biometric methods like FaceID or TouchID.
- **Hardware Signers**: Devices designed specifically for secure authentication, such as hardware wallets.

- **SSS/MPC Signers**: Signers based on Shamir's Secret Sharing (SSS) or Multi-Party Computation (MPC) schemes, such as those offered by Web3Auth.
- **Passwordless Authentication Signers**: Methods like email OTPs, magic links, social logins, WebAuthn, or third-party identity providers for seamless and secure access.

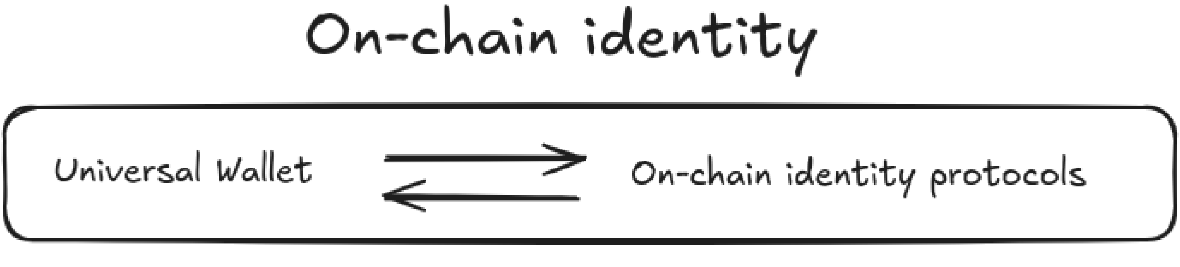
These varied options allow for adaptable and user-friendly authentication while maintaining the underlying security needed for smart contract wallets.

5. On-chain identity paradigm: the missing native identity layer of the Internet

Blockchain technology proposes a new identity paradigm on the internet. Identity services will be provided via on-chain protocols as a public good. As a human, with just a device and an internet connection without any permission of any entity(permissionless and open), you get access to identity services(neutrality) that are not controlled by any centralized trusted third parties. It is the on-chain identity paradigm that shares the same principles of on-chain finance(DeFi), on-chain value, or on-chain governance. On-chain identity protocols are deployed and executed on a decentralized public good infrastructure.

The on-chain identity will be the native identity layer/stack of the internet. The on-chain identity protocols stack is based mainly on Blockchain, smart contracts, and Zero-knowledge proofs(ZKPs).

The on-chain identity paradigm consisted of two main components:
- On-chain identity protocols
- Universal wallets



