Newton Protocol: Verifiable Automation Layer for Onchain Finance

By Sean Li

Table of Contents

- Newton Protocol: Verifiable Automation Layer for Onchain Finance
  - Introduction
  - Background
  - Problem
    - Broken UX and underutilized capital
    - Untrusted and risky automation
    - Missing primitives for crypto AI agents
  - Protocol Features
    - Why Verifiable Automation Matters
    - A Pragmatic Approach to Scalability, Security, and Decentralization
    - Applications and Use Cases
      - Long-Term Asset & Strategy Management
      - Short-Term Trading & Execution Automation
      - Commerce & Payments Automation
      - DAO & Institutional Automation
  - Protocol Participants
    - Developer
    - Operator
    - User
    - Validator
  - Architecture
    - Smart Accounts
    - zkPermissions
      - Data-Driven Execution Conditions
      - Risk and Price Sensitivity Checks
      - Transaction Volume and Timing Constraints
      - Default and Baseline Permissions
    - Execution Orchestrator
  - Developer Tooling & Extensibility
  - Development Priorities and Roadmap Focus
  - Closing Summary

Introduction

The first major hurdle in web3 was access. Magic solved it by eliminating seed phrases and browser extensions, making crypto usable for everyday users through inventing embedded wallets. But access alone is not adoption. Beyond onboarding, users face a fragmented, complex landscape where even simple tasks, bridging assets, staking, managing portfolios, require manual effort across multiple protocols. Billions of dollars flow through onchain systems daily, yet much of it remains idle or inefficiently managed because the complexity deters participation.

To unlock the next phase of adoption, crypto needs more than better interfaces - it needs **application abstraction**: a future where users express goals like “maximize my stablecoin yields across chains,” and intelligent agents handle the complexity behind the scenes. Achieving this demands a new foundation of trust. Today's offchain automation is often opaque, unverifiable, and reliant on bots or relayers with no guarantees of correctness or security, leaving a dangerous gap between user intent and actual execution.

Newton Protocol closes this gap. Built by Magic and the Magic Newton Foundation, Newton introduces a verifiable automation layer for the onchain economy. Every agent action is executed within user-defined boundaries and cryptographically proven through a combination of trusted execution environments (TEEs) and zero-knowledge proofs (ZKPs). No blind trust required.

By turning automation itself into a provable, trust-minimized primitive, Newton transforms how users, developers, and operators interact with decentralized systems, laying the groundwork for agentic finance, programmable commerce, and a user-driven onchain economy.

Background

Magic was co-founded in May 2018 by University of Waterloo engineers Sean Li and Jaemin Jin. Sean previously co-founded Kitematic, which was acquired by Docker in 2015 and evolved into Docker Desktop - a product used by millions of developers monthly worldwide. Jaemin was an early engineer at Uber, where he played a pivotal role in launching and scaling Uber for Business, a major revenue line alongside Uber Eats. The team is comprised of talents from top technology and crypto firms like Coinbase, OpenSea, Alchemy, Aptos, Affirm, Meta, Apple, and Docker.

Over the past six years, Magic has led the embedded wallet category, onboarding over 50 million wallets. Today, it's trusted by more than 200,000 developers and powers over 2 million monthly active wallets across applications like Polymarket, WalletConnect, Helium, Immutable, Forbes, and Naver. Magic has raised approximately $90 million from investors including PayPal Ventures, Placeholder, DCG, Volt Capital, Polygon, Naval Ravikant, Balaji Srinivasan, and others.

Problem

Broken UX and underutilized capital

Magic's embedded wallets simplified access to web3, but blockchain fragmentation, manual workflows, and steep learning curves continue to deter mainstream adoption. Liquidity remains trapped, and user experience barriers prevent most users from benefiting from DeFi and other onchain systems.

Stablecoins highlight the scale of this inefficiency: of the $230 billion in circulation, only ~40% is actively deployed in DeFi. If this fragmentation persists, over $1 trillion in capital could remain idle by 2030. The problem will only compound as real-world assets (RWAs), a much larger asset class, enter the ecosystem.

