jmdt

JMDT Layer 2 Blockchain

The Technical White Paper

Abstract

> **JMDT** is the Truth Layer for all information - Restoring authenticity in digital infrastructure by privately verifying humans, and decentralising their data.

Jupiter Meta Data Token Chain (JMDT) is a modular, Ethereum-based **Layer 2 (L2) blockchain protocol** designed to address the scalability, privacy, and compliance limitations of traditional blockchain systems. Built with **Zero-Knowledge Proofs (ZKPs)**, **Decentralized Identity (DID)**, and our own proprietary **Asynchronous Validation Consensus (AVC)**, JMDT delivers a high-performance, privacy-preserving infrastructure tailored for both decentralized applications and enterprise-grade solutions.

At the core of JMDT is its **Layer 3 (L3) Enterprise Privacy Mesh**, a **Directed Acyclic Graph (DAG)-based architecture** that allows enterprises to process **private, high-throughput transactions** locally. These state transitions are **ZK-validated and cryptographically committed to L2**, ensuring global auditability without exposing raw data. This architecture provides organizations with full **data sovereignty**, while retaining the **integrity and verifiability of blockchain finality**.

JMDT's **dynamic rollup architecture** anchors L2 state changes to **Ethereum Layer 1**, optimizing for gas fees, block time, and settlement efficiency. All zk-proofs are generated via **Rust-based circuits** executed in the **RISC Zero zkVM**, offering reproducibility, auditability, and verifiable computation with STARK-level guarantees.

The protocol's **AVC consensus mechanism** combines:

- **VRF-based Buddy Node selection** with adaptive weights from Seed Nodes,
- **zk-proof integration** to strengthen validation integrity,
- a **Gossip protocol** for efficient propagation of blocks, votes, and metadata, and
- **CRDT-based conflict resolution** to guarantee convergence even under partitions.

This hybrid design achieves **scalable, asynchronous, and secure block finality** while avoiding the high communication and energy costs of traditional consensus models.

With native support for **DID-based Zero knowledge authentication**, **privacy-preserving queries**, **immutable storage via immudb**, and **tokenized governance using JMDT Utility Token**, JMDT is engineered to power next-generation decentralized applications and ecosystems.

By uniting **verifiability, modular enterprise layers, and cryptographic privacy**, Jupiter Meta Data Token stands as a forward-compatible, high-assurance blockchain protocol built for the decentralized future.

---

1. Introduction

Jupiter Meta Data Token (JMDT) is an **Ethereum-based Layer 2 (L2) blockchain protocol** built to deliver **scalable, privacy-preserving, and verifiably secure computation** for both enterprise and decentralized applications. By integrating **Zero-Knowledge Proofs (ZKPs)**, **Decentralized Identity (DID)**, and **AVC consensus model**, JMDT offers a **next-generation infrastructure** designed for **data sovereignty**, **modular deployment**, and **interoperability across Web3 ecosystems**.

JMDT's architecture spans across three layers:

- **L3 DAG Layer** for enterprise-grade, high-throughput data operations.
- **L2 Rollup Layer** for privacy-preserving execution and zk-based consensus.
- **L1 Finality Layer** for Ethereum anchoring and global settlement.

1.1 Why Jupiter Meta Data Token (JMDT)?

Conventional blockchains struggle with performance, privacy, and real-world compliance. JMDT addresses these systemic limitations through a tightly integrated architecture:

- **Zero-Knowledge Proofs (ZKPs)**: Enabling human Identity verification, private transaction validation, and 100% data privacy - through trustless infrastructure that does not reveal any information beyond the fact that the statement is true.
- **Decentralized Identity (DID)**: W3C-compliant, privacy-preserving authentication framework that verifies PII with on-chain records without accessing any PII whatsoever.
- **Layer 2 Scaling**: High-throughput zkRollup engine built using RISC Zero zkVM for verifiable Rust-based execution. Designed for Faster settlements and the ability to process large scale concurrent transactions in seconds.
- **AVC Consensus**:
  - **Scalable and secure block finality** through quorum-based buddy voting and asynchronous global tally
  - **Energy-efficient operation** by eliminating mining and token-weighted privilege
  - **Low-latency, fault-tolerant propagation** of consensus-critical data, and
  - **Seamless compatibility** with DAG-based Layer-3 enterprise extensions.

  AVC is optimized for use cases requiring **privacy, auditability, and reliability** — including digital identity verification, decentralized finance applications and cross-organizational data collaboration.

- **Ethereum Compatibility**: L2 rollups seamlessly settle on Ethereum L1 using verifiable zk-proofs.
- **Immutable Ledger via immudb**: Ensures tamper-proof audit trails across all data and DAG activity.
- **Privacy-Preserving Queries**: Decentralized Identity based access with zk-proofed data capture, enabling GDPR-compliant data sharing.
- **Enterprise Mesh Model**: Supports interoperable DAGs per organization, enabling modular private ecosystems with shared L2 accountability.