On-chain identity paradigm

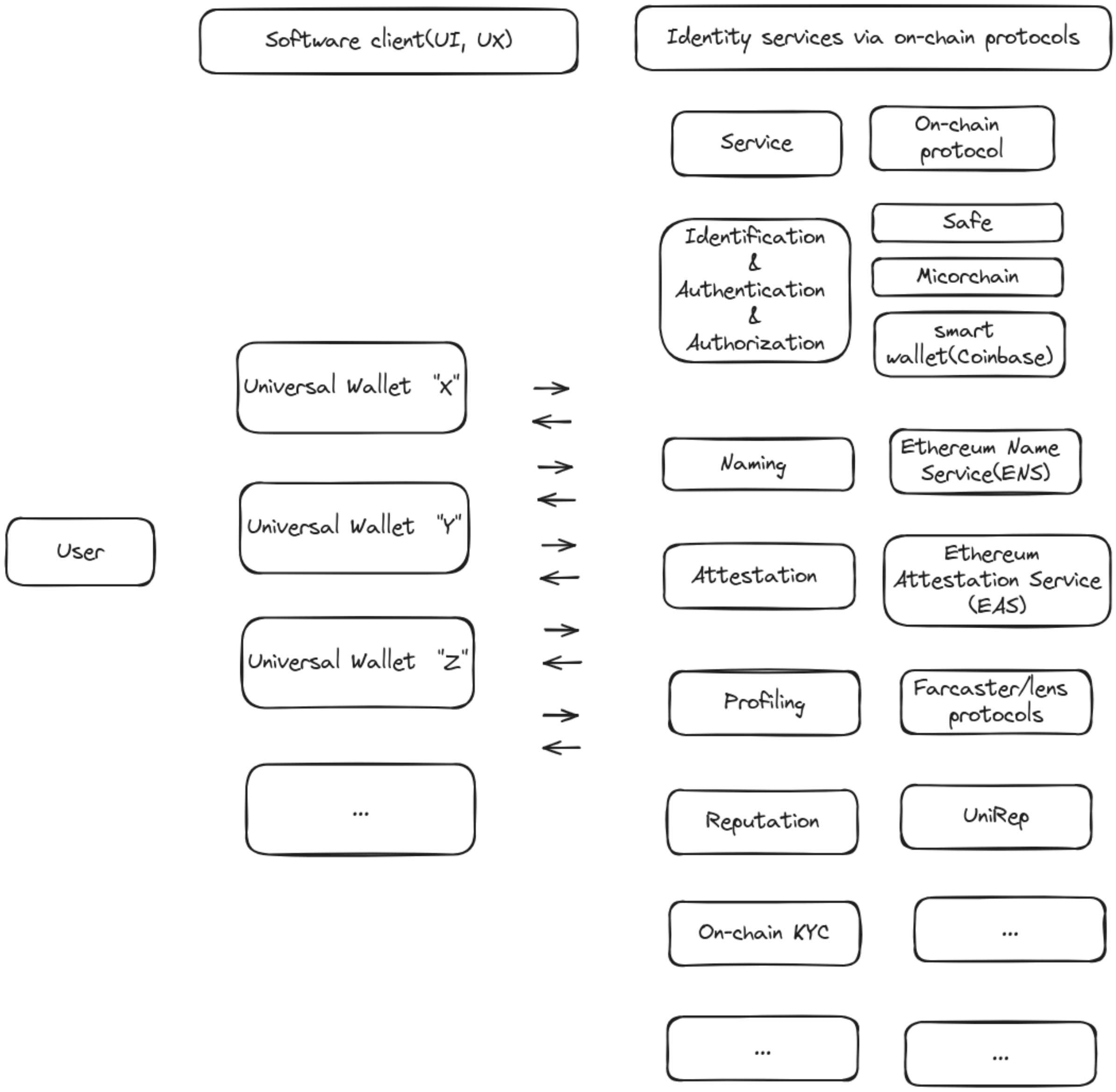
On-Chain Identity Protocols

On-chain protocols offer a wide range of identity services, including **identification, authentication, authorization, naming services, profiling, reputation, attestation**, and more. These protocols empower individuals to own and control their digital identity by leveraging blockchain infrastructure.

On-chain identity protocols can be broadly categorized into two types:

1. **Identity Management Protocols**: These protocols focus on managing and governing an individual’s digital presence, including **identification, authentication, authorization**, and **account recovery**.
2. **Identity Service Protocols**: These provide additional services surrounding digital identity, such as **reputation systems, attestations, naming services**, and **profiling**.

Wallets act as the client software enabling users to interact with these protocols during their daily on-chain activities. Much like browsers or mobile phones today, wallets serve as the gateway for managing and utilizing on-chain identities seamlessly.



On-chain identity services protocols

6. Proof systems(verifiable computation(VC) and zero-knowledge(ZK) protocols) empower Smart contact wallet capabilities

Traditional wallets, built on **public key cryptography (PKC)**, focus on providing cryptographic services for data, such as signing transactions. This functionality forms the core of the first generation of wallets.

With the advent of **Account Abstraction (AA)**, wallets have evolved into **Smart Contract Wallets (SCWs)**, powered by standards like **ERC-4337**. SCWs are programmable wallets based on smart contracts, enabling advanced functionalities beyond static cryptographic services.

