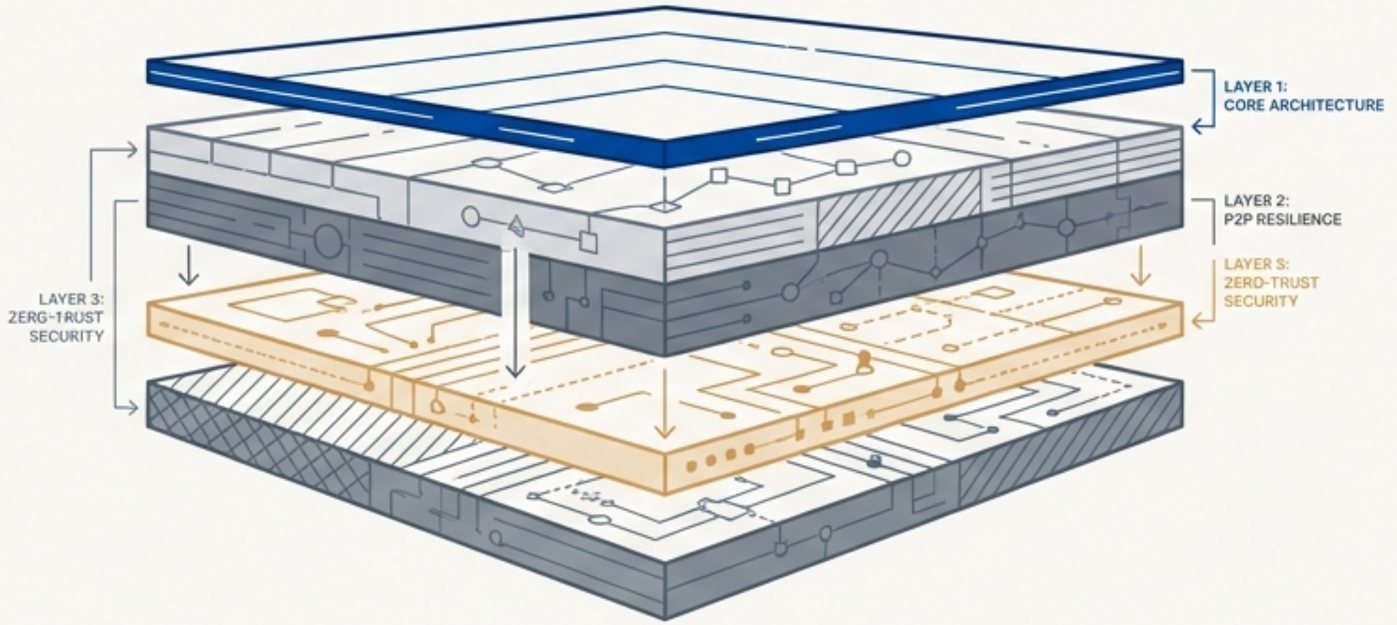
GenosDB: The Minimalist Graph Database for Real-Time, Decentralized Applications.



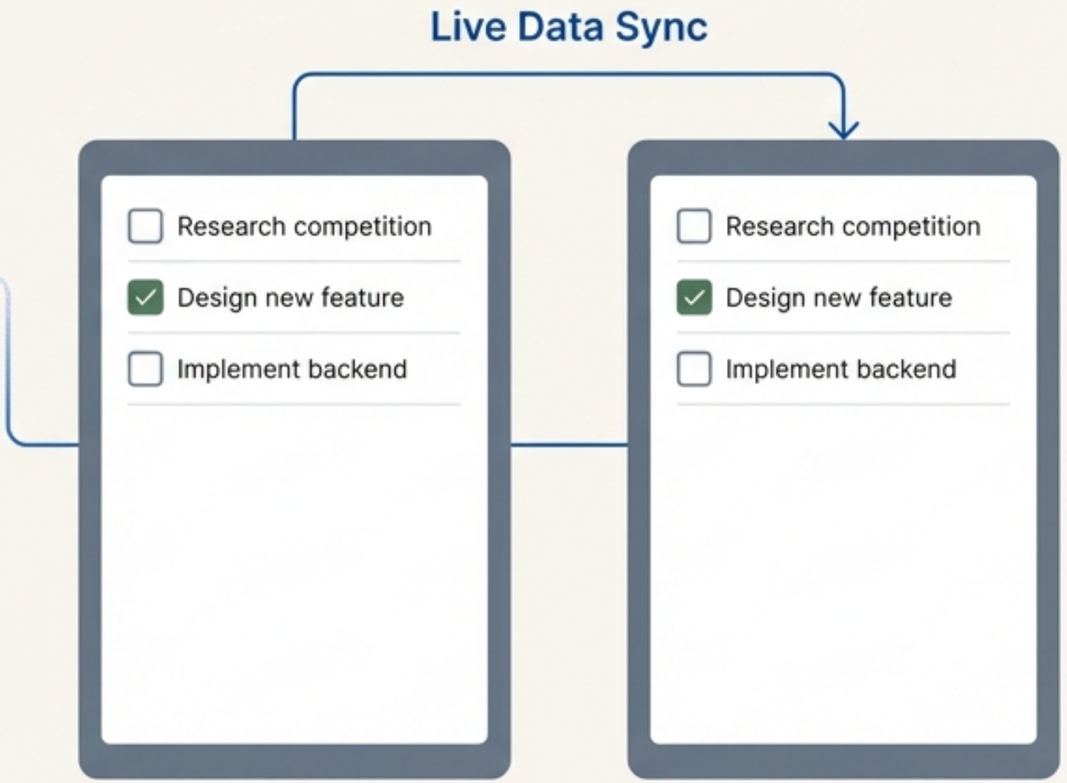
A tour of its effortless developer experience, resilient P2P architecture, and zero-trust foundation.