1.2 Core Objectives

JMDT is the Truth Layer for all information on the internet - a single source truth for all of humanities data.

We are built to support completely Decentralised Identity verification, on-chain data validation and proof-based data sharing. This is how do it;

Objective	Description
Scalability	Achieves >2,000+ TPS via zkRollups and DAG batching
Privacy	End-to-end privacy via ZKPs and DID-based access controls
Security	zkVM-based execution, AVC consensus, and stake-based accountability
Interoperability	Fully compatible with Ethereum, and Web3 SDKs
Efficiency	Bloom Filters reduce propagation redundancy; zkRollups optimize gas fees for L1 finality
Governance	JMDT token enables on-chain DAO governance and validator staking

1.3 A Future-Proof Blockchain Infrastructure

JMDT combines the rigor of zero-knowledge cryptography with the practical needs of enterprises and decentralized communities. With **zkRollups for privacy**, **DAGs for performance**, **DIDs for access control**, and **Ethereum for settlement**, JMDT offers a **layered, composable, and audit-ready blockchain foundation** for:

- Decentralized finance (DeFi)
- Web3 Data Marketplaces
- Identity management platforms
- AI-driven analytics pipelines
- Public sector and compliance-first applications

---

2. Vision and Mission

> “Building the truth layer for human intelligence—where verified humans own their data, enterprises access verified insights, and privacy is absolute.”

We are enabling a single source of truth for all human information.

A **next-gen blockchain infrastructure** where **decentralized identities, zero-knowledge proofs**, and **verifiable computation allow us to redefine how data is owned, controlled and monetized** without compromising compliance **across digital platforms.**

-

Privacy-first. Truth-verified. Human-owned. Decentralized by design.

Currently, JMDT’s Web3 ecosystem has cryptographically verified over 20 million humans without exposing their identity. Allowing them to monetize authentic data that they actually own, and provide insights that previously enriched everyone but them.

Enterprises are inturn given access to this decentralised human intelligence with a verifiable chain of custody—from verified human and immutable blockchain storage to predictive insights that result in massive monetary value.

---

3. Technology Stack

1. **Zero-Knowledge Proofs (ZKPs)**

   JMDT leverages **Zero-Knowledge Proofs (ZKPs)** to enable **privacy-preserving identity verifications, authentication, transactions, and computations** without revealing sensitive information. ZKPs allow users to verify transaction validity without exposing underlying data, ensuring security and confidentiality across financial, identity, and enterprise applications. JMDT supports both **zk-SNARKs (Succinct Non-Interactive Argument of Knowledge)** and **zk-STARKs (Scalable Transparent Argument of Knowledge)** for efficiency and quantum resistance.

2. **Decentralized Identity (DID)**

   JMDT incorporates **W3C-standardized Decentralized Identifiers (DIDs)** to provide users with **self-sovereign identities** that enable authentication across **dApps, DeFi, and enterprise services**. **DID-based authentication** ensures **secure, private access** while preventing unauthorized data exposure. This eliminates reliance on centralized identity providers and enhances compliance with **GDPR, HIPAA, and data sovereignty regulations**.

3. **Layer 2 Scaling**

   JMDT uses **zk-rollups** and **immutable ledger-based state synchronization** to significantly enhance transaction throughput and efficiency. By bundling multiple transactions into a single proof before submitting them to **Ethereum (L1)**, JMDT achieves **lower gas fees, reduced congestion, and enhanced finality**, making it ideal for **DeFi, gaming, and high-frequency enterprise transactions**.

4. **AVC Consensus Algorithm**
   The **AVC Consensus Algorithm** is a purpose-built framework for decentralized, high-

throughput blockchain systems. It ensures **secure participation**, **adaptive fairness**, and **scalability** by combining verifiable randomness, zero-knowledge validation, and fault-tolerant data structures.

AVC integrates:

1. **VRF-based Buddy Node Selection** – A deterministic verifiable random function, seeded and weighted by Seed Node feedback, selects a unpredictable randomized buddy set each round. This ensures fairness, geo-diversity, and resistance to Sybil or cartel attacks.
2. **Zero-Knowledge Proof Integration** – zk-proofs strengthen block validation, allowing lightweight and privacy-preserving verification of transactions and state transitions without revealing sensitive data.
3. **Gossip Protocol** – A decentralized backbone for disseminating transactions, buddy votes, and zk-proofs efficiently across the JMDT Decentralized Node (JMDN) network.
4. **Bloom Filters** – Used to prevent duplicate or replayed transactions during gossip and validation, reducing bandwidth overhead.
5. **Conflict-Free Replicated Data Types (CRDT)** – Enable eventual consistency in buddy and network states (e.g., votes, digests, mempool views) without central coordination, ensuring convergence even under partitions.
6. **immudb Append-Only Ledger** – Provides tamper-proof storage of finalized blocks, while malicious or invalid data is flagged and stored separately for auditability.