Enhancing SCWs with Off-Chain Computation

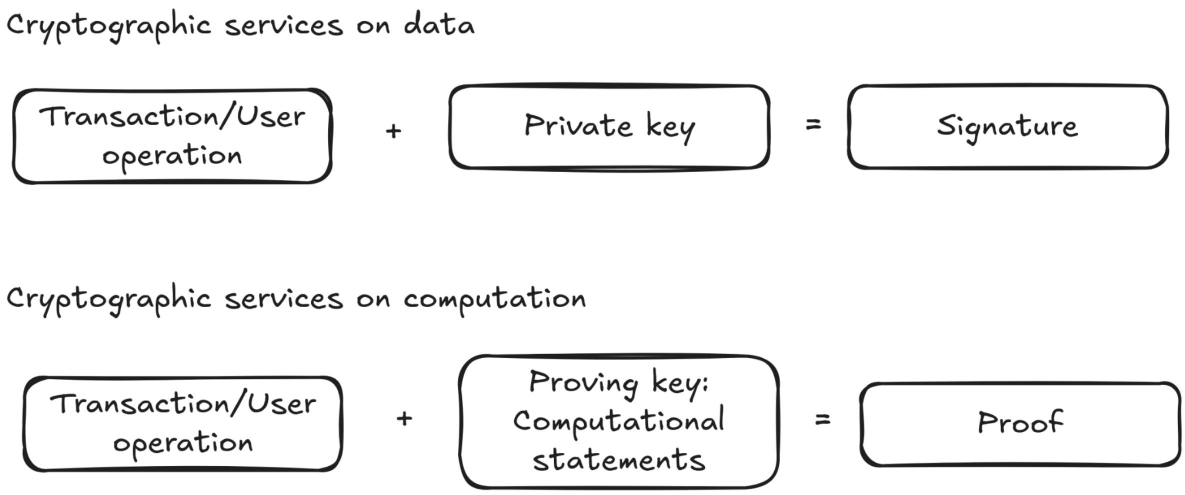
While SCW functionalities are typically implemented on-chain via smart contracts, a significant part of these operations can be moved to **off-chain programs executed in the browser**, secured by cryptography. This approach enhances SCW capabilities by incorporating **cryptographic services for computation**, leveraging **proof systems** and **Merkle trees** to enable more scalable and flexible operations.

- **Proof Systems**: Allow two parties to exchange general computational statements in a trust-minimized manner without relying on a trusted third party. These statements are encoded using **Zero Knowledge (ZK) circuits**, ensuring only the necessary information is shared.
- **Off-Chain Validation**: User operation (UserOp) validation logic can be programmed off-chain using ZK circuits and Merkle trees. The cryptographic proof is then generated in the browser, proving that the UserOp meets its validation logic.

Streamlining On-Chain Execution

Once the cryptographic proof is generated off-chain, the smart contract wallet only needs to verify the proof’s validity on-chain. This approach reduces on-chain computational costs while maintaining the security and trust-minimization principles of blockchain systems.

This evolution from on-chain-centric SCWs to a hybrid model of **on-chain verification and off-chain computation** represents the next leap in wallet technology, enabling scalable, efficient, and user-friendly interactions.


Cryptographic services on data
Transaction/User operation
+
Private key
=
Signature
Cryptographic services on computation
Transaction/User operation
+
Proving key: Computational statements
=
Proof

Cryptographic services on data and computation

Wallets are evolving beyond simple tools for managing keys to becoming powerful, programmable systems that offer **user-friendly, trust-minimized experiences**. By leveraging **Account Abstraction (AA)** and **proof systems**, wallet logic can now be executed both on-chain and off-chain, enhancing scalability, security, and usability. There are well-defined interfaces for dealing with cryptographic services on data in Ethereum, it is crucial to define interfaces for accessing cryptographic services on computation with different proofs system protocols.

Trust-Minimized Wallet Logic

Wallets can now implement business logic (e.g., a more intuitive self-custody experience) **off-chain** in a **trust-minimized** manner using **cryptographic security**. This allows for scalable execution at low gas costs while maintaining robust security principles.

- **On-Chain Logic**: Managed through smart contracts on the ledger, secured by cryptoeconomic incentives.
- **Off-Chain Logic**: Executed client-side via **Verifiable Computation (VC)** or **Zero Knowledge Proofs (ZKP) protocols**, secured through cryptographic mechanisms.

The Role of Proof Systems

At Microchain Labs, we envision **proof systems** (ZK circuits and Merkle trees) as a core component of universal wallet architecture. Wallet logic, such as transaction validation or user operation management, is programmed into ZK circuits and executed off-chain. Cryptographic proofs are then used to verify the validity of computational statements on-chain. This approach allows for implementing complex wallet features **at scale**, while minimizing on-chain overhead.

The Evolution of Self-Custody Wallets

1. **Static Self-Custody Wallets**:

    - Identifier + Private Key/Public Key + Client Software.
    - Ownership is defined by a private key or seed phrase.
2. **Programmable Self-Custody Wallets** (Account Abstraction):
    - Identifier + Smart Contract + Client Software.
    - Ownership extends to programmable logic via AA.
3. **Programmable Self-Custody Wallets with ZK**:
    - Identifier + Smart Contract + Proving Keys/Verification Keys/Merkle Tree + Client Software.
    - Ownership scaled to include advanced features with cryptographic proofs for validation.

