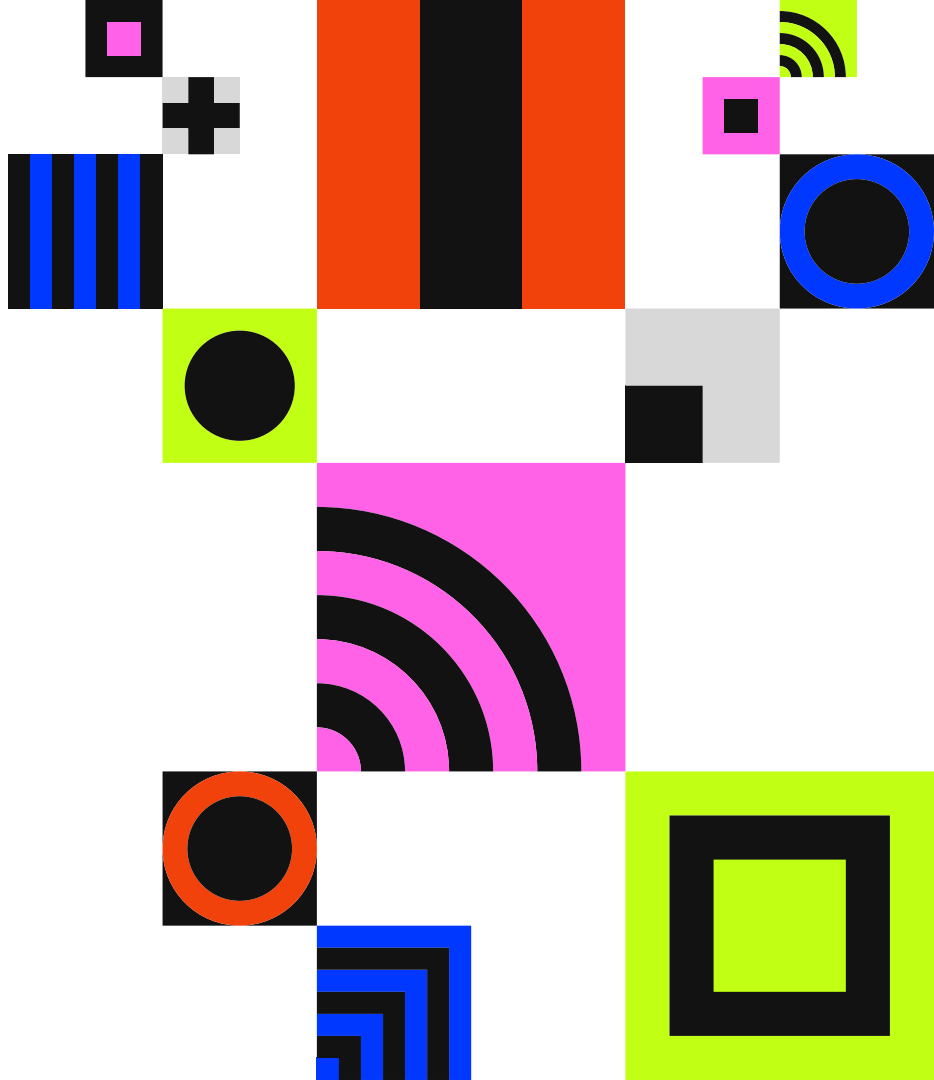