The consensus design delivers:

- **Scalable and secure block finality** through quorum-based buddy voting and asynchronous global tally,
- **Energy-efficient operation** by eliminating mining and token-weighted privilege,
- **Low-latency, fault-tolerant propagation** of consensus-critical data, and
- **Seamless compatibility** with DAG-based Layer-3 enterprise extensions.

AVC is optimized for use cases requiring **privacy, auditability, and reliability** — including digital identity, decentralized finance and cross-organizational data collaboration.

5. **Ethereum Interoperability**

   JMDT is **fully compatible with Ethereum smart contracts**, enabling **seamless integration with existing DeFi applications, NFT marketplaces, DAOs, and enterprise solutions**. Developers can build and deploy Solidity-based applications on JMDT while benefiting from **L2 scalability, lower gas costs, and enhanced security**.

6. **Efficient L1 Commitments**

   To ensure security and transparency, **all Layer 2 transactions on JMDT are committed to Ethereum (L1)** based on **dynamic optimization of gas fees, block times, and network congestion**. This **adaptive commitment strategy** balances cost-efficiency with **on-chain finality**, ensuring robust security without excessive transaction fees.

7. **Immutable Storage**

JMDT leverages **immutability-focused storage mechanisms, including immudb**, to ensure **tamper-proof, cryptographically verifiable data recording**. This is critical for **auditable enterprise applications, supply chain tracking, and regulatory compliance**.

8. **Privacy-Preserving Queries**

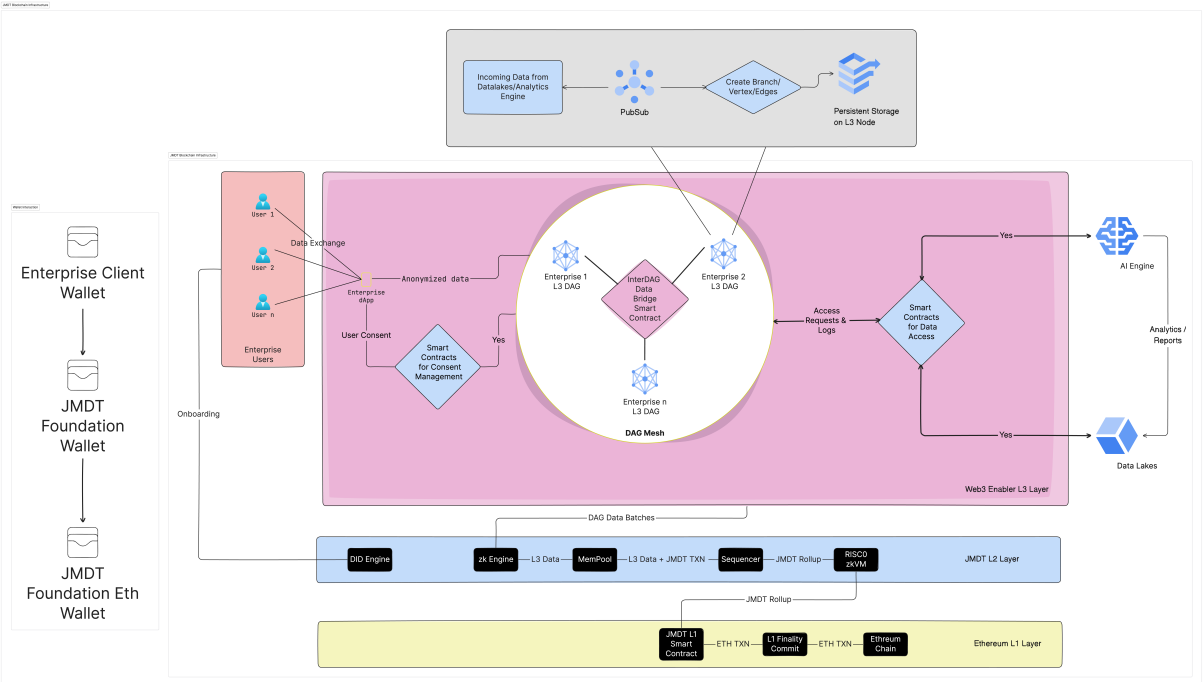
   JMDT introduces a **DID-based querying mechanism** that enables **privacy-preserving access to data**. This ensures that:

   1. Users and enterprises can query blockchain data **without exposing sensitive details**.
   2. **Zero-Knowledge cryptography** prevents unauthorized access while allowing verifiable computations.
   3. **Federated and differential privacy techniques** ensure compliance with data protection laws while enabling valuable insights.
   4. With these advanced privacy-preserving mechanisms, JMDT fosters a **secure, scalable, and interoperable blockchain ecosystem** for **financial services, enterprise data sharing, and next-generation decentralized applications**.