Untrusted and risky automation

Existing automation solutions, often driven by Telegram bots, require users to hand over private keys, exposing them to hacks, phishing, and systemic risks. Despite these vulnerabilities, billions in transaction volume highlight strong user demand, but also an urgent need for secure, verifiable alternatives, especially as AI agents introduce new risks like hallucinated actions.

Missing primitives for crypto AI agents

Building multichain dapps remains complex and fragmented, making agents a more preferred form factor for both users and developers than traditional dapps. While automation capabilities are improving, there is still a critical need for secure execution, verifiable automation, and protocol-level trust - primitives essential to bridge top AI developers into crypto finance and commerce.

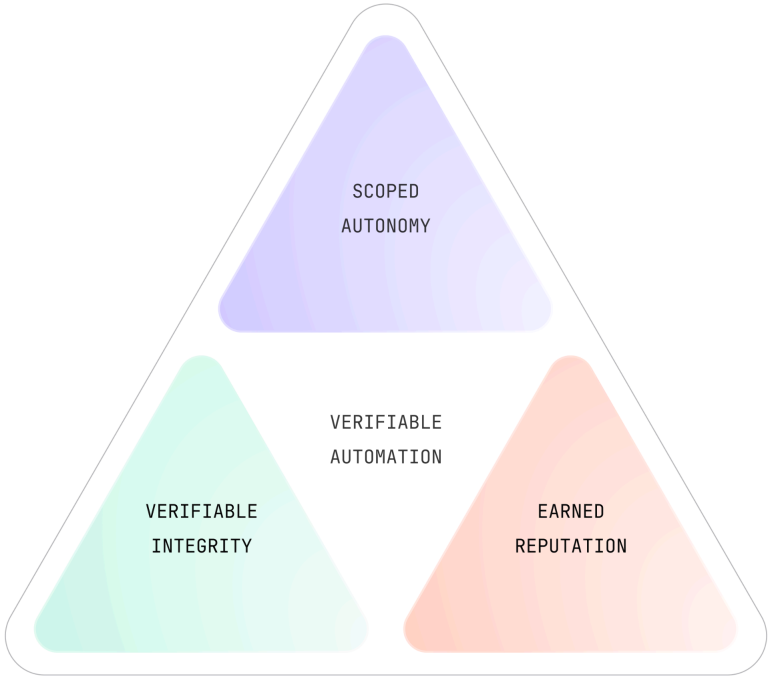
Protocol Features

Newton Protocol powers a new economy where users can discover, select, and delegate onchain actions to agents through an **automation marketplace**. These agents, built by Magic or third-party developers, securely automate financial activity across multiple blockchains. Similar to how Farcaster users interact with the protocol through Warpcast, Newton is paired with a client interface where users can browse available agents, interact through intuitive UIs and natural language prompts, and manage their assets seamlessly across apps, protocols, and chains, much like engaging with an OpenAI operator, but for onchain finance.

No technical expertise is required. These agents automate everything from cross-chain trading to portfolio rebalancing and AI-driven strategies, achieving a step-function improvement in usability across today's fragmented web3 landscape.

Newton Protocol enforces trust through **verifiable automation**, grounded in three core principles:

- **Scoped Autonomy**: Users define the boundaries of agent behavior with zkPermissions, encoding expressive rules that are enforced by design.
- **Verifiable Integrity:** Agents generate cryptographic proofs to prove every action faithfully aligns with the user's declared intent and constraints.
- **Earned Reputation:** Agents build reputation through proven performance, while economic penalties deter misaligned behavior and protect user interests.



Why Verifiable Automation Matters

Traditional smart contracts are limited by deterministic, onchain execution, but many high-value workflows require offchain computation or AI-driven decision-making. Newton Protocol introduces verifiable automation: a system where even offchain actions can be audited and enforced using zero-knowledge proofs (ZKP) and trusted execution environments (TEEs). This gives users the ability to delegate intent without sacrificing control or trust.

Just as Chainlink brought offchain data onchain in a verifiable way, Newton Protocol brings offchain automation and computation onchain, making it possible to build systems where even

AI-powered decisions and cross-chain operations remain accountable, verifiable, and secure by design.