This evolution enables wallets to balance **security, scalability, and usability**, paving the way for **next-generation self-custody** tools that are user-centric and cost-efficient. With programmable logic and cryptographic proofs, the future of wallets is set to redefine how we interact with blockchain ecosystems.

7. Microchain Labs

At Microchain Labs, we are researching and developing **trust-minimized off-chain techniques** to create a **Web2-like user experience (UX)** for self-custody. By leveraging **programmable cryptography**—including **proof systems (VC/ZKP protocols)** and **Merkle trees**—we aim to deliver scalable, secure, and user-friendly solutions.

Key Concepts: Limited Scope Accounts (LSAs) and ZK Session Keys

Our work focuses on introducing innovative concepts like **Limited Scope Accounts** and **ZK Session Keys**, striking the best balance between enhanced UX and robust safety for users. These tools push the boundaries of what self-custody can achieve, bringing smart contract wallets closer to mainstream usability.

To improve scalability and reduce costs, we move complex **smart contract wallet logic off-chain** to the client side. Computations are performed in the browser, and the results—along with their validity—are communicated to the smart contract account via **cryptographic proofs**.

- **ZKPs (Zero Knowledge Proofs)**: Ensure trust-minimized state machine through cryptographic security, enabling **scalable operations** at **low gas costs**.

ZK technology is **revolutionizing self-custody UX**, transforming smart contract wallets by making them more efficient, secure, and user-friendly. By combining trust-minimized off-chain computation with scalable cryptographic solutions, we are taking smart contract wallets to the next level, paving the way for widespread adoption. We’re implementing different concepts as ERC-7579 Modules. ERC-7579 is a modular smart account standard in the Ethereum ecosystem, and modules are its building blocks.

ZK session keys

Session keys are temporary keys within a smart contract wallet that come with specific, user-defined permissions. They are designed to handle a precise set of operations under predefined conditions, ensuring both security and usability.

Key Features of Session Keys

- **Scoped**: Restricted to specific on-chain actions.
- **Ephemeral**: Valid only for a set duration.
- **User-Authorized**: Fully controlled and approved by the user.

Smart Session Manager

Biconomy and Rhinestone co-developed a **Smart Session Manager**, a robust on-chain permissions system that is **composable and interoperable** across all **ERC-7579-compliant smart accounts**. This solution is:

- **Highly Customizable**: Can adapt to diverse use cases.
- **Compatible with ERC-7715**: Works seamlessly with advanced session key standards.

ZK Session Keys: Off-Chain Permissions Validation

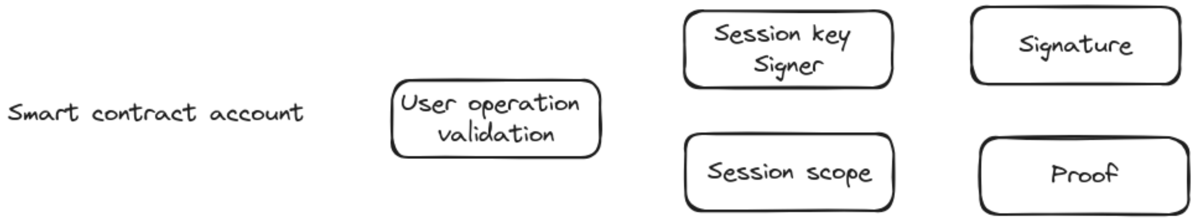
ZK Session Keys take the concept further by introducing an **off-chain permission system**. Using **Zero Knowledge Proofs (ZKPs)** on the client side, they validate user operations off-chain, ensuring that the session's permissions are met before any action is executed. This approach minimizes on-chain computation, reduces gas costs, and enhances scalability.