---

4. Jupiter Meta Data Token Architecture

> Jupiter Meta Data Token (JMDT) is a multi-layered blockchain infrastructure designed to support **scalable, privacy-preserving,** and **enterprise-grade data processing**, from user consent to Ethereum finality. The architecture is divided into three core layers—L3 (enterprise DAGs), L2 (zkRollup + consensus), and L1 (Ethereum anchoring).



4.1 Layer 3 – Enterprise DAG & Consent Infrastructure

- **Enterprise DAG Nodes**: Each organization or dApp (e.g., SuperJ, Hercules, gaming, finance, healthcare) operates a private DAG node for high-throughput, localized operations. DAG nodes are synchronized internally using **RAFT consensus**, while state transitions are periodically committed to L2.
- **Smart Contracts for Consent & Access**:
  - Govern user onboarding, consent capture, and access rights for data exchange.
  - Only anonymized data is published to DAGs post-consent.
- **InterDAG Bridge**:
  - Facilitates cross-enterprise collaboration using shared smart contracts.
  - Enables access requests, logging, and secure off-chain queries.
- **Pub/Sub DAG Mesh**:
  - Supports streaming analytics and real-time data ingestion via Pub/Sub architecture.
  - Every node builds **vertices and branches**, logged with **persistent storage**, and synchronized using **RAFT**.

4.2 Layer 2 – ZK-Enabled State and Rollup Layer

- **zk Engine (SNARK + STARK)**: Verifies DAG transactions and batch validity using zero-knowledge proofs.
- **DID Engine**: Provides W3C-compliant Decentralized Identity, allowing private yet verifiable authentication.
- **AVC Consensus Mechanism**:
  - Combines **VRF-based buddy selection**, **asynchronous quorum validation**, **zk-proof enhanced verification**, **gossip-based propagation**, and **CRDT-driven conflict resolution**.
  - Ensures **fast and scalable finality**, **tamper-proof auditability via immudb**, and **high availability** across a decentralized peer network without permanent validator roles.
- **Sequencer and MemPool**:
  - Aggregates DAG state transitions into zk-proofs using a modular zkVM.
- **RISC Zero zkVM**:
  - Executes rollup verification logic as a deterministic, auditable guest program.
  - Outputs STARK proofs submitted to Ethereum for L1 anchoring.

4.3 Layer 1 – Ethereum Settlement Layer

- **JMDT Smart Contracts**:
  - Receive L2 commitments, verify zkVM proofs, and finalize state on Ethereum.
- **L1 Finality Commit**:
  - Ensures integrity and auditability of enterprise and validator activity.
  - Supports transparent public settlement while preserving L3 data privacy.

4.4 Key Properties

- **Privacy-Preserving**: We use Decentralised Identity verification, Zero-knowledge proof data validation and data sharing - ensuring personal data is verified without even being shared.
- **Enterprise-First**: The L3 DAG Mesh Network allow private computation, internal tokenization, and industry-specific logic.
- **Verifiable Rollups**: RISC Zero zkRollups provide immutable, trust-minimized proofs of all activity.

---

5. Ethereum as the Foundation Layer

Ethereum serves as the foundational Layer 1 blockchain for JMDT due to its robust smart contract capabilities, security, and decentralized ecosystem. The Ethereum network provides the necessary infrastructure for executing smart contracts, ensuring trustless interactions, and facilitating scalable Layer 2 solutions through rollups.

Key Benefits of Building on Ethereum:

- **Security:** Ethereum's battle-tested consensus mechanisms and extensive developer community provide a highly secure environment for decentralized applications.
- **Scalability with Layer 2:** Ethereum’s ecosystem supports zk-rollups and other Layer 2 solutions, enabling high transaction throughput with reduced gas costs.
- **Interoperability:** Ethereum's widespread adoption allows JMDT to interact seamlessly with other Web3 protocols, DeFi platforms, and enterprise blockchain applications.
- **Smart Contract Functionality:** Ethereum’s Solidity-based smart contracts enable automation and secure execution of decentralized agreements.
- **Sustainability:** With Ethereum’s transition to Proof-of-Stake (PoS), energy consumption has been significantly reduced, aligning with JMDT’s sustainability goals.

By leveraging Ethereum as its base layer, JMDT ensures that its Layer 2 solution inherits Ethereum's decentralization, security, and interoperability while introducing efficiency and privacy enhancements through Zero-Knowledge Proofs and DID-based identity management.

6. JMDT Decentralized Identity (DID)

JMDT implements **W3C-standardized Decentralized Identifiers (DIDs)** to provide users with secure, self-sovereign identities. Users with a **JMDT DID** can seamlessly authenticate into decentralized applications (dApps) built on the JMDT Layer without having to share any personally identifiable information (PII) whatsoever.