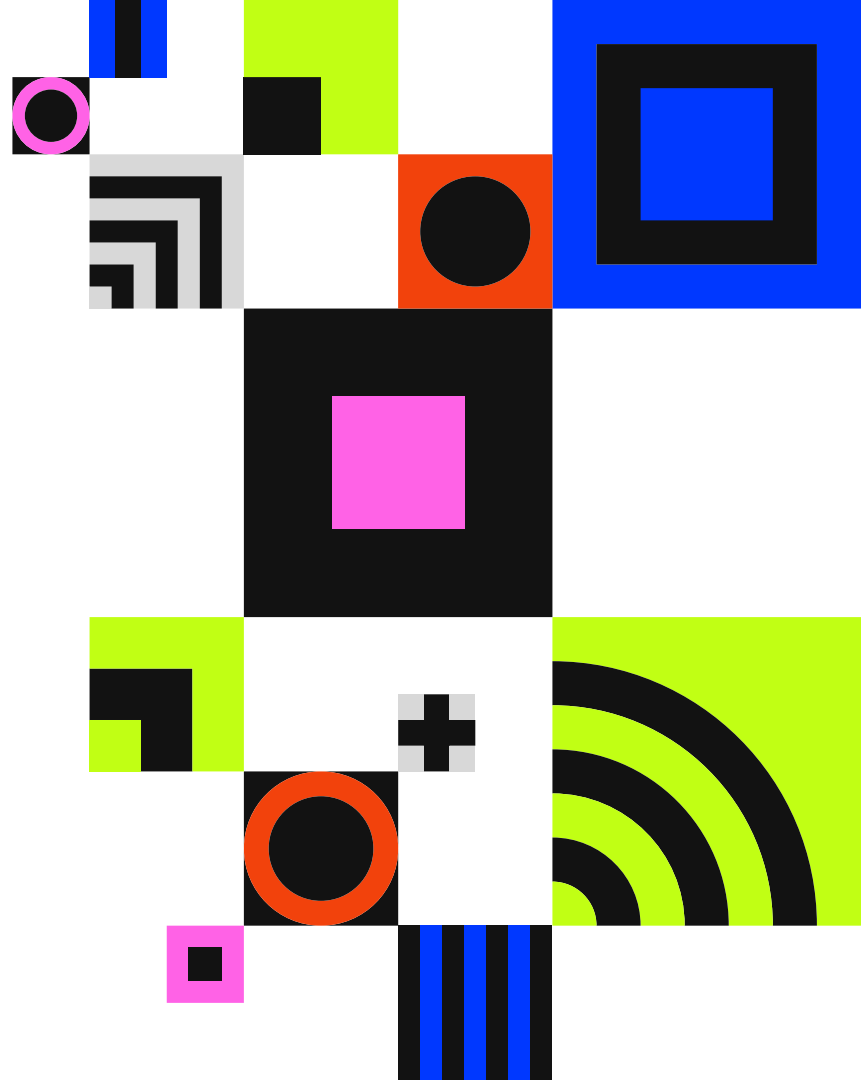