A Pragmatic Approach to Scalability, Security, and Decentralization

Rather than chasing theoretical ideals, Newton Protocol takes a pragmatic approach to infrastructure, prioritizing user experience and security. It leverages:

- **Zero Knowledge Proof (ZKP):** To generate ZKPs that validate the correctness of offchain computations.
- **Trusted Execution Environments (TEE):** For hardware-based trusted execution with remote attestation and economic guarantees via collateral.
- **Minimal, app-specific rollup design:** To optimize performance and security over generalized computation.

Applications and Use Cases

Newton unlocks a new category of high-value financial, commercial, and decentralized automation use cases by enabling developers to publish verifiable agents - secured through ZKP, permissioned through dynamic policies, and executed off-chain in a scalable, trustless manner. These include, but are not limited to:

Long-Term Asset & Strategy Management

- **Cross-Chain Automated Strategies**: Execute complex multi-chain asset strategies, such as recurring token purchases, liquidity provision, or portfolio rebalancing, with verifiable proof that each step follows user-specified timing, pricing, and risk boundaries.
- **Adaptive Yield Aggregation**: Deploy agents that continuously reallocate capital across yield protocols based on real-time APYs, onchain liquidity, and volatility metrics, with cryptographic enforcement of custom stop-losses, caps, and whitelists.
- **Automated Vault & Risk Management**: Monitor portfolio health metrics such as collateralization ratios and risk thresholds, and automatically trigger repayments, deleveraging, or reallocations when breaches occur, minimizing liquidation risks with cryptographic triggers.
- **Verifiable Cross-Chain Arbitrage**: Enable automated arbitrage bots to operate across DEXes and blockchains, provably executing only when user-defined profit margins, fee tolerances, and slippage constraints are met.

Short-Term Trading & Execution Automation

- **Verifiable Copy Trading Networks**: Mirror trades from trusted lead traders in real time, with provable guarantees that mirrored trades stay strictly within user-defined limits (e.g., maximum position size, allowed pairs, and slippage constraints).
- **Limit & Range Order Execution**: Agents monitor real-time price feeds (or multi-source oracles) and automatically execute swaps, mints, or burns when specified price conditions are met, with verifiable enforcement to prevent manipulation.
- **AI-Governed Trading Agents**: Deploy AI-powered decision-making models as verifiable zk-circuited agents that dynamically adjust strategies based on market conditions, with every model inference cryptographically verified onchain.

Commerce & Payments Automation

- **Programmable Stablecoin Commerce**: Automate programmable stablecoin payments, recurring billing, and metered usage services, with agent-enforced compliance checks, spending limits, jurisdiction filters, and onchain auditability.
- **Customizable Personal Automation**: Power “personal agents” that execute bespoke automation workflows for users, such as paying subscriptions, managing cross-chain payroll, maintaining collateralization, or reinvesting rewards, all governed by user-controlled zkPermissions.

DAO & Institutional Automation

- **DAO Treasury Operations Automation**: Enable DAOs to delegate automated, permissioned actions, such as yield optimization, contributor payments, or risk hedging, through verifiable agents governed by multi-party zkPermission policies.
- **Custodial Compliance & Rules Enforcement**: Allow custodians and asset managers to define provable delegation rules, such as spending caps, execution windows, and KYC/AML credential checks, without giving up raw key control.

Protocol Participants

Newton Protocol enables a decentralized market economy for secure, verifiable automation by coordinating four key participant roles:

Developer

Developers build automation services or AI agents that range from simple scripting logic to advanced machine learning models or decision trees. Each service is packaged as a containerized application secured with TEEs and ZKPs. Using the Newton SDKs and zkML frameworks, developers can easily transition their off-chain services into verifiable agents by defining the logic,

constraints, and interfaces through which agents interact with users and blockchain protocols. Newton ensures that not only is the agent's execution protected inside secure hardware, but also that a cryptographic proof of correct execution is generated, enabling users to independently verify the agent's behavior without revealing any proprietary algorithms or internal model details.