API Status: Stable Beta

Build Real-Time UIs with a Declarative API

GenosDB is designed around a reactive core. Forget manual state management and REST polling. Subscribe to a query, and your UI updates automatically as data changes, whether locally or from a peer across the globe.

```
// Power a live to-do list with a single subscription
const { unsubscribe } = await db.map(
  { query: { type: 'task' }, field: 'createdAt' },
  ({ id, value, action }) => {
    // Logic to add, update, or remove items from the UI
    // based on the 'action' ('initial', 'added', 'updated', 'removed')
  }
);
```



The Core API: Simple Verbs for Powerful Actions.

The API is designed for clarity and efficiency. Four core methods handle all persistent state management. Top-level `await` is recommended for the cleanest code.

put(value, id?)

Creates or updates a node.

```
const userId = await db.put({ name: 'Alice',
role: 'user' });
```

get(id, cb?)

Retrieves a node, with an optional callback for real-time updates on that specific node.

```
const { result } = await db.get(userId);
```

link(sourceId, targetId)

Creates a directed relationship between two nodes.

```
await db.link(projectId, userId);
```

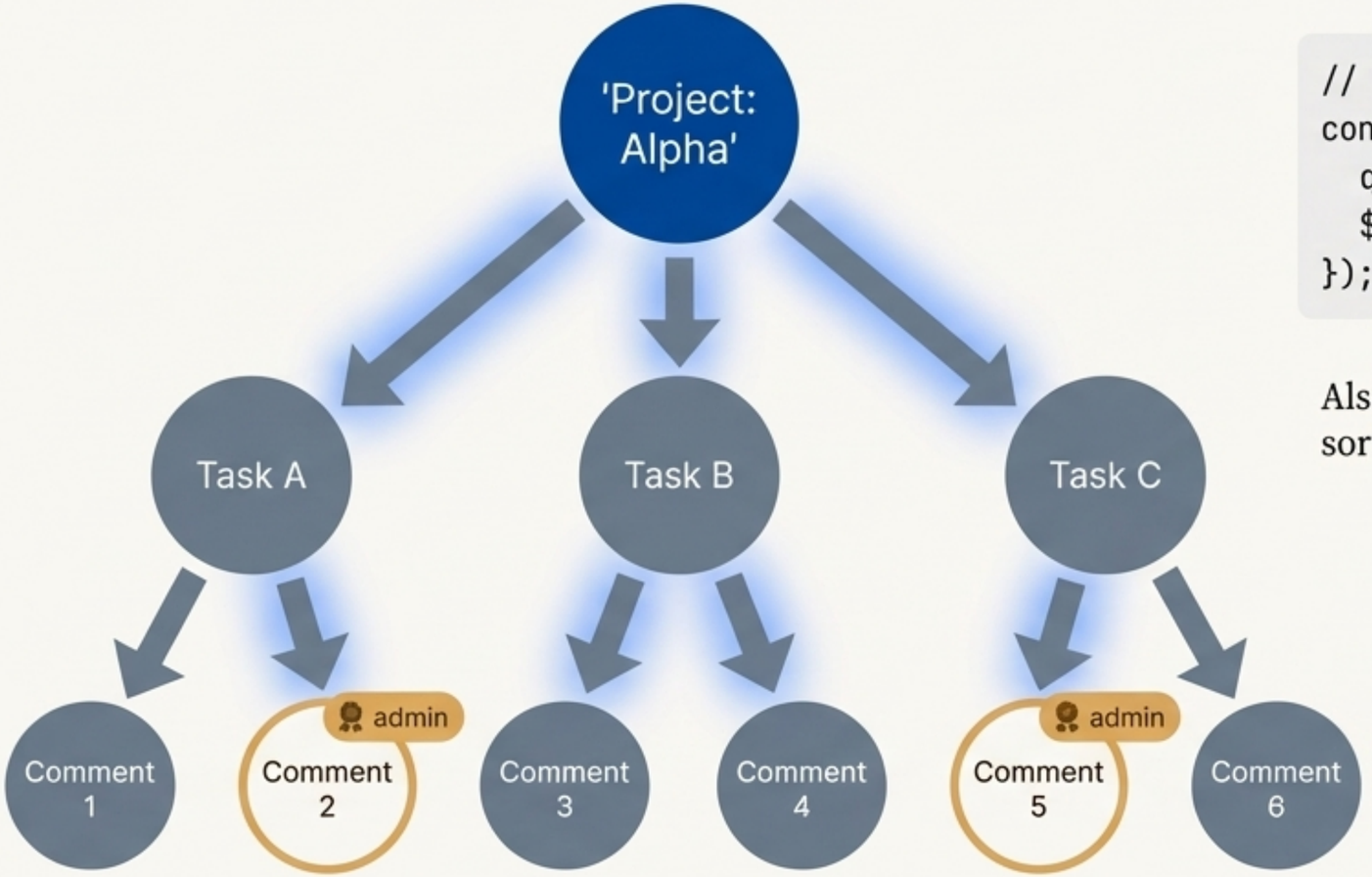
remove(id)