# Besu Evolution



# Agenda

- 1 Besu Roadmap: A Look Back & Ahead
- 2 Besu & Linea
- 3 Private Network Evolution



# Besu Roadmap: A Look Back & Ahead

# Hyperledger Besu 2H 2023 Roadmap

[Link](#)
\* May be delayed


**HYPERLEDGER**  
**BESU**

## Multi-Chain Besu, Flexible Infrastructure

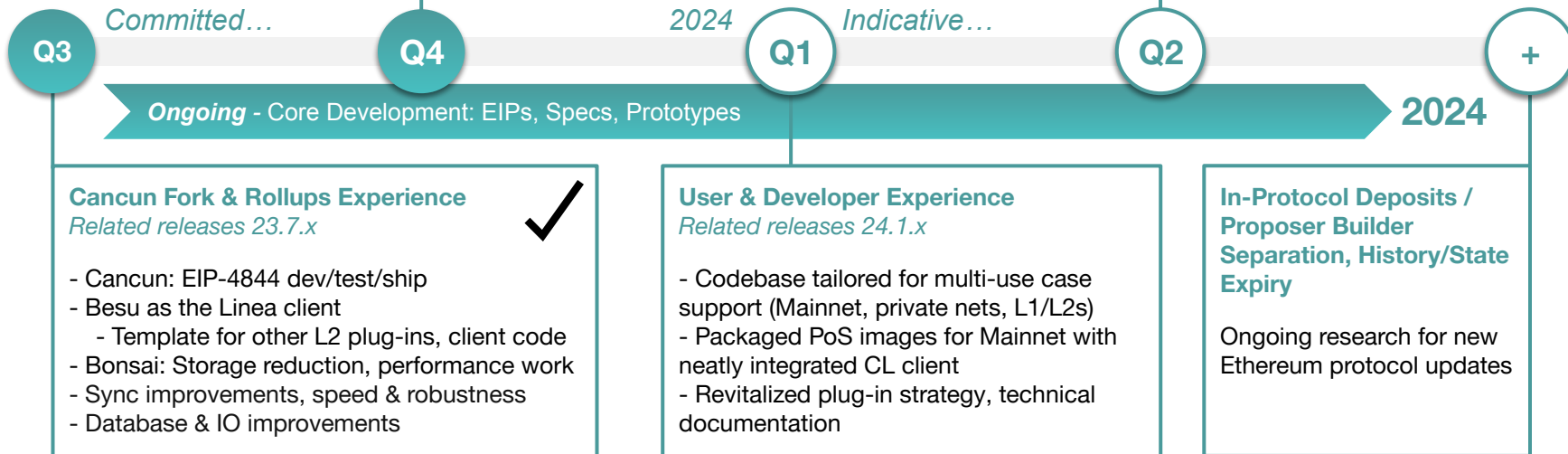
*Related releases 23.10.x*

- Modular consensus mechanisms via plug-ins
- Fleet: new feature, trie-log shipping for Bonsai
  - Light client infrastructure for near-head scale
- Sync: new formats for validators, snap server
- Bonsai-friendly Archive

## Prague Fork, Client Evolution

*Related releases 24.4.x*

- Prague: EOF dev/test/ship
- Verkle Tries: prototype
  - Bonsai adaptation for Verkle, full light client exploration
- Besu as a customizable L2/L3 sequencer



# Look Back

Q3

## Cancun Fork & Rollups Experience

*Related releases 23.7.x*



- Cancun: EIP-4844 dev/test/ship
- Besu as the Linea client (Q3/Q4)
  - Template for other L2 plug-ins, client code
- Bonsai: Storage reduction, performance work
- Sync improvements, speed & robustness
- Database & IO improvements

- Cancun
  - 4844 - Data-availability scaling for Mainnet (and cheaper L2s)
  - Modifications to proof of stake
  - New OpCodes (TStore, others)
- Bonsai improvements
  - Further storage reductions (still in progress)
  - Performance improvements, shipped flat DB
  - Preparing for Verkle Tries
- Sync
  - SnapSync robustness improvements
- RPC Improvements
  - Performance improvements on Debug and Trace namespace
  - Accuracy work, cleanup for spec compliance
- Consensys Team on Linea
  - Plug-in approach for Besu on L2
    - ZK Tracer Plug-In
    - Sequencer Plug-In

# Look Forward

## Multi-Chain Besu, Flexible Infrastructure

*Related releases 23.10.x*

- Modular consensus mechanisms via plug-ins
- Fleet: new feature, trie-log shipping for Bonsai
  - Light client infrastructure for near-head scale
- Sync: new formats for validators, snap server
- Bonsai-friendly Archive

Q4

Q1

## User & Developer Experience

*Related releases 24.1.x*

- Codebase tailored for multi-use case support (Mainnet, private nets, L1/L2s)
- Packaged PoS images for Mainnet with neatly integrated CL client
- Revitalized plug-in strategy, technical documentation

- Bonsai Evolution
  - Archive-friendly Bonsai
- Fleet-Mode, near-head RPC scaling
  - Using Bonsai's flat state DB, lighter nodes can serve state RPC and scale up and down quickly
- Sync
  - SnapSync Server (all networks on Bonsai)
  - Validator-focused sync on Mainnet
- Plug-In Revitalization
  - Gathering requirements from Linea work and developer community for feature development
  - New documentation with open-source templates
- Modular Consensus mechanisms
  - May slip to 2024...
  - Decoupling of consensus mechanism from Besu components
- Codebase Cleanup
  - Follows consensus cleanup for more modular multi-use case support

# Looking Ahead

- Bonsai everywhere...
  - Simplified code-base, simplified configuration, simplified testing
  - Faster sync, faster performance, on both public & private networks
  - Future compatible with Verkle Tries
- Besu everywhere...
  - Focus on cross-cutting improvements to Mainnet and private net UX
  - More L2 participation following plug-in / Linea pattern
  - Single-sequencer Besu roll-ups everywhere
    - Scale for existing public and private networks