Operator

Operators voluntarily participate in the Newton Protocol's automation marketplace, committing to run automation services and earn fees by fulfilling user orders. Within the marketplace, users submit tasks, and operators compete to execute them efficiently and verifiably. The protocol enforces that operators must act not only in a censorship-resistant manner, accepting orders without discrimination, but also prove the correctness of their execution by submitting ZKPs for each request they serve. To balance performance and operational costs, operators may post a stake collateral, allowing them to defer individual proof submissions and instead submit an aggregated batch proof at the end of each protocol epoch. The Newton protocol autonomously enforces accountability, trustlessly slashing operators for dishonest behavior or failure to meet commitments. Over time, each operator builds a public reputation based on verifiable execution history, task success rates, and user feedback, fostering a competitive, decentralized marketplace for high-trust, censorship-resistant automation services.

User

Users actively participate in the automation marketplace as consumers of automation services. They create and submit verifiable orders for operators to fulfill, paying fees to access agents that execute financial tasks on their behalf, such as trading, staking, or monitoring positions. Through intuitive prompts, users define execution permissions, task parameters, and delegate operational control to trusted agents. Future iterations will empower users to review and rate operators, provide feedback, and contribute to the evolving reputation system, enhancing transparency, accountability, and competitiveness across the marketplace.

Validator

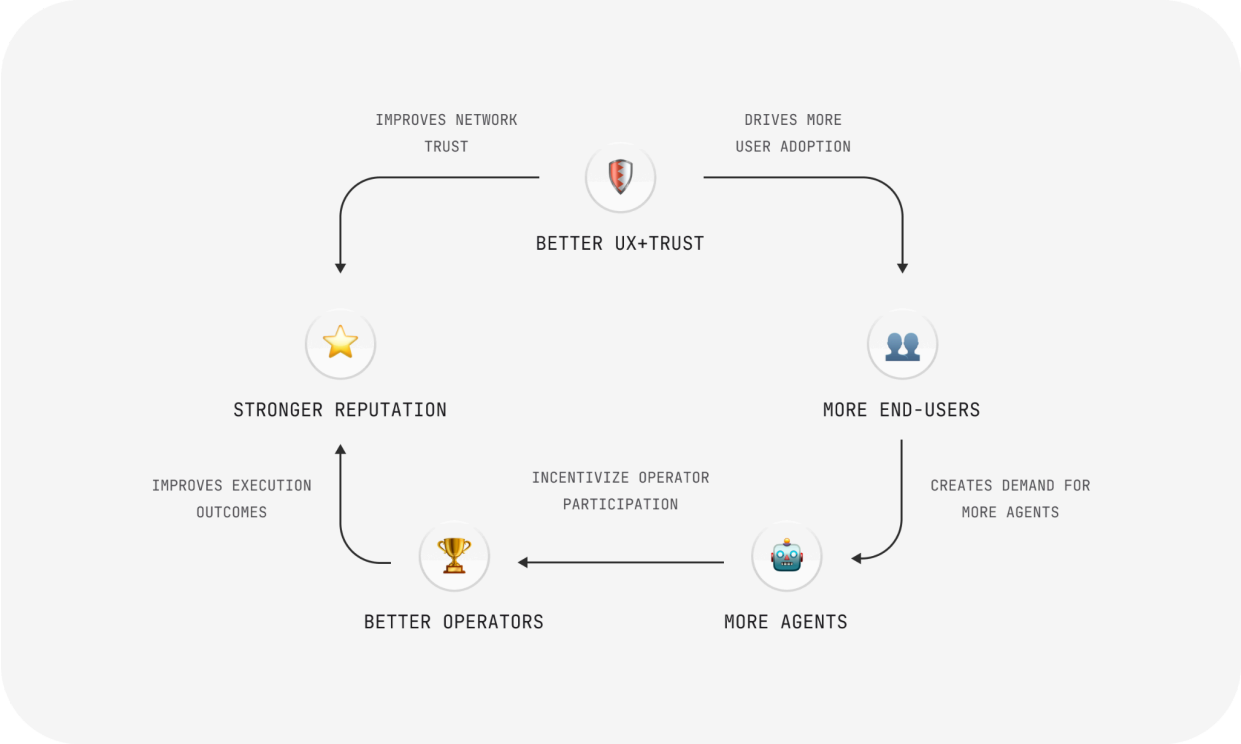
Validators secure the protocol by participating in delegated proof-of-stake (dPoS) consensus. They are responsible for validating each block and generating fast validity proofs to ensure fast finality.