This DID-based ZKP authentication ensures **privacy, security, and user control** over identity verification.

6.1 How Users Create a JMDT DID

- **Onboarding Through Services:** Users can create a **JMDT DID** by registering through **a service provider, dApp, or enterprise system** that integrates JMDT's identity framework.
- **Self-Generated DID:** Advanced users and enterprises can **self-generate** a DID using **cryptographic key pairs as mentioned in JMDT Documentation** and register them on the JMDT network.
- **Enterprise-Managed DIDs:** Organizations can issue **enterprise-controlled DIDs** for employees, partners, or customers, enabling **private, ZK verifiable access** to enterprise services.

6.2 Benefits of JMDT DID-Based Authentication

- **PII-Protected Access:** Services utilizing **JMDT DID** can provide **secure, decentralized Identity authentication** without accessing personal data using our on-device Zero Knowledge computation.
- **Interoperability with Web3 & Enterprises:** Users can seamlessly access partnered **blockchain decentralized applications, and enterprise systems** while maintaining full privacy and security.
- **Self-Sovereign Identity:** Users have **full control over their credentials and the data**, reducing reliance on centralized identity providers.
- **Enhanced Security: Zero-Knowledge cryptographic techniques** prevent unauthorized access to user data while allowing verifiable interactions.

7. Zero-Knowledge Proofs: SNARKs & STARKs

Zero-Knowledge Proofs (ZKPs) are cryptographic techniques that allow one party to prove knowledge of certain information without revealing the information itself. ZKPs ensure privacy

and security while maintaining verifiability, making them a cornerstone of blockchain scalability and data integrity.

JMDT leverages two primary types of ZK proofs:

- **Succinct Non-Interactive Argument of Knowledge (SNARKs):** SNARKs provide short and efficient proofs with minimal computational overhead. They are widely used in privacy-preserving transactions, enabling verifiable computations without exposing sensitive data. However, SNARKs require a trusted setup, which can introduce security concerns if compromised.
- **Scalable Transparent Argument of Knowledge (STARKs):** STARKs offer enhanced scalability and transparency by removing the need for a trusted setup. They use hash-based cryptography, making them quantum-resistant and highly secure. STARKs are particularly useful for handling large-scale computations in a decentralized and trustless manner.

By integrating both SNARKs and STARKs, JMDT ensures a balance between efficiency, security, and scalability. This enables secure, private transactions while maintaining computational efficiency and eliminating trust dependencies.

7.1 ZK Circuit Implementation and RISC Zero Integration

Jupiter Meta Data Token employs a dual-layer ZK architecture, leveraging custom zk-circuits written in **Rust** and executed within the **RISC Zero VM** for high-assurance proofs. This architecture ensures both transaction-level privacy and verifiable state transitions from Layer 3 and Layer 2 to Ethereum Layer 1.

Circuit Design in Rust

All ZK circuits in JMDT are authored in **Rust** using RISC Zero's toolchain. This allows developers to write normal Rust programs, which are then compiled into **zkVM-executable guest binaries**, preserving auditability, performance, and modular design.

Circuit Types:

1. **Transaction Validation Circuits (L2)**:
    - Written in Rust as deterministic logic.
    - Validate balances, DID authentication, signature correctness, and state transitions within L2.
    - Output a commitment hash that is consumed by the zk-rollup aggregator.
2. **Aggregation & State Transition Circuits (L2 → L1)**:
    - Aggregate multiple rollup blocks or DAG state transitions.
    - Execute recursive verification logic or merkle root reconciliation.
    - Generate a succinct STARK proof using RISC Zero's zkVM.

RISC Zero zkVM for Ethereum Finality

JMDT utilizes the **RISC Zero zkVM** to provide **verifiable computation** for all Ethereum L1 commitments. It acts as a zk-powered virtual CPU that proves execution of arbitrary Rust code.

- **Workflow**:
  1. DAG state updates and rollup transitions are executed in a zkVM guest program.
  2. The zkVM produces a **STARK proof** and a journal (public output).
  3. This proof is submitted to a JMDT smart contract on Ethereum for validation.
- **Advantages**:
  - No need to define manual R1CS or arithmetic circuits.
  - Enables fast iteration, modular upgrade paths, and readable cryptographic logic.
  - Guarantees integrity and reproducibility of enterprise workflows and DAG transitions.

Security and Upgradeability

By writing ZK circuits in Rust and compiling them for the zkVM:

- We achieve full type safety, memory safety, and logic reuse.
- Circuits are easy to audit and version-control.
- ZK proofs are consistent across validator nodes and enterprise DAGs, with deterministic reproducibility.

---

8. Consensus Mechanism: Asynchronous Validation Consensus (AVC) with zkProofs and Peer Quorum

To achieve fast, tamper-resistant, and scalable block finality in the Jupiter Meta Data Token (JMDT) Layer-2 blockchain, we implement **Asynchronous Validation Consensus (AVC)** — a quorum-based, non-synchronous consensus model designed for decentralized environments and strengthened with zk-proof guarantees.