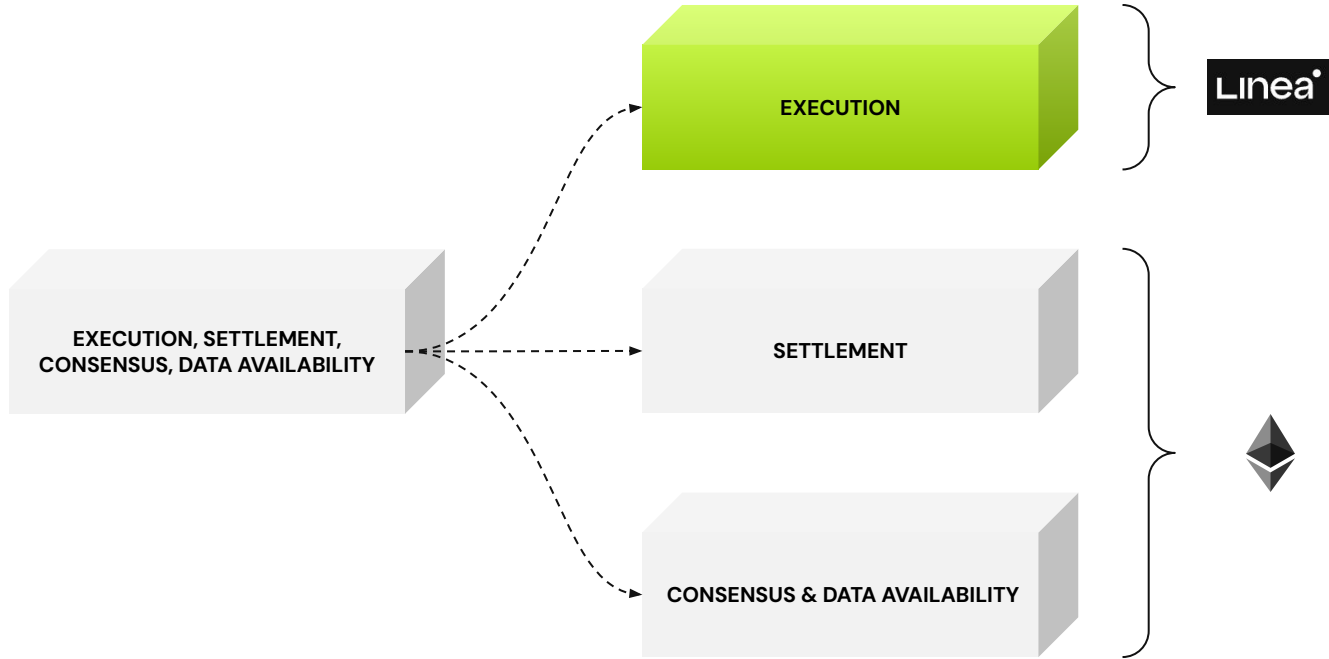


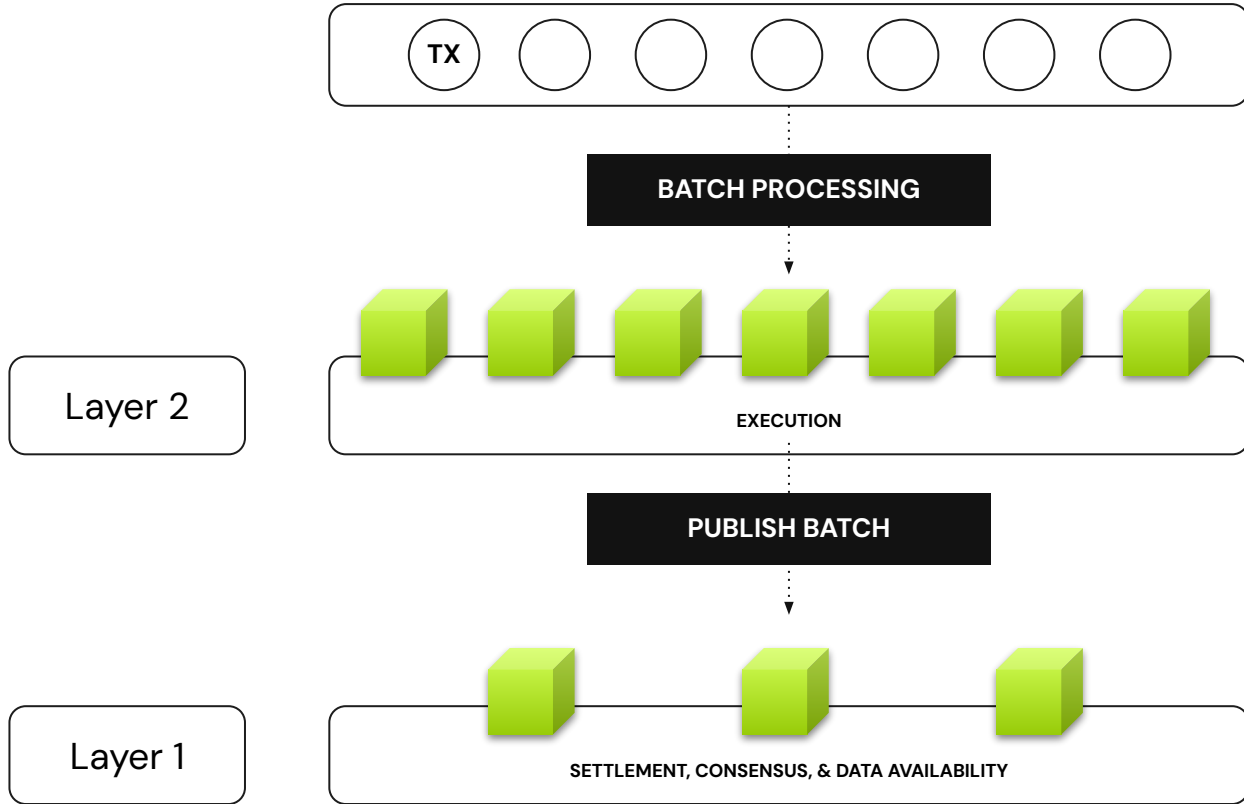
# Linea-Besu & the Modular Blockchain



### MONOLITHIC