Together, these participants create a flywheel effect that accelerates ecosystem growth:

1. The protocol, paired with a native user client for seamless user-agent interaction, delivers a better experience, significantly improved security, and trust - attracting more users.
2. Growing user demand motivates developers to build more automation agents across diverse verticals and use cases.
3. New agents attract more operators, fostering supply-side competition, greater liveliness, and more efficient service delivery, all contributing to an improved user experience.

4. Higher quality agents and verifiable execution deliver superior outcomes, further amplifying user adoption and expanding the active user base.
5. Reputation and verifiable usage data form the foundation of a merit-based economy, where trusted operators and agents are consistently rewarded, creating a virtuous cycle of trust, adoption, and innovation.

This positive feedback loop turns Newton Protocol into a decentralized market economy for verifiable automation - where trust is built into every transaction, and value is distributed across a permissionless agent economy.



Architecture

Newton Protocol is composed of three integrated components:

Smart Accounts

Users' wallets are secured by smart account standards like ERC-4337 and EIP-7702 to enable cryptographically secured policy-based delegation. These smart accounts allow users to issue advanced granular revocable permissions that authorize agents to act on their behalf, but only within clearly defined logic.

Zero-knowledge proof based permissions remove the need for constant user approval, offloading execution to agents without giving up control. Permissions can be updated or revoked

at any time, offering a secure and programmable delegation model essential for intent-driven automation.

zkPermissions

To ensure agents act within user-defined boundaries, every session key is bound to a zkPermission, a zero-knowledge circuit that encodes offchain automation rules and constraints. These include:

Data-Driven Execution Conditions

Permissions based on oraclized, onchain, or real-world data feeds.

- **Oraclized or real-time data conditions**: e.g. “Only swap if onchain sentiment data indicate bullish market conditions.”
- **Advanced market condition check**: e.g. “Only allow a trade if the 15-minute moving average crosses above the 1-hour moving average (golden cross).”
- **Multi-asset relative price condition**: e.g. “Only allow swapping ETH for BTC if ETH/BTC trading pair is up more than 5% in the last 24 hours.”
- **Liquidity + slippage constraint**: e.g. “Allow swaps only if pool liquidity exceeds $10M and expected slippage is under 0.5%.”
- **Multi-market data feed aggregation**: e.g. “Authorize staking only if ETH gas fees < 20 gwei AND on-chain funding rates are positive AND L2 transaction volume exceeds $100M in the last 24h.”
- **Real-world event synchronization**: e.g. “Allow a trade only after an on-chain oracle signals that CPI (Consumer Price Index) data has been released and indicates inflation < 2%.”

Risk and Price Sensitivity Checks

Permissions based on volatility, deviation, and risk-adjusted signals.

- **Volatility-gated execution**: e.g. “Only permit execution if implied volatility index (e.g., DVOL) drops below 30.”
- **Risk-adjusted price movement**: e.g. “Only buy if the current price is 2 standard deviations below 7-day moving average and RSI < 30.” (oversold conditions)
- **Deviation-triggered limit orders**: e.g. “Trigger sell only if the price deviates more than Â±5% from the oracle median price in the last hour.”

- **Multi-block safety check**: e.g. “Require two consecutive block confirmations that the price condition holds before executing a trade.”

Transaction Volume and Timing Constraints

Permissions based on transaction grouping, execution windows, and scheduling.

- **Grouped transaction limits** - e.g. "Limit total transaction volume to no more than $500 USD worth of assets per execution."
- **Time-weighted execution windows**: e.g. “Allow up to 3 executions per day, but only if TWAP (Time-Weighted Average Price) is within 1% of VWAP (Volume-Weighted Average Price).”
- **Timing restrictions** - e.g. "Only allow execution on Mondays between 8 AM and 12 PM UTC."

Default and Baseline Permissions

Baseline session key permissions enforced for every agent, including common permission types such as spending limits, expiration dates, and token allowlists.