Deletes a node and cleans up all its references.

```
await db.remove(userId);
```

Unleash Graph Power with Recursive Traversal.

The `map` method is more than a filter; it's a graph exploration tool. The `$edge` operator traverses a node's entire descendant tree, returning a flat list of descendants that match a sub-query. This allows for complex, multi-hop traversals in a single, declarative call.

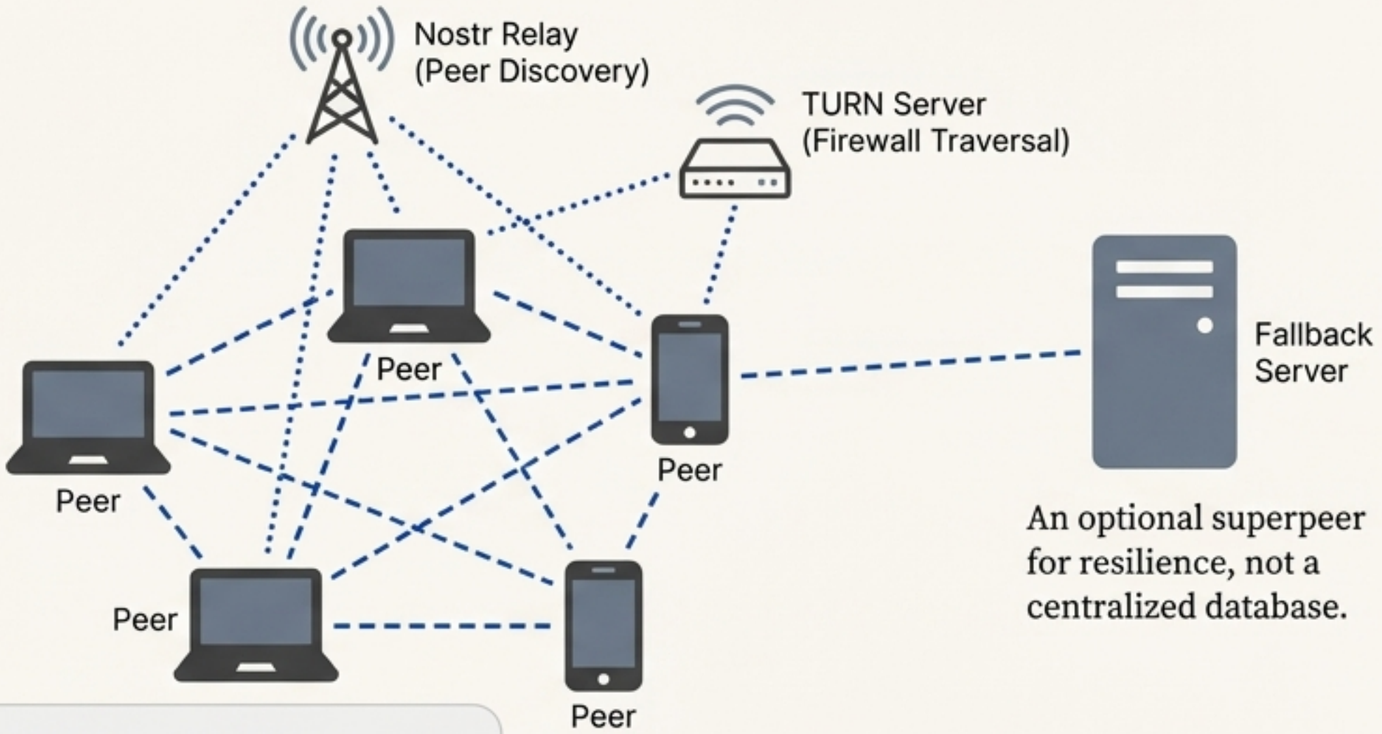



```
// Find all comments in 'Project Alpha' made by admins
const { results } = await db.map({
  query: { type: 'Project', name: 'Alpha' },
  $edge: { type: 'Comment', authorRole: 'admin' }
});
```