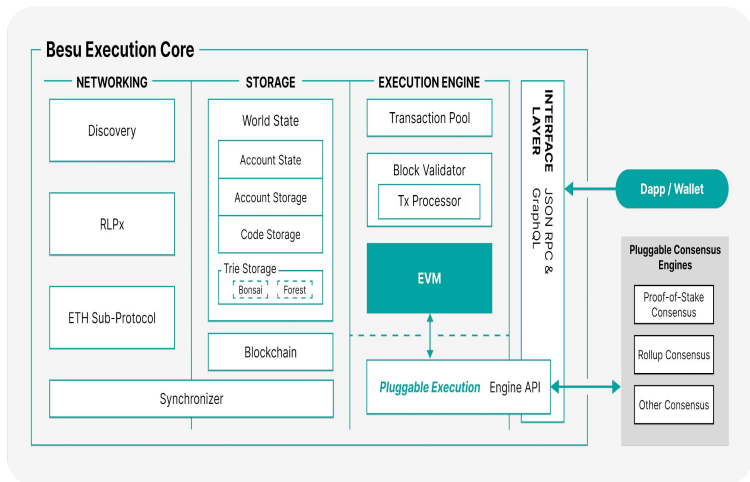
### MODULAR





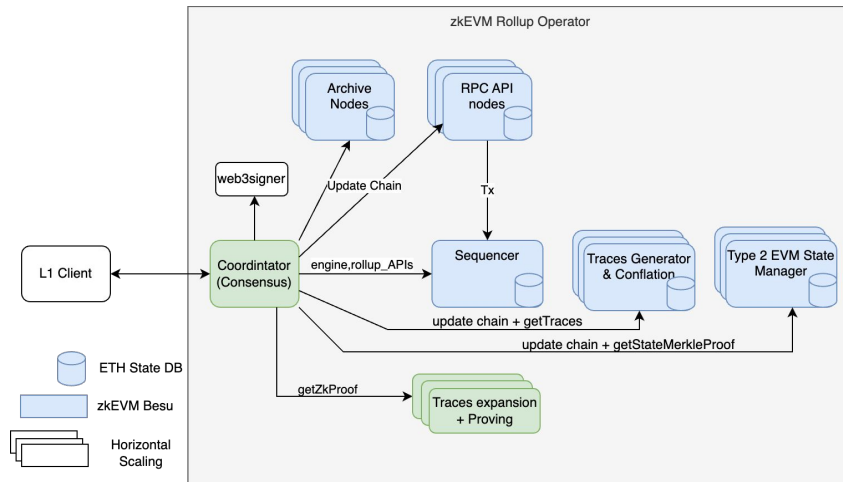
# Adapting Execution Context

Besu was built as a Swiss-army knife...