Unlike producer-driven consensus, AVC relies on **collective validation by randomized buddy sets**, ensuring fairness, resilience, and transparency. Nodes independently verify proposed blocks, align on Merkle digests, and reach quorum without relying on fixed committees or synchronous rounds.

8.1 Key Components

Asynchronous Validation

- When the Sequencer proposes a block, **all JMDN nodes immediately enter validation mode**.

- This event-driven, non-blocking design allows **parallel validation** across the network without reliance on global timing or designated leaders.

Buddy Set Quorum

- A **VRF-based Buddy Node Selection Algorithm**, weighted by Seed Node feedback, deterministically selects a buddy set of *k* nodes each round.
- Buddies perform:
  - Signature and DID checks,
  - Balance sufficiency and ownership validation,
  - zk-proof verification for batch integrity and privacy.
- A block is accepted when **≥ q_buddy = ⌈2k/3⌉ buddies** sign and broadcast a **BuddyVote(digest)**. This threshold guarantees overlap between quorums and resilience against Byzantine behavior.

zk-Proof Integrity

- Each block must include a **zk-proof** (generated off-chain by a zkVM or prover service).
- zk-proofs validate batched execution correctness and are verified independently by all buddies.
- This ensures **privacy-preserving validation** without exposing internal state transitions.

Conflict Resolution

- If buddy digests diverge, **CRDT-based reconciliation** merges results to achieve eventual consistency.
- A **temporary scoped leader** may coordinate the merge, but no permanent authority exists.
- **Bloom filters** prevent replayed or duplicate messages, lowering bandwidth overhead and speeding up block relay.

Immutable Ledger via ImmuDB

- Finalized blocks are committed to **ImmuDB**, ensuring:
  - **Append-only, tamper-proof history**,
  - **Auditability** for enterprise use cases,
  - Separate storage of flagged or invalid transactions for accountability.

8.2 Security and Operational Benefits

Feature	Benefit
🔐 zk-proof validation	Privacy-preserving and verifiable transaction proofs

Feature	Benefit
⚖️ VRF + quorum	Randomized, fair buddy sets; strong Byzantine fault tolerance
🌐 Gossip + Bloom	Efficient, low-latency peer-to-peer communication
📜 CRDT-based reconciliation	Convergent state even under partitions
🪨 Incentive alignment	Sequencer rewarded only after quorum validation succeeds
📊 Adaptive weighting	Seed Nodes penalize malicious nodes and reward honest ones

8.3 Why This Matters

The **AVC protocol** ensures that block finality in the JMDT ecosystem is:

- **Verifiable** through zk-proofs,
- **Decentralized** through randomized buddy-set validation,
- **Responsive** with asynchronous quorum instead of synchronized rounds,
- **Audit-ready** with immudb-backed append-only history,
- **Adaptive** via dynamic weight updates that favor reliable nodes.

This design also forms the foundation for **Layer-3 DAG extensions** and enterprise-specific consensus frameworks, making AVC a **future-proof consensus model** for data-driven blockchain applications.

---

9. Security Model

> The security of the **JupiterMeta Data Token (JMDT)** protocol is anchored on a layered approach involving **cryptographic guarantees** and **ZK-based verifiability** across L3, L2, and L1 layers.

9.1 Trust and Threat Assumptions

Layer	Trust Model	Threats Mitigated
L1 (Ethereum)	Assumes Ethereum finality and censorship-resistance	Network reorgs, malicious smart contracts
L2 (JMDT Chain)	Assumes ≥ 2/3 honest, randomized and democratized validators in AVC	Sybil attacks, block manipulation, equivocation
L3 (Enterprise DAG)	Assumes internal integrity within enterprise	Insider tampering, unauthorized data access
ZK Proof System	Assumes soundness of zkSNARK/zkSTARK cryptographic assumptions	Proof forgery, data leakage

Layer	Trust Model	Threats Mitigated
RISC Zero zkVM	Assumes deterministic Rust guest execution	Non-determinism, circuit manipulation

9.2 Security Guarantees

1. **Integrity**:
   - All L2 and L3 state transitions are validated via **zk-SNARK/STARK proofs**.
   - Transactions are only finalized if proofs are verified and consensus is reached.
2. **Confidentiality**:
   - Personally identifiable information (PII) is never committed to-chain.
   - DID-based authentication ensures access control without revealing identity.
   - L3 DAGs keep enterprise data off-chain; only zk-committed proofs reach L2.
3. **Availability**:
   - L2 consensus leverages **Gossip Protocol** for fast propagation.
   - DAGs are internally redundant and support CRDT-based recovery.
4. **Accountability**:
   - All block proposals and zk-proofs are attributable to known DID identities or validator keys.
   - Invalid or conflicting proofs are rejected automatically by L2 validators.