Operators must generate and submit a ZKP (Zero-Knowledge Proof) for every execution attempt, attesting that the logic followed the encoded zkPermission. This ensures even offchain actions are provably aligned with the user's intent, without exposing private data or relying on trust.

Execution Orchestrator

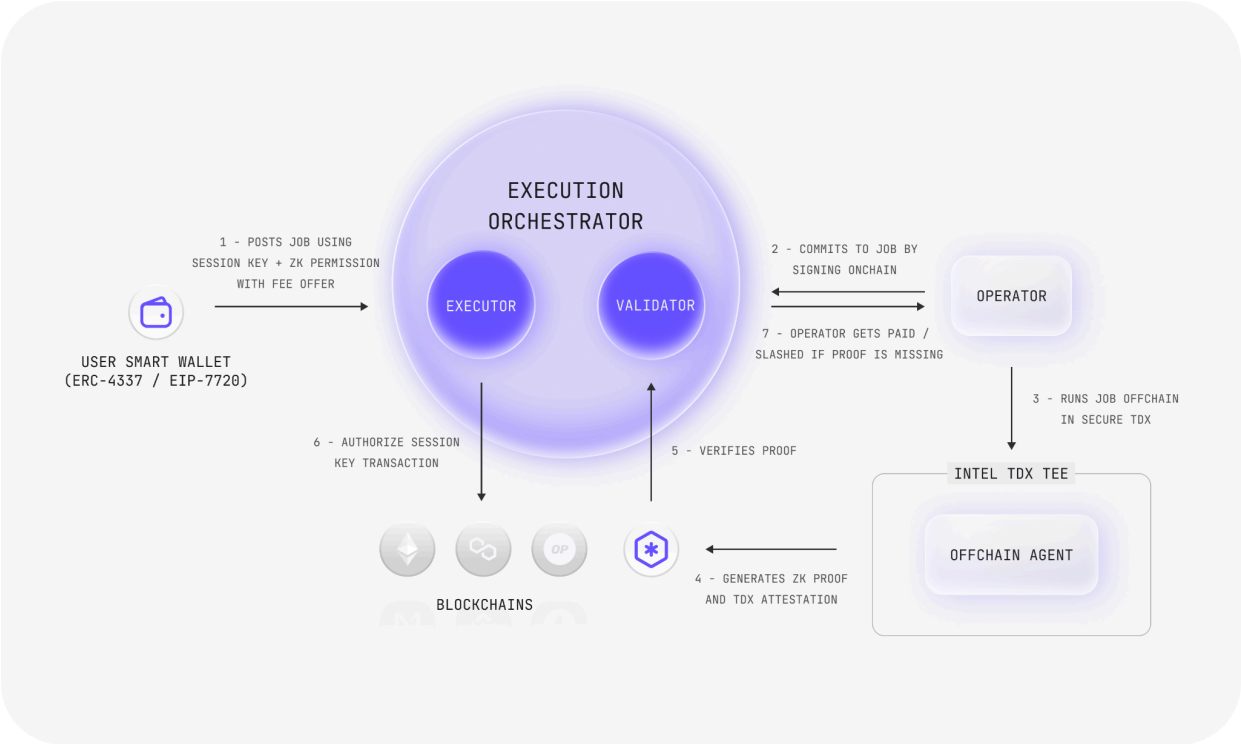
The Execution Orchestrator acts as an orderbook-based marketplace that matches automation orders between users and operators, coordinating how tasks are fulfilled. Users submit automation intents, structured requests for outcomes like “rebalance my portfolio weekly”, along with an associated fee. Over time, this will evolve into an open marketplace, enabling both human-to-agent and agent-to-agent transactions powered by verifiable execution.