How Validation Works

Validation combines two cryptographic services:

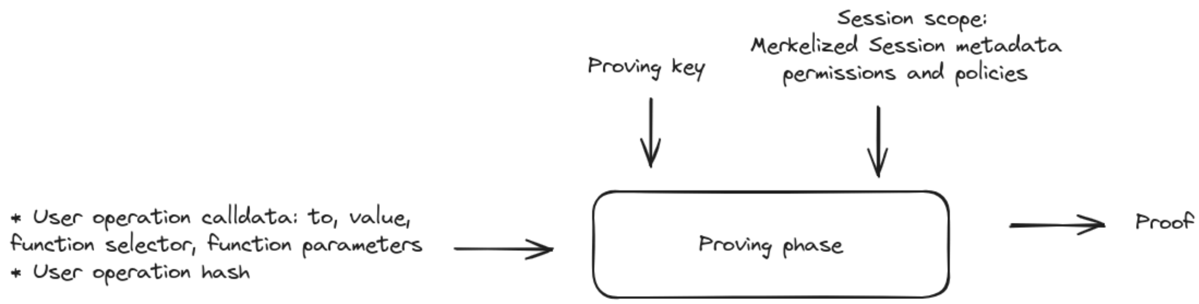
- **Signature**: Ensures the authenticity of the data.
- **Proof**: Verifies the computational logic of the user operation.

With ZK session keys, we can create a trust-minimized, scalable, and secure framework for managing user operations, bridging the gap between advanced functionality and seamless user experiences.



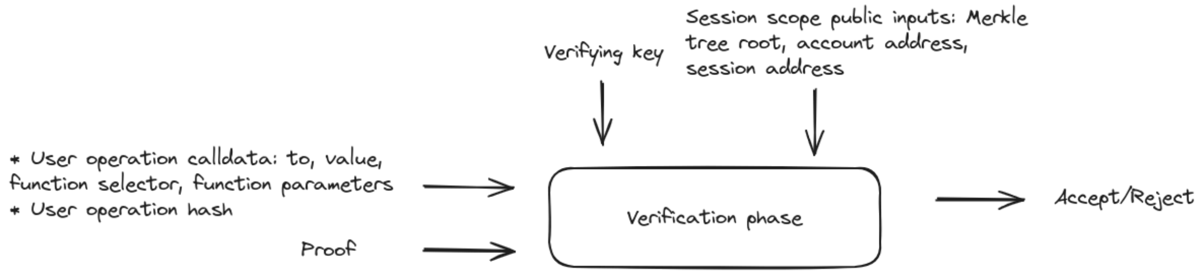
User operation validation via zk session keys

We have a proving phase that generates an off-chain proof to prove that the user's operation matches the session permissions and policies.


Session scope:
Merkelized Session metadata
permissions and policies
Proving key
* User operation calldata: to, value,
function selector, function parameters
* User operation hash
Proving phase
Proof

Proving phase

Then, we verify the proof on-chain.


Session scope public inputs: Merkle
tree root, account address,
session address
Verifying key
* User operation calldata: to, value,
function selector, function parameters
* User operation hash
Proof
Verification phase
Accept/Reject

Verifying phase

The session Merkle tree root must be part of the on-chain state. We can program any computational statement around Merkelized session data and user operation calldata via ZK circuits.

Limited Scope Accounts (LSAs)

Our research and development (R&D) introduced **Limited Scope Accounts (LSAs)**—a trust-minimized, off-chain concept designed to create a more **user-friendly self-custody experience**. LSAs fundamentally redefine how Ethereum accounts operate by focusing on **customized, human-centric behaviors**.

Traditional Ethereum accounts can perform any operation on the ledger, as long as the account signer authorizes it. This default behavior lacks customization and fails to reflect the real-world behaviors and intentions of the account owner.

What are Limited Scope Accounts?

Limited Scope Accounts reverse this default behavior.

- **Default State**: An LSA cannot perform any operation on the ledger.
- **Customized Behavior**: Over time, specific capabilities and behaviors are defined based on the account owner's needs.

This approach allows LSAs to be tailored to each user's **intended on-chain behavior**, aligning more closely with their **real-world identity and actions**.

Key Use Cases for LSAs

1. **Enterprise Governance**:
    - Enterprises can configure employee accounts to function based on predefined governance rules. Employees use smart accounts on-chain according to the enterprise's operational policies.
2. **Public Identity Management**:
    - Governments can use LSAs to create on-chain identities for citizens, limiting account functionality to specific government-approved or regulated protocols.
3. **Gaming Protocols**:
    - Gamers can define the behavior of their accounts for specific on-chain gaming protocols, ensuring their smart accounts only interact with approved games and follow predefined rules.