9.3 Potential Attack Surfaces

Vector	Mitigation
Sybil attacks on validator set	Unpredictable Randomized Validator Sets + DID verification
zk-circuit manipulation	Rust-based zkVM ensures reproducible, deterministic execution
Data leakage in queries	All data access is gated via DID + ZK queries with no plaintext exposure
L3 DAG integrity compromise	DAG nodes must commit proofs to L2; failures are detected and slashed
Delayed L1 commitments	Dynamic L1 batching policy ensures gas-fee aware but time-bounded commits
Smart contract bugs	All Ethereum contracts are audited and open-sourced

9.4 Formal Verification and Auditing

- All zkVM guest programs undergo **formal verification** for determinism and safety.

- Ethereum smart contracts (L1 commit verifier, JMDT logic) will be audited by independent firms prior to mainnet deployment.
- The DID module conforms to **W3C standards** and supports verifiable credential flows.

This security model ensures **robust protection across privacy, integrity, and availability dimensions** while maintaining performance, scalability, and compliance.

---

10. Tokenomics & Incentives

> The total supply of JMDT is hard capped at 1 Billion tokens, ensuring a fixed and transparent monetary base for sustainable growth and decentralized governance. The allocation is thoughtfully structured to incentivize participation, support long-term ecosystem development, and provide liquidity for market access.

10.1 Allocation Breakdown

The token distribution is divided into two major pools:

- L3 Developments, Enterprises & Early Adopters – 55%
  This is the largest allocation and supports real-world adoption through incentivizing L3 applications, enterprise integrations, DAG-based data contributors, testnet validators, and ZK-proof participants. It anchors the network's utility and network effects.
- R&D and Foundation Initiatives – 10%
  Reserved for protocol-level research, zero-knowledge innovation, cryptographic upgrades, and grants to academic or research institutions building on the JupiterMeta blockchain.
- Founding Team and Employees – 10%
  Allocated to the core team responsible for engineering, cryptography, business, and operational execution. Tokens are subject to long-term vesting schedules to ensure commitment.
- Founders and Promoters – 25%
  Designated for the original visionaries and promoters behind the JMDT protocol. Locked and vested to reflect long-term alignment with the ecosystem's mission and values.

10.2 Vesting & Unlock Strategy

To protect long-term network integrity and reduce sell pressure:

- All team, promoter, and investor tokens are subject to **cliff-based linear vesting**.
- Early adopter tokens are released in **phased batches** over a 6–12 month horizon.
- The Foundation's deployment of ecosystem grants and treasury tokens is governed by internal frameworks.

This allocation model ensures the **JMDT token remains utility-focused**, incentivizes genuine contribution, and unlocks sustainable value for all stakeholders — from developers and data contributors to validators and enterprise partners.

---

11. How Enterprises build on JMDT Blockchain Stack

1. Choose Your Enterprise Layer

- **L3 DAG Network:** Enterprises deploy private, high-throughput Directed Acyclic Graph (DAG) nodes for internal data operations (e.g., user activity, insights, compliance logs).
- Each L3 node is permissioned, independently scalable, and **anchored to JMDT Layer 2** for rollup-based finality and zk-proof auditability.

2. Set Up Enterprise DID & Access

- Register a **Decentralized Identifier (DID)** representing the enterprise and link sub-identities for departments or teams.
- Manage keys using secure wallets.

3. Deploy Custom Workflows on DAG

- Build DAG-based business logic in Python or Rust—e.g., for supply chains, compliance, CRM, or AI-inference logs.
- Use the DAG SDK and APIs to track state transitions and hash each operation.

4. Rollup to Layer 2 (JMDT Chain)

- Periodically batch DAG transactions and submit zk-rollups to L2 using **zk-SNARK proofs**.
- JMDT tokens are used to commit rollups, and **Ethereum L1 finality** is optionally supported.

5. Use JMDT for Operations

- **Jupiter Meta Data Token (JMDT)** fuels DAG rollups, L2 commitments, and node staking.
- Enterprises receive JMDT via private allocation, grants.

6. Run or Choose Infrastructure

- Host your L3 DAG node on GCP, AWS, or on-prem, with integration into **JMDT.io RPC**, IPFS, and immudb-based logs.
- Use lightweight sequencers or relayers for DAG-to-L2 communication.

7. Monitor & Reward

- Use **Explorer dashboards** to visualize DAG graphs, participation, rollup health, and proof verification.

- Trigger user or team-level rewards via smart contracts or DID-linked incentive programs.

11.1 Layer 3s on JMDT Blockchain

Hercules

Hercules is a research assistant for data-backed decision making.

SuperJ

A decentralised consumer intelligence app that currently allows over 20 millions users to ZK-verify their identity, share data and monetize information.

