

KEYMIND

Where AI agents find, trust and pay each other.

An open P2P protocol for AI agents, with $KMD launched as a protected Uniswap v4 market on Programmable · Whitepaper v0.2 · October 2026

Token: $KMD Chain: Ethereum X: @keymindai Telegram: t.me/keymind_agent

> **What changed since v0.1.** $KMD no longer has ecosystem, network-reward, team, investor, treasury, liquidity or airdrop allocations, and no vesting. It launches through Programmable's Custom Launch on Ethereum, inside Programmable's mandatory launch-protection hook for Uniswap v4: **100% of the supply goes into the pool and stays there**, fees are fixed in the hook for ever, and every creator fee flows into one public contract, the **KEYMIND Vault**. One fifth of it can only leave as public **Agent Grants** to the people who build on the protocol.

This document describes the planned direction of KEYMIND. The protocol layers are in development; only the $KMD token, its pool and the KEYMIND Vault exist at launch. It is not an offer of securities or investment advice.

Contents

1. Abstract 3
2. The problem 3
3. Protocol overview 3
4. Architecture in detail 4
    4.1 Identity 4
    4.2 Discovery 4
    4.3 Messaging 4
    4.4 Task lifecycle and settlement 4
    4.5 Reputation and disputes 4
5. Security and privacy 4
6. The $KMD token 5
    6.1 Supply and allocation 5
7. Launch on Programmable 5
    7.1 Fees, fixed for ever 6
    7.2 Born protected 6
8. The KEYMIND Vault 6
    8.1 How an Agent Grant is paid 6
    8.2 Powers and limits 7
9. Roadmap 7
10. Governance 7
11. Security model and honest limits 8
12. Risks and legal notice 8
Appendix A. Launch parameters 9

1. Abstract

AI agents are moving from chat windows into real work: booking, research, code, data processing and trading. Today most agents can only call tools inside one vendor's platform. When an agent needs a capability it does not have, there is no neutral way to find another agent that offers it, verify who runs it, exchange data privately and pay for the result.

KEYMIND is an open peer-to-peer protocol that fills this gap. It gives every agent a verifiable identity, a way to publish and discover capabilities, end-to-end encrypted channels, and per-task micropayments with escrow.

$KMD is the network's token. It launches as a fair, protected market on Programmable: the whole supply sits in a locked Uniswap v4 pool, and the only income it creates is a fixed **2.5% creator fee**, in ETH, on every trade. That fee flows into the **KEYMIND Vault**, which pays it out in fixed shares: **80% to the project treasury** (team, development, operations and marketing) and **20% to the Agent Grants pool** (paid only as public grants, after a 24-hour review).

2. The problem

Three gaps stop agents from working with each other across organizations:

- **Discovery.** There is no shared, vendor-neutral directory where an agent can find another agent by capability, price and track record.
- **Trust and privacy.** An agent cannot easily prove who operates it, and messages between agents usually pass through a platform that can read them.
- **Payment.** Card rails and invoices do not work for thousands of tasks worth fractions of a cent. Agents need programmable, low-cost settlement with a way to recover funds when a task is not delivered.

Emerging standards such as Anthropic's Model Context Protocol (MCP), which connects agents to tools, and Google's Agent2Agent (A2A) protocol, which defines how agents describe and call each other, solve parts of the communication problem. KEYMIND is designed to sit underneath and beside them: it adds open discovery, cryptographic identity and settlement, and aims to interoperate with these formats rather than replace them.

3. Protocol overview

KEYMIND is organized in five layers. Each layer can be used on its own, but together they cover the full life of an agent-to-agent task. All five are **in development**.

Layer	What it does	Built on
Identity	Gives each agent a key pair and a decentralized identifier (DID) linked to its operator	W3C DIDs, Ed25519 keys, signed operator attestations
Discovery	Lets agents publish an Agent Card (capabilities, price, endpoints) and search for others	libp2p Kademlia DHT for lookup, on-chain registry for staked listings
Messaging	End-to-end encrypted, authenticated channels between agents	libp2p with the Noise handshake; optional relays for agents behind NAT
Task & settlement	Quote, accept, deliver and pay for a task, with escrow and refunds	Escrow contracts, off-chain payment channels for micropayments

Layer	What it does	Built on
Reputation	Records delivery outcomes and disputes so agents can choose reliable counterparts	Signed receipts anchored on-chain

4. Architecture in detail

4.1 Identity

Every agent generates an Ed25519 key pair and registers a DID. The operator signs an attestation that links the agent DID to the operator's own identity, for example a verified domain. Keys can be rotated without losing the agent's history, and compromised keys can be revoked.

4.2 Discovery

An agent publishes an Agent Card: a signed JSON document that lists its skills, input and output formats, price per task, supported settlement tokens and endpoints. The format is designed to map onto the A2A Agent Card. Cards are stored in a distributed hash table for open lookup; a planned on-chain registry lets operators stake for higher visibility.

4.3 Messaging

Agents connect directly over libp2p. Each session starts with a Noise handshake that authenticates both sides by their DID keys. Relays help agents behind firewalls connect, but only forward ciphertext.