Why LSAs Matter

Limited Scope Accounts enable **customizable, secure, and intent-driven on-chain behaviors** for Ethereum accounts. By introducing this layer of validation and restriction, LSAs bring **self-custody closer to real-world identity dynamics**, empowering users and organizations to operate securely, efficiently, and in alignment with their unique needs

ZK policy engine for smart accounts

At Microchain Labs, we are redefining smart contract wallets by taking **complex wallet computations off-chain** using **proof systems**. This approach enables us to build **advanced wallet features** that are scalable, cost-efficient, and secured by **cryptographic proofs**.

How It Works

Instead of relying solely on smart contracts for wallet functionality, we utilize **off-chain state machines** which are secured by cryptography to execute computations on the client side (e.g., browsers or mobile applications). These programs:

1. Generate computational results based on specific inputs.
2. Produce **cryptographic proofs** (Client-Side Proofs) verifying the computation's validity.
3. Offload complex logic to the client while maintaining **on-chain trust** through proof validation.

On-chain, we manage verification keys, state roots, and the validation of cryptographic proofs. **Proof systems** like Zero-Knowledge Proofs (ZKPs) and Verifiable Computations (VCs) ensure trust-minimization through cryptographic security while significantly reducing gas costs.

Introducing the ZK Policy Engine

Our **ZK Policy Engine** is designed to handle account and key scopes, permissions, and policies at various levels.

- **First Version**: Focuses on managing accounts and policies for smart accounts.
- **Future Enhancements**: Incorporates advanced tools like:
  - **zkTLS**: Web proofs for secure online interactions.
  - **ZK Coprocessor**: Enables reading and computation over historical on-chain data.
  - **zkEmail**: Email proofs for seamless identity verification.

Proof systems like **SNARKs** are disrupting the traditional smart contract wallet paradigm by enabling **trust-minimized, scalable solutions** that deliver a **user-friendly experience** at low cost. Our approach bridges the gap between on-chain and off-chain state management and computation, unlocking the full potential of smart contract wallets for everyday users.

Our Smart Accounts Ownership Framework

Humans behave differently based on culture, experience, and context—and this variability extends to their on-chain interactions. At Microchain Labs, we are introducing a **framework** that allows individuals to define and govern the **intended behavior of their blockchain accounts**.

Defining On-Chain Behavior

Through this framework:

- Users can **customize the behavior of their blockchain accounts**, ensuring the account only permits expected actions while **blocking unexpected or unauthorized operations**.
- Over time, individuals can discover, reflect on, and refine their **on-chain behavior** in various contexts.

This approach enables users to align their **on-chain presence** with their real-world behaviors, maintaining consistency between digital and physical identities.