Also supports MongoDB-style filtering (`$gt`, `$in`, `$or`), sorting, and pagination.

How It Works: The Resilient P2P Sync Engine

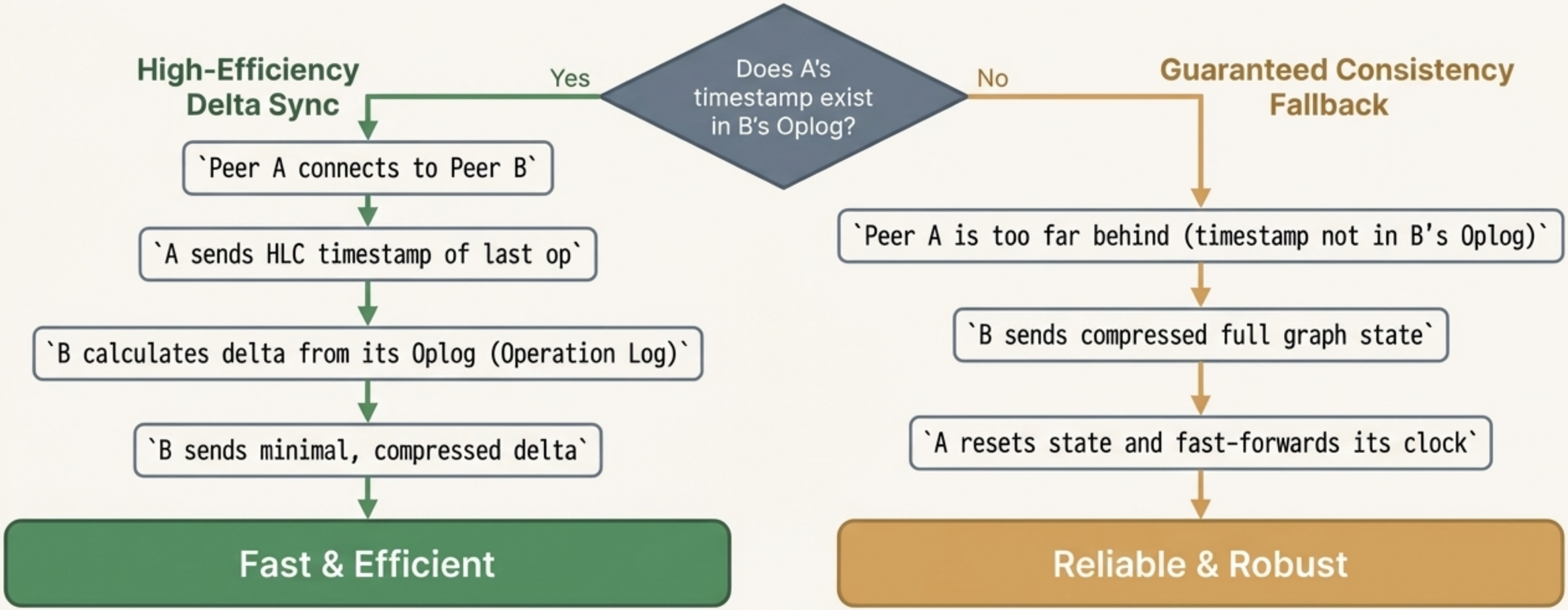
GenosDB runs serverless by default. The `GenosRTC` module enables automatic, real-time, delta-based synchronization of all graph operations between connected peers. There is no central authority; the database lives at the edge.




```
const db = await gdb('my-app-db', {
  rtc: true, // Enables the entire P2P stack
  relayUrls: ['wss://...'], // Optional custom relays
  turnConfig: [{…}]         // Optional TURN servers
});
```

Always in Sync: The Hybrid Delta Protocol

The sync engine intelligently switches between two modes to ensure eventual consistency with optimal performance. This provides the speed of delta-syncing with the absolute reliability of full-state reconciliation.