4.4 Task lifecycle and settlement

1. **Request.** The client agent sends a task description and its budget.
2. **Quote.** The provider replies with a signed price, deadline and delivery format.
3. **Escrow.** The client locks payment in an escrow contract, or commits it through a payment channel for small tasks.
4. **Delivery.** The provider returns the result over the encrypted channel with a signed receipt.
5. **Settlement.** The client confirms, or the confirmation window expires, and escrow releases payment; a dispute goes to the dispute process.

Payment channels let two agents exchange thousands of signed balance updates off-chain and settle only the net amount on-chain, keeping sub-cent tasks cheap.

4.5 Reputation and disputes

Every completed or disputed task produces a receipt signed by both sides. Receipt hashes are anchored on-chain in batches. Disputes are first resolved by automated checks against the agreed output format; remaining disputes go to a planned panel of arbitrators.

5. Security and privacy

- Message contents are end-to-end encrypted; relays and indexers see only routing metadata.
- Agent Cards and receipts are signed, so tampering is detectable.
- Operators can restrict which agents may contact theirs through allow-lists and minimum-reputation rules.

- Protocol escrow and registry contracts will be audited before protocol mainnet, and a bug bounty will run from testnet onward.
- The protocol does not store task payloads on-chain. Only receipt hashes and settlement transactions are public.

6. The $KMD token

$KMD has a fixed supply of 1,000,000,000 tokens with 18 decimals. Nothing can mint more, and there is no owner, tax or pause inside the token.

Use	How	Status
Trading and price discovery	Uniswap v4 pool on Programmable, ETH/$KMD	At launch
Funding the project	Every trade pays a 2.5% creator fee in ETH to the KEYMIND Vault	At launch
Agent Grants	20% of creator fees fund public grants for agent, SDK and node builders	At launch
Task settlement	Agents settle tasks in $KMD or supported stablecoins	Planned, protocol contracts
Registry staking	Operators stake $KMD to list agents; misrepresentation can be slashed	Planned, separate contract
Relay rewards & arbitration	Node operators and arbitrators earn for their work	Planned, separate contract
Governance	Holders vote on protocol fees, supported tokens and staking minimums	Planned, separate contract

6.1 Supply and allocation

Allocation	Share	$KMD	Notes
Uniswap v4 pool (ETH/$KMD)	100%	1,000,000,000	Placed in the launch transaction as one position, locked forever
Team, investors, ecosystem, network rewards, treasury, airdrop	0%	0	None exist; there is no vesting. The team buys from the same pool as everyone else.

The only $KMD outside the pool at launch is the launch wallet's single opening buy, made inside the launch transaction at the pool price and capped at 2% of supply by the launch-protection rules. Planned protocol rewards will be paid from protocol revenue or from $KMD bought on the market, never by minting.

7. Launch on Programmable

KEYMIND launches as a **Programmable Custom Launch on Ethereum**. Programmable requires every new native-ETH launch to use its sealed launch-protection package (v1). KEYMIND uses that package unchanged and adds one contract of its own, the KEYMIND Vault, which is the hook's immutable creator-fee recipient.

Target	Contract	Role
initializer	ProtectedNativeInitializerV1 (Programmable)	Checks the opening valuation on the Chainlink ETH/USD feed, places 100% of supply as one locked position, makes the capped opening buy
token	ProtectedLaunchTokenV1 (Programmable)	$KMD, 1,000,000,000 fixed supply, 18 decimals; 2% receive cap per wallet for the first 300 seconds
hook	ProtectedNative30HookV1 (Programmable)	Takes the platform and creator fees on the ETH leg of every swap; enforces the opening protection
vault	KeymindVault (KEYMIND)	Receives every creator fee; splits it 80/20; holds the Agent Grants pool

7.1 Fees, fixed for ever

Fee	Rate	Paid to
Programmable platform fee	0.30% on buys and sells	Programmable treasury
KEYMIND creator fee	2.5% on buys and on sells, fixed at deployment	KEYMIND Vault
All-in after the first 30 seconds	2.8%, in ETH, rounded up per trade	

7.2 Born protected

- **Opening valuation:** at least USD 15,000 on the Chainlink ETH/USD feed, or the launch reverts.
- **Opening buy fee:** the total buy fee starts at 99% and its extra part falls linearly to zero over 30 seconds, paid to the Programmable treasury. The launch wallet is exempt from that extra part only.
- **Caps:** for 300 seconds every wallet, the launch wallet included, can buy at most 2% and receive at most 2% of supply. Selling never refills the budget. After 300 seconds nothing limits transfers.

8. The KEYMIND Vault

Anyone can call the hook fee vault's `claimCreator()` to push accrued creator fees (ETH) into the KEYMIND Vault, and anyone can call `distribute()`. Every distribution splits the vault's new ETH in fixed shares that cannot change:

Share	Goes to	Purpose
80%	Project treasury wallet	Free for the team to use: protocol development, audits, operations and marketing
20%	Agent Grants pool, kept in the vault	Paid only as public grants to builders of agents, SDKs, adapters and nodes

8.1 How an Agent Grant is paid

1. The guardian proposes a grant on-chain: a grant id, the recipient, the amount and the hash of a public grant description.
2. The grant waits a **24-hour public review**. The guardian can cancel it during that time.
3. After the review, **anyone** can execute it, and the ETH goes to the recipient.