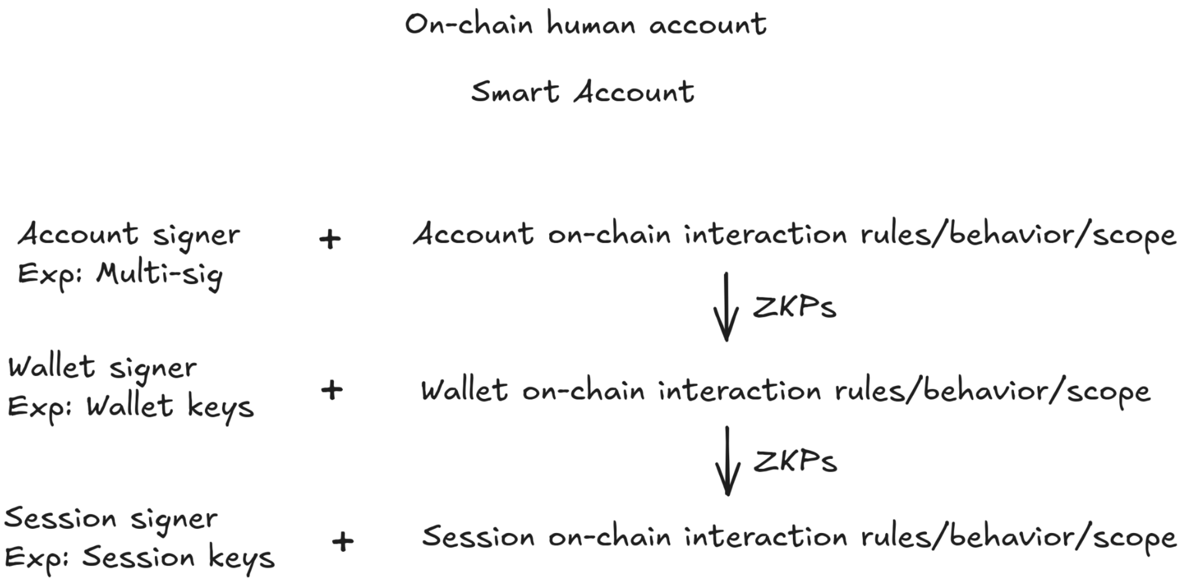
Self-Custody at Scale

The framework is built on the **programmable ownership and self-custody paradigm**, where each individual is responsible for managing their account and defining its scope and behavior. A **user-friendly self-custody experience** is only possible when on-chain accounts are tailored to reflect well-defined human behaviors.

Scaling with Proof Systems

By leveraging **proof systems** like Zero Knowledge Proofs (ZKPs), we are scaling this vision to enable secure, trust-minimized, and customizable account behavior at low cost. This is a critical step toward achieving **human-centric, user-friendly self-custody experiences** in the long term.

Self-custody is more than ownership—it's about empowering humans to govern their **on-chain presence** in ways that reflect their unique identities and behaviors. While the journey is complex, it is essential for creating a scalable and intuitive blockchain experience.


On-chain human account
Smart Account
Account signer
Exp: Multi-sig
+
Account on-chain interaction rules/behavior/scope
ZKPs
Wallet signer
Exp: Wallet keys
+
Wallet on-chain interaction rules/behavior/scope
ZKPs
Session signer
Exp: Session keys
+
Session on-chain interaction rules/behavior/scope

Microchain smart account ownership framework

In our framework, account management is divided into specific roles with clearly defined responsibilities, ensuring seamless and secure on-chain interactions:

Account Signer

- **Primary Role**: Responsible for managing the account's **scope, behavior, and capabilities** over time.
- **Key Responsibilities**:
  - Adding new wallet signers and defining their scope.
  - Updating on-chain expected behavior, context, and capabilities as needed.
- **Validation**: When adding a new wallet signer, the account must provide **cryptographic proof** (via ZKPs) that the wallet signer's scope aligns with the account's defined scope.
- **Limitation**: The account signer does not directly handle user operations.

Wallet Signer

- **Primary Role**: Executes user operations based on the wallet's scope and context.
- **Key Responsibilities**:

    - Proving, via **Zero Knowledge Proofs (ZKPs)**, that user operations are within the wallet's defined scope.
    - Adding new session keys for temporary or limited use cases.

Session Signer

- **Primary Role**: Executes specific user operations according to its **limited scope**.

Validation Framework

Every user operation within this structure is validated against the account's **well-defined behavior, scope, and rules**, ensuring that:

1. Operations align with the account's governance.
2. Trust-minimization is maintained through **ZKPs** for proof of compliance.

This layered approach ensures secure and scalable account management, enabling advanced self-custody features while maintaining flexibility and trust at every level of interaction.

8. Conclusion

"The Internet was built without an identity layer." — Kim Cameron, Microsoft Chief Identity Architect, *The Laws of Identity* (2005)

The current state of human identity on the internet is dominated by a few centralized, trusted third parties. This model stands in stark contrast to the principle that **identity is a fundamental human right**, one that should be owned and controlled by individuals, not corporations.

Ethereum addresses this critical gap by introducing a **native identity layer** for the internet through **on-chain protocols**, treating identity as a **public good**.

On-Chain Identity: A Paradigm Shift

The on-chain identity paradigm establishes a **decentralized, trust-minimized infrastructure** where identity services—such as authentication, authorization, and reputation—are provided as protocols rather than platforms. This approach ensures that identity is managed transparently and securely, free from the influence of centralized entities.

The Future of Digital Identity

As blockchain technology continues to evolve, it is clear that **human identity services** will increasingly rely on decentralized protocols rather than centralized providers. The role of the identity provider (IDP) will shift from private corporations to **protocols** operating on public, decentralized infrastructure, ensuring that identity remains a **human right** governed by individuals themselves.

Ethereum's vision for on-chain identity is a crucial step toward building a more equitable, secure, and decentralized internet.