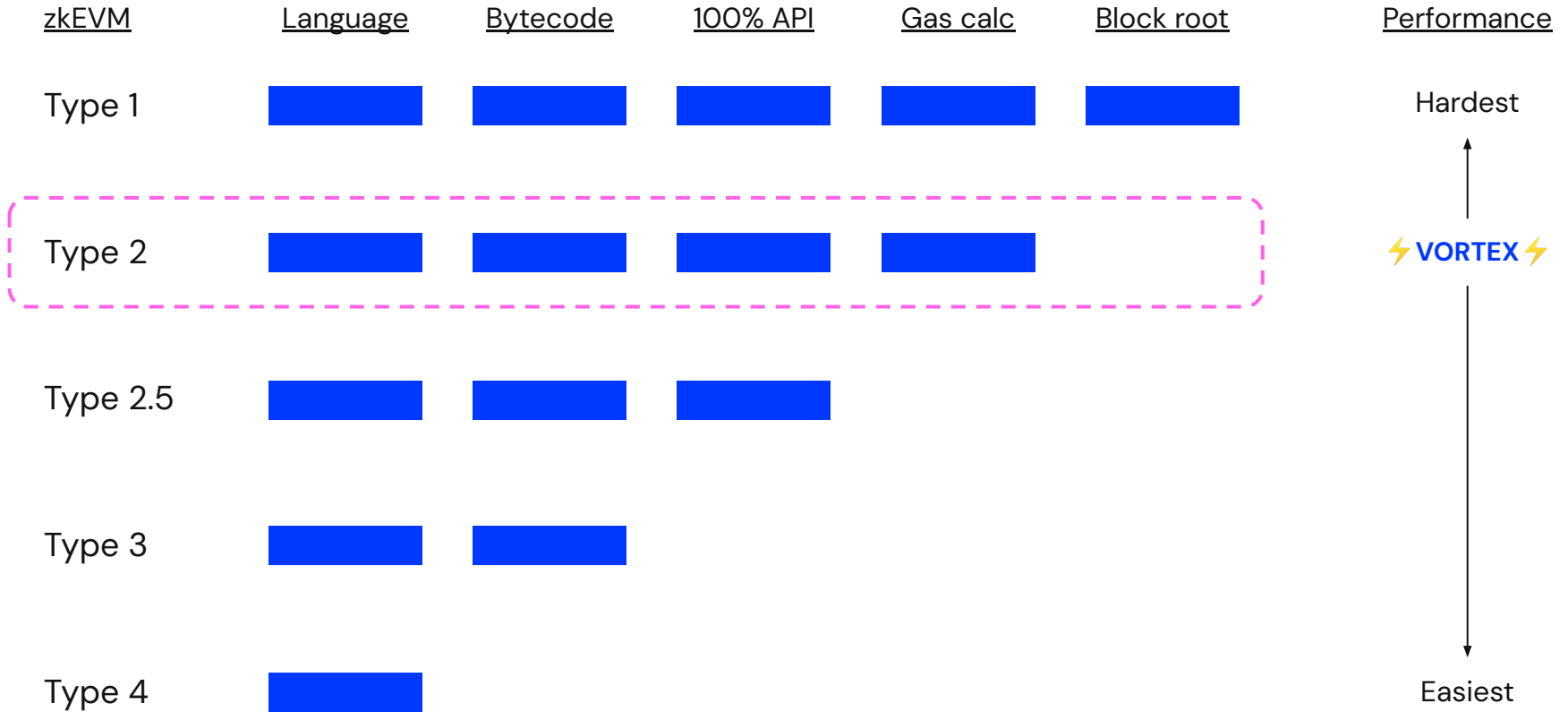
Besu's Execution Context

Mainnet & Private Networks



Linea's Execution Context

Mainnet Future State



# Linea Client (Linea-Besu)

Besu nodes fill two roles within the Linea network...