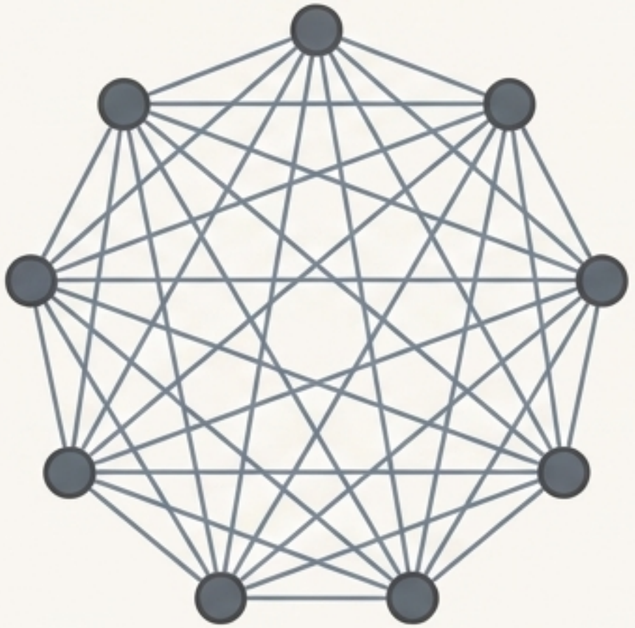


Scaling from Small Teams to Massive Networks.

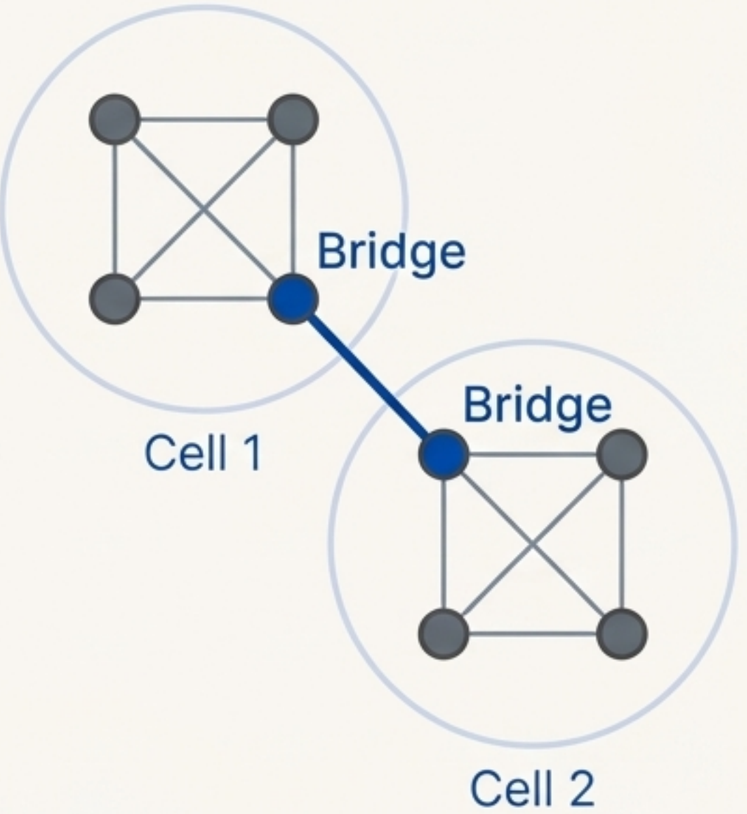
Standard P2P mesh networks don’t scale. GenosDB implements an optional Cellular Mesh overlay, organizing peers into logical “cells” with bridge nodes for inter-cell communication.

This reduces connection complexity from $O(N^2)$ to $O(N)$.

Standard P2P: O(N²)