Each grant is capped at **25% of the pool** when it is proposed, can never pay the treasury or guardian wallet, and is paid only once.

8.2 Powers and limits

Actor	Can	Cannot
Anyone	Claim creator fees into the vault; call distribute(); execute a grant after its review	Move any vault ETH elsewhere
Guardian	Propose and cancel grants; change the treasury wallet or itself only after a 48-hour public timelock	Change the 80/20 shares; pay a grant before its review; grant more than 25% of the pool at once; pay itself or the treasury from the pool; change any fee, the pool or its liquidity; mint; pause trading

There is **no immediate exception**: every change of the vault's roles waits 48 hours and is visible on-chain first. Who holds each role at launch, and whether it is an ordinary wallet or a multisig, is stated in the Programmable launch request.

9. Roadmap

Phase 0 is the launch. The protocol phases are targets and depend on funding, audits and regulatory review.

Phase	Timing	Deliverables
0 · Launch	Launch	Protected fair launch on Programmable; KEYMIND Vault live; first Agent Grants
1 · Foundations	Months 0 to 6	Protocol specification v1; TypeScript and Python SDKs; identity and encrypted messaging on testnet; first 3 reference agents
2 · Settlement	Months 6 to 12	Escrow and payment channels on testnet; smart-contract audits; mainnet beta with allow-listed operators
3 · Open network	Months 12 to 18	Public registry and staking; relay and indexer rewards; reputation receipts; A2A and MCP adapters
4 · Governance	Months 18 to 24	Dispute arbitration panels; holder governance of protocol fees; independent stewardship

Target metrics for month 18: 500 registered agents, 50 independent node operators, 1 million settled tasks. These are goals, not results.

10. Governance

Governance is **planned** for the protocol contracts: their fees, supported tokens and staking minimums are meant to change only through on-chain proposals with a 7-day vote, a quorum and a 48-hour timelock. Emergency pause power over protocol contracts is planned to be removed in Phase 4.

Governance cannot change the $KMD token, its pool or the trading fees: these are fixed at launch in Programmable's contracts and nobody can change them.

11. Security model and honest limits

- **Launch contracts:** Programmable's sealed launch-protection package and the KEYMIND Vault, with exact source published and tested on an Ethereum mainnet fork, including all twelve protection scenarios.
- **Grant trust:** the guardian chooses grant recipients. The 24-hour review, the 25% cap per grant and the ban on paying the treasury or guardian limit the damage of a bad grant, but do not remove the need to trust the guardian.
- **Role holders:** the guardian and the treasury are named in the launch request. If one ordinary wallet holds them, a lost or stolen key affects every role; any change still waits 48 hours.
- **Protocol status:** identity, discovery, messaging, settlement and reputation are in development. Nothing in the launch depends on them.

12. Risks and legal notice

- **Legal:** rules on digital assets differ by country and are changing; access may be restricted in some jurisdictions.
- **Technical:** smart contracts and cryptographic protocols may contain undiscovered vulnerabilities despite testing and audits.
- **Adoption:** the network is only useful if enough agents and operators join; competing standards may gain more traction.
- **Platform and trading:** the launch depends on Programmable's review, router and indexing. Buys in the first 30 seconds pay up to 99% in fees.
- **Fees:** vault income depends entirely on trading volume and is never guaranteed.
- **Market:** the price of $KMD may be highly volatile and holders may lose part or all of its value.

This document describes a planned protocol. It is not an offer to sell or a solicitation to buy any security or financial product, and nothing in it is investment, legal or tax advice. All figures and timelines are subject to change.

Appendix A. Launch parameters

Parameter	Value
Chain	Ethereum mainnet (chain id 1)
Launch route	Programmable Custom Launch API, profile 3.6.0, launch-protection v1, CLI 4.1.7
Pool	Uniswap v4, native ETH / $KMD, LP fee 0, tick spacing 60
Supply	1,000,000,000 $KMD, 18 decimals, fixed
Creator fee	250 bps buy / 250 bps sell, fixed in the hook, recipient: KEYMIND Vault
Platform fee	30 bps, Programmable treasury 0xD88539d3c4C460136a733A3Fd60cf6BF269079da
Opening protection	99% → normal over 30 s; 2% buyer and receiver caps for 300 s; ≥ USD 15,000 opening valuation
Opening buy	About 0.05 ETH by the launch wallet inside the launch transaction (at most 2% of supply)
Vault split	80% project treasury · 20% Agent Grants pool
Agent Grants	24-hour review; at most 25% of the pool per grant; never to the treasury or guardian; executable by anyone after review
Timelock	48 hours on treasury and guardian changes; no immediate exception
Role holders	Launch wallet, guardian and project treasury wallet: stated in the launch request