## Participatory Nodes

---

Sub-Roles:

- Archive Node (PMT, named network)
- RPC Node

Receives updates from the coordinator and other nodes with minimal modifications to Besu to archive data and serve traffic.

```
--network=linea
```

## Operator Nodes

---

Sub-Roles:

- Trace Generation
- State Management
- Sequencer

Besu instances modified with Plug-Ins to operate the Linea network. They help prove traces, store state, and sequence transactions in the mempool.

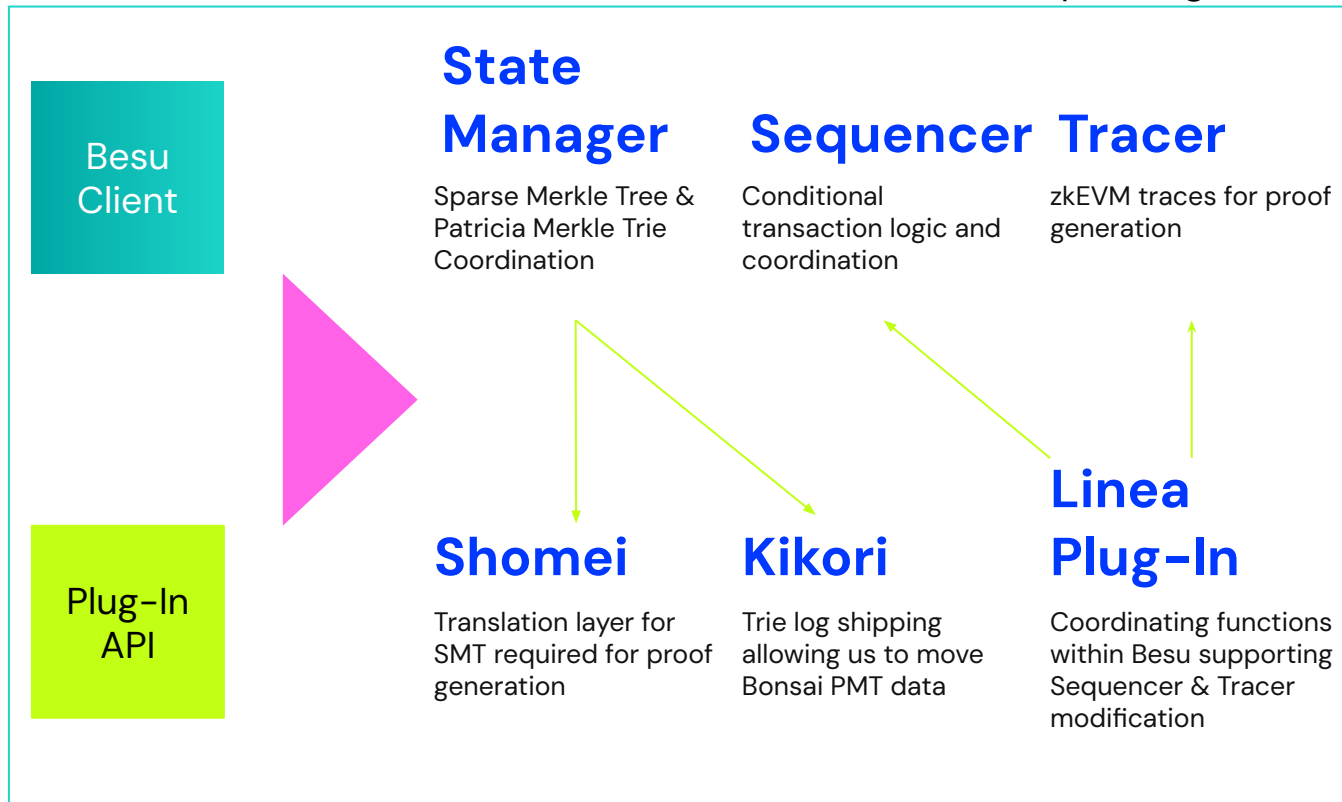
# Component Breakdown

A story of plug-ins

Besu was designed with a plug-in system to extend the client. The plug-in API was first used for integrating different storage backends.