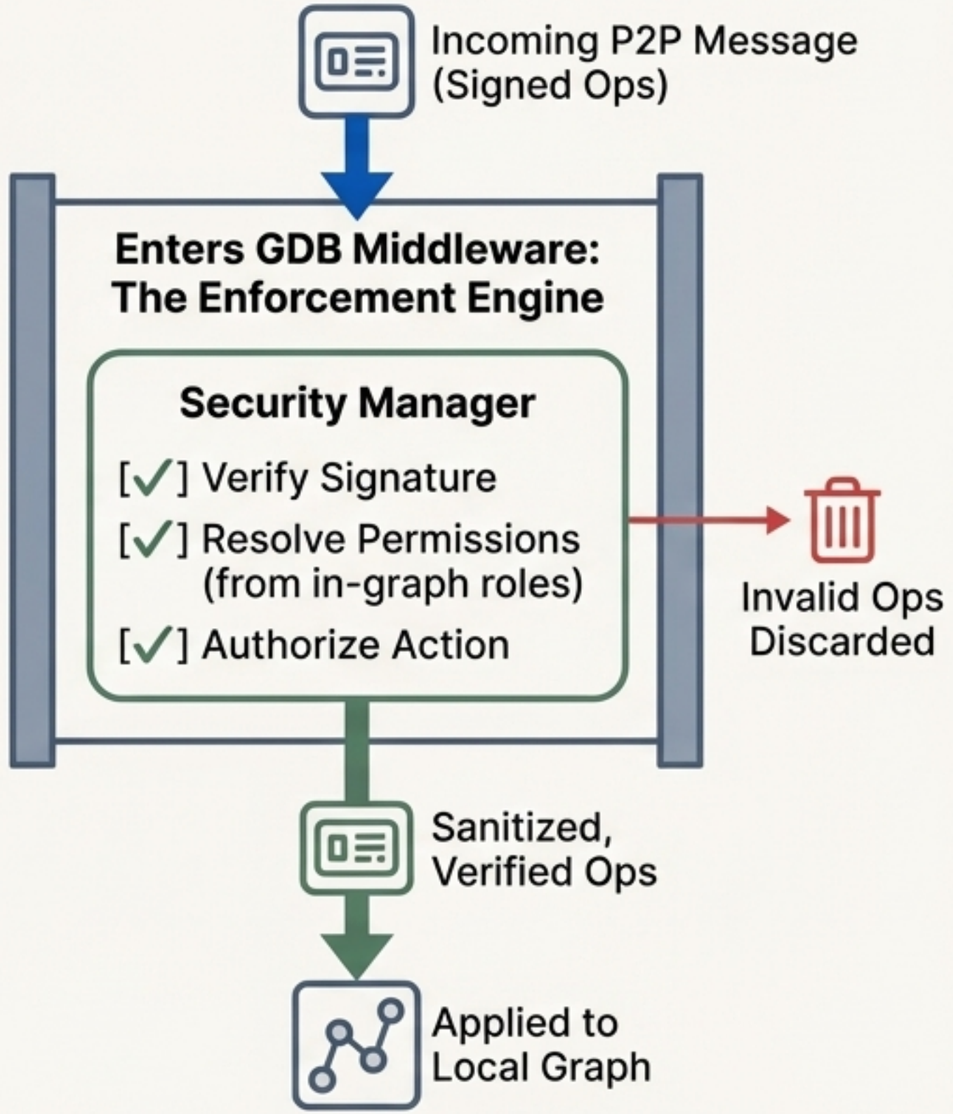


Cellular Mesh: O(N)



The Zero-Trust Foundation: Integrated Security & Identity

The **Security Manager** (SM) transforms GenosDB into a secure platform by **integrating a complete security model directly** into the **data pipeline**. Every **peer independently validates and enforces** the network's rules. Unauthorized data is never applied.

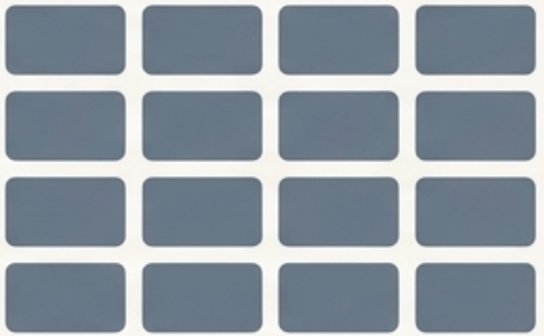


Sovereign Identity with Modern Authentication

User identity is established through a cryptographically secure Ethereum-style key pair, giving users true ownership. All permissioned operations are signed, providing an unforgeable proof of origin.

WebAuthn

Passwordless login using biometrics or hardware keys. Supports silent session resume for a seamless UX.



Mnemonic Phrases

Implements BIP39 for deterministic key generation, allowing for user-friendly account creation and recovery.

Distributed Authorization with In-Graph RBAC

The Role-Based Access Control system is embedded and synchronized as part of the graph data. Security rules are distributed and agreed upon with the same eventual consistencyas the data they protect.