- **Executor**: Processes and stores zkPermissions, which encode the user's automation rules and permission constraints.
- **Validator**: Match automation intents between users and operators via a limit orderbook mechanism, and validate updates to the Executor’s stored state across chains. Validators are responsible for verifying that submitted execution proofs are correct before finalizing state transitions.
- **Operators**: Accept tasks posted by users through the Execution Orchestrator, executing the defined automation logic off-chain in a secure and scalable manner. Operators can run services inside Intel TDX enclaves to ensure trusted computation, or operate

zk-circuited machine learning model agents that embed verifiable execution directly into ZKPs. Upon task completion, two cryptographic artifacts are produced and submitted:

- **TEE Attestation**: Proves that the task was executed inside a trusted hardware enclave with the correct code integrity.
- **Zero-Knowledge Proof (ZKP)**: Verifies that the execution strictly adhered to the user's permission constraints or that the ML model was evaluated correctly according to verifiable logic.

Once both the TEE attestation and/or ZKP are verified on-chain, the approved action is executed via a scoped session key, completing the task in a trust-minimized, verifiable, and censorship-resistant manner. This architecture enables high scalability by performing intensive computation off-chain while maintaining on-chain cryptographic verification of correctness and trust.



Developer Tooling & Extensibility

Newton is designed to make it easy for developers to build and extend a vibrant ecosystem of automation agents. It provides powerful SDKs for composing and issuing zkPermissions - flexible, composable rulesets that developers use to define granular constraints on transaction execution. These rules behave like building blocks, enabling developers to stack and assemble custom permission policies that are enforced by the Newton Protocol in a trust-minimized, verifiable manner.

Our framework supplies a trustless rule-making system that allows application developers to define rich, extensible policies using both onchain and offchain data sources, including: historical flow-of-funds analysis, blockchain analytics and state proofs, verifiable credentials and identity attestations, dynamic allowlists and contextual transaction parameters, real-time oracle and sentiment feeds, etc.

In addition to zkPermission tooling, Newton offers SDKs for circuitizing existing off-chain services and AI models, transforming them into verifiable agents. These agents can then be registered on the protocol with clearly defined: input and output data formats, estimated computational costs, associated verification keys for trustless proof verification upon inference completion.

By streamlining the creation of both permission-guarded policies and verifiable off-chain services, Newton empowers developers to rapidly build, deploy, and extend decentralized automation agents with maximum security, scalability, and modularity.

Development Priorities and Roadmap Focus

Newton Protocol is committed to delivering end-to-end trust, security, and usability as our highest priorities. Our initial rollout focuses on core automation capabilities designed for simplicity, safety, and verifiability, ensuring users can interact confidently from the start.

In the early phase, users will be able to engage with simple, native automations, such as recurring token purchases, through a streamlined interface. These automations will be executed inside TEEs and generate cryptographic proofs to ensure each action is verifiably correct. User delegation will be safeguarded by a baseline set of session key permissions, enforcing clear operational boundaries without requiring technical expertise.

As Newton Protocol matures, we will expand from these foundational capabilities to unlock the full potential of verifiable automation. Priorities for future development include:

- Extending the baseline permission model into advanced zkPermission frameworks, enabling users to define highly expressive, programmable execution rules.

- Implementing full marketplace mechanics, supporting both human-to-agent and agent-to-agent task orchestration.

- Delivering developer tooling and SDKs to simplify the creation of both automation strategies and verifiable agent-based services.

This phased approach ensures that Newton prioritizes user safety, trust, and experience from the beginning, while progressively opening the protocol to broader innovation and a decentralized economy of automation agents.

Closing Summary

Newton Protocol establishes a new foundation for the onchain economy, a system where users can safely delegate complex tasks to verifiable agents, with every action cryptographically proven to respect user-defined rules. By combining TEEs and ZKPs, Newton turns automation from a point of failure into a foundation of trust, ensuring security, flexibility, and transparency at scale.

Built for extensibility and composability, Newton empowers users to define dynamic permissions, enables developers to build sophisticated agents and AI-driven services, and rewards operators for reliable, auditable execution. In doing so, Newton addresses core barriers in decentralized systems, fragmentation, security risk, and usability challenges, and creates a practical path for scaling crypto adoption.

As AI and decentralized technologies converge, Newton offers the verifiable infrastructure needed to unlock a new era of programmable finance, commerce, and autonomous agents. It rewrites the trust architecture of web3, turning automation into a foundational primitive for an open, secure, and user-driven onchain economy.