---

12. Comparative Analysis

> The following table compares **Jupiter Meta Data Token (JMDT)** with leading L2 blockchain infrastructure projects across key dimensions relevant to both enterprise and developer adoption.

Feature / Platform	JupiterMeta Data Token (JMDT)	Polygon zkEVM	Starknet	Aztec Network	Hyperledger Fabric
Layer Type	L2 + L3 (Enterprise DAG)	L2 (zkEVM)	L2 (STARK Rollup)	L2 (Private zk-SNARKs)	Private Consortium
Scalability (TPS)	2,000+ TPS (L2); 10K+ (L3 DAG)	2,000–4,000 TPS	3,000+ TPS	<100 (privacy bottleneck)	Depends on config
Privacy Model	zk-SNARK/STARK + DID + zkVM	zk-SNARKs	zk-STARKs (public)	zk-SNARKs (Private Tx)	No native privacy
zk Circuit Strategy	Rust + RISC Zero zkVM	Circom + Groth16	Cairo VM	Circom + Aztec Noir	None
Finality Time (L2)	~3–10 sec	~5–10 min	~10–15 min	~30–60 min	Instant (centralized)
L1 Settlement (Ethereum)	Dynamic zk-rollup to L1	Periodic zk-rollup	STARK rollup	Delayed zk-rollup	Not applicable
Data Availability	zk-anchored + immudb logs	Ethereum DA Layer	On-chain calldata	On-chain	Off-chain (private)
Enterprise Readiness	✅ DAG layer, DID, privacy	⚠ Limited DID/enterprise SDKs	⚠ Limited identity & compliance features	❌ Not enterprise-ready	✅ Proven enterprise use

Feature / Platform	JupiterMeta Data Token (JMDT)	Polygon zkEVM	Starknet	Aztec Network	Hyperledger Fabric
Token Model	JMDT (Governance + Utility)	MATIC (Gas & Staking)	STRK	AZTEC	None (no token)
Smart Contract Support	Rust (zkVM) + Solidity	Solidity (zkEVM)	Cairo	Aztec Noir + limited Solidity	Chaincode (Go/Java)
Governance	Foundation	Off-chain + Foundation	TBD (future DAO)	Aztec Foundation	Centralized consortium

Key Takeaways:

- **JMDT** uniquely combines **scalability**, **privacy**, and **enterprise DAGs**, with **AVC consensus** and **zkVM-based Ethereum finality**.
- It bridges the gap between **public zk-rollups** (like Starknet) and **enterprise-grade blockchains** (like Hyperledger).
- The inclusion of **Decentralized Identity (DID)** and **privacy-preserving queries** gives JMDT a clear advantage for GDPR-compliant and audit-friendly use cases.

---

13. Conclusion

JupiterMeta Data Token (JMDT) represents a paradigm shift in blockchain infrastructure, combining **privacy, scalability, and decentralization** into a single, seamless ecosystem. By integrating **Zero-Knowledge Proofs (ZKPs), Decentralized Identity (DID), and AVC consensus mechanism**, JMDT ensures **privacy-enhanced, high-performance, and verifiable transactions** across diverse use cases.

JMDT's **Layer 2 (L2) framework** optimizes transaction throughput through **zk-rollups**, enabling **cost-effective and scalable interactions** while committing transactions to **Ethereum (L1) dynamically** based on the best available gas fees, block time, and network conditions. The **Layer 3 (L3) DAG architecture** provides **a high-throughput enterprise solution**, allowing businesses to manage **private, off-chain transactions with cryptographic integrity** while anchoring verifiable state proofs on L2.

JMDT also brings **enterprise-grade security and compliance** through **DID-based authentication**, ensuring **secure, trustless identity verification** without exposing personal data.

The **AVC consensus model**, integrating **Bloom Filters**, and **asynchronous quorum validation**, enables rapid and fault-tolerant block propagation. It minimizes communication overhead, supports efficient peer validation without global coordination, and achieves robust state convergence through **zk-proof–backed validation** across decentralized buddy nodes.

With **Ethereum interoperability, zk-proof-based privacy preservation, and tokenomics-driven governance**, JMDT fosters an ecosystem that supports:

- **DeFi innovation** with privacy-focused smart contracts and optimized transaction finality.
- **Web3 data marketplaces** that enable **secure, private, and decentralized data exchange**.
- **Enterprise applications** that require **confidentiality, high throughput, and regulatory compliance**.
- **Decentralized identity solutions** that empower users with **sovereign control over their credentials**.

The **future of blockchain** demands a balance between **efficiency, privacy, and interoperability**and JMDT delivers on all fronts. By pioneering **scalable zero-knowledge rollups, enterprise DAG integration, and DID-based access control**, JMDT is poised to become **the backbone of next-generation decentralized infrastructure**.

For collaborations and inquiries, visit docs.jmdt.io

Document Version: 1.3 | Nov 2025

jmdt