Customizable Role Hierarchy

```
superadmin
 > admin
    > manager
       > user
          > guest
```

Granular Permissions

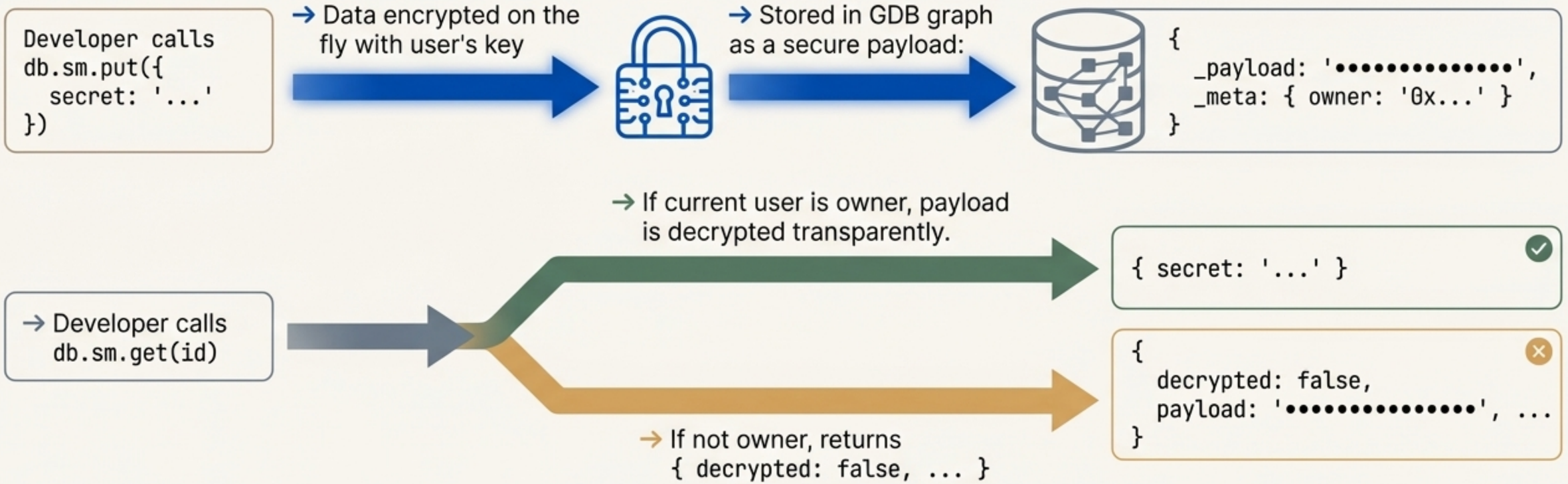
`read` `write` `link` `delete` `deleteAny` `assignRole`

Time-based Roles

Role assignments can be set with an optional expiration date.

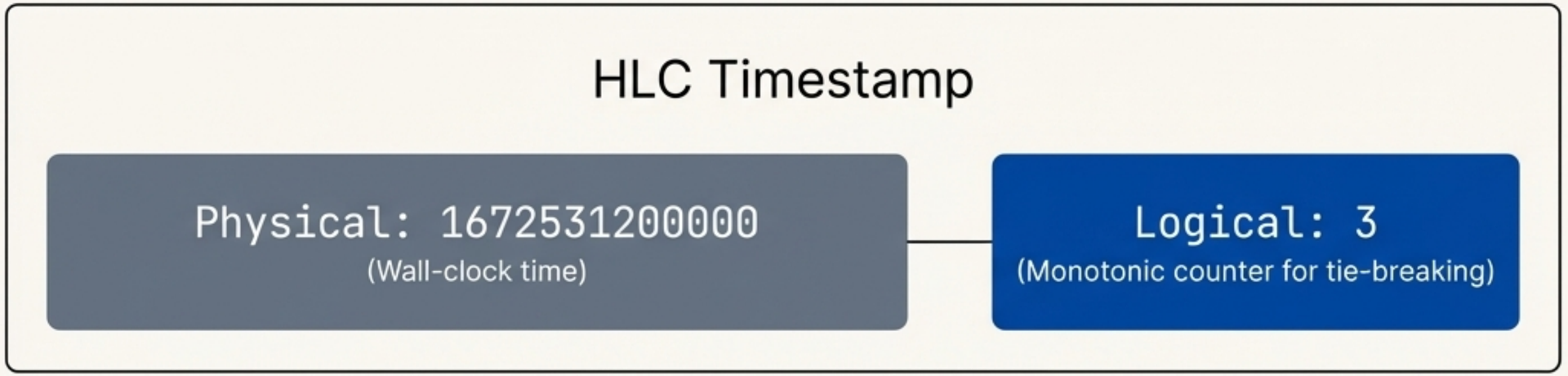
True Privacy with Transparent End-to-End Encryption

The Security Manager provides user-controlled E2EE. Data is encrypted with a key derived from the user's private key. Only the owner can decrypt it. For all other users, the data remains unintelligible cip ciphertext.



Guaranteeing Consistency with Hybrid Logical Clocks.