We have since expanded the plug-in API to more areas like the state trie, transaction pool, and are enhancing it further for consensus.

Linea Operating Client

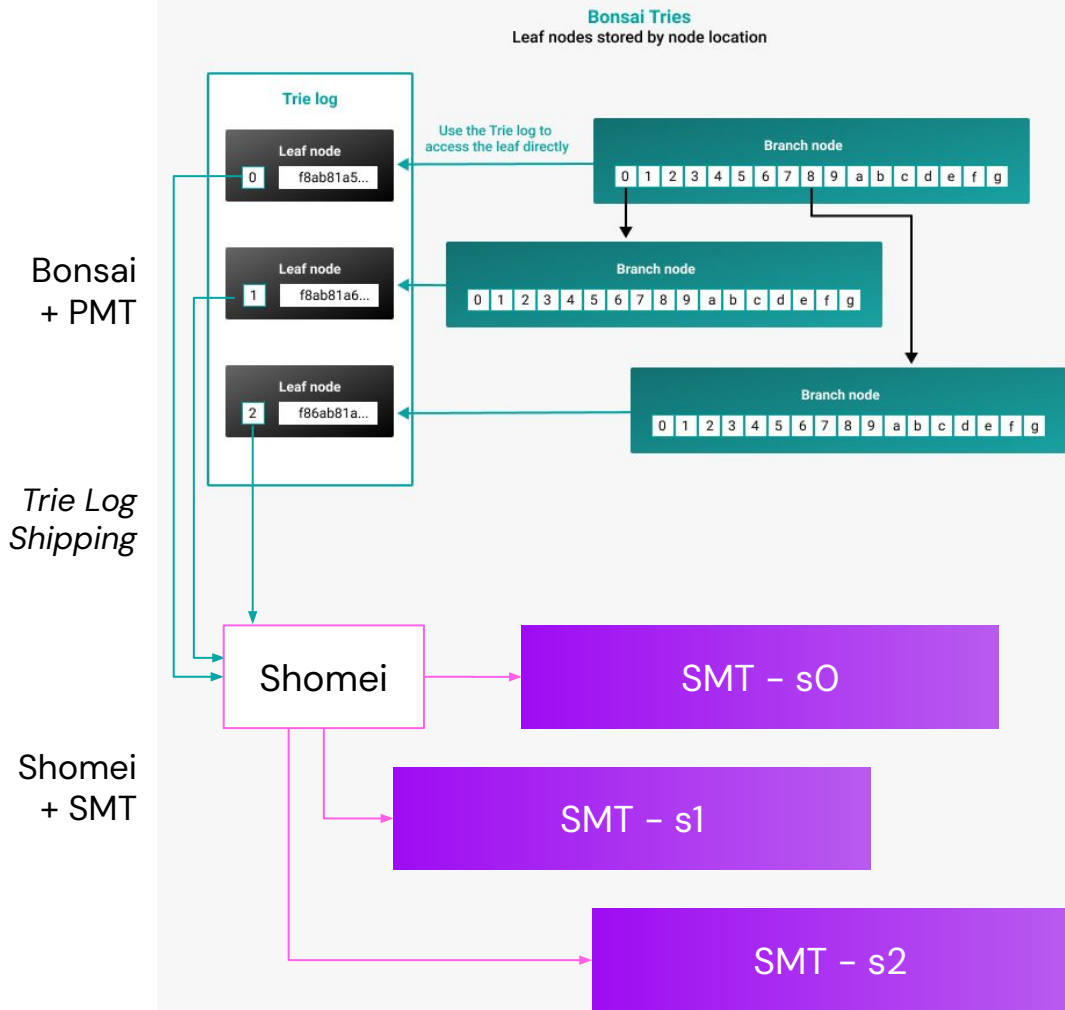


Component Double-click

# State Manager

Besu Main node follows the chain and creates a Bonsai PMT from the zkEVM state transitions.

Besu Main node ships Bonsai trie logs via RPC to Shomei components. Shomei applies Linea SMT specification and modifies a separate tree used in proof generation.