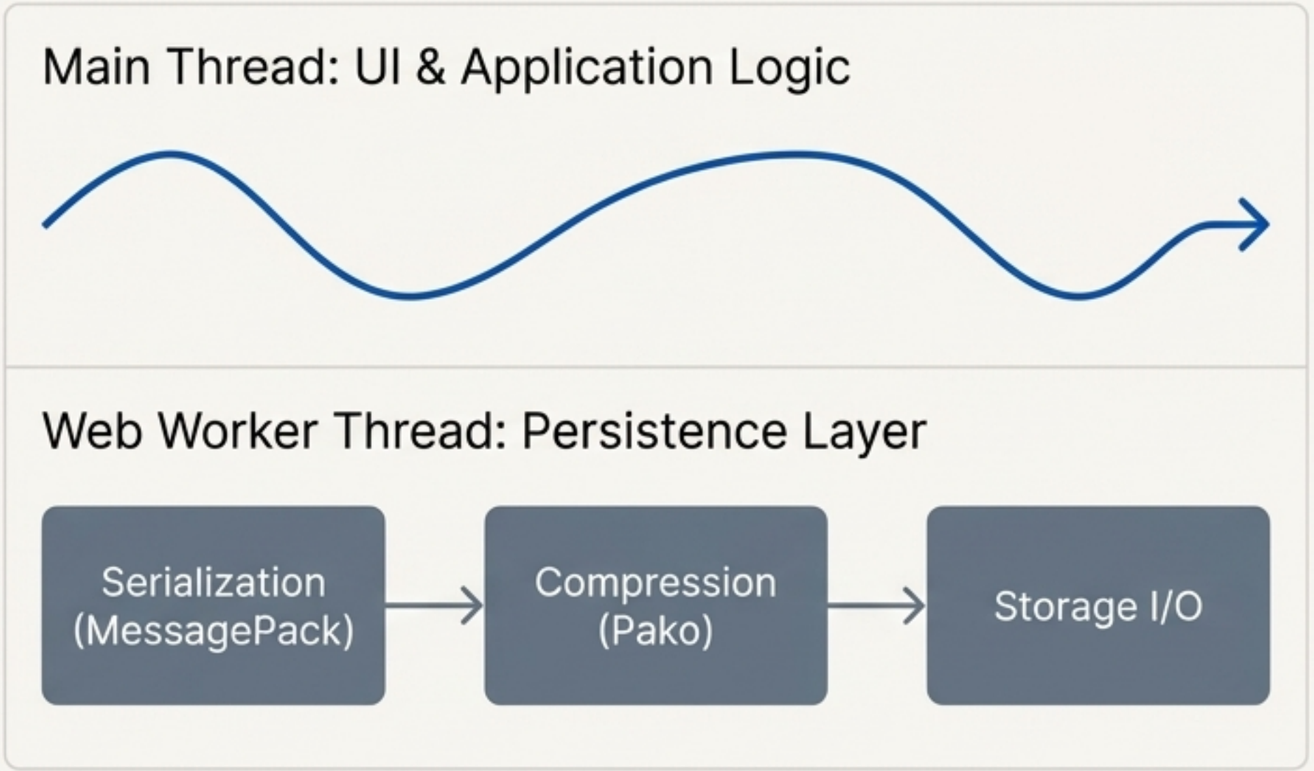
To ensure deterministic conflict resolution, every operation is timestamped with a Hybrid Logical Clock (HLC). This combines physical time with a logical counter, preserving causal order and enabling a reliable “Last-Write-Wins” strategy, even with system clock skew.



When resolving conflicts, the `physical` component is compared first. The `logical` component is used only to break ties, ensuring a total ordering of all events in the system.

A Non-Blocking Core for a Fluid User Experience

All intensive I/O operations—storage access, serialization, and compression—are offloaded to a dedicated Web Worker. This ensures the main UI thread remains responsive at all times, even under heavy write loads.



Tiered Storage Strategy

1. **Primary:** Synchronous OPFS (Origin Private File System) for lowest latency.
2. **Secondary:** Asynchronous OPFS.
3. **Fallback:** IndexedDB for universal compatibility.

Your Journey with GenosDB Starts Now.

You have the tools to build powerful, real-time, and decentralized applications with a simple, reactive API. We're excited to see what you'll create.

📖 Explore Examples

Dive into our examples guide to see complete, working code for common use cases.

`View Examples`

🛰️ Master P2P Communication

For advanced features like video/audio streaming, consult the full GenosRTC API Reference.

`Read RTC Docs`

🐞 Contribute & Give Feedback

Your feedback is invaluable. Open an issue or contribute on GitHub.

`GitHub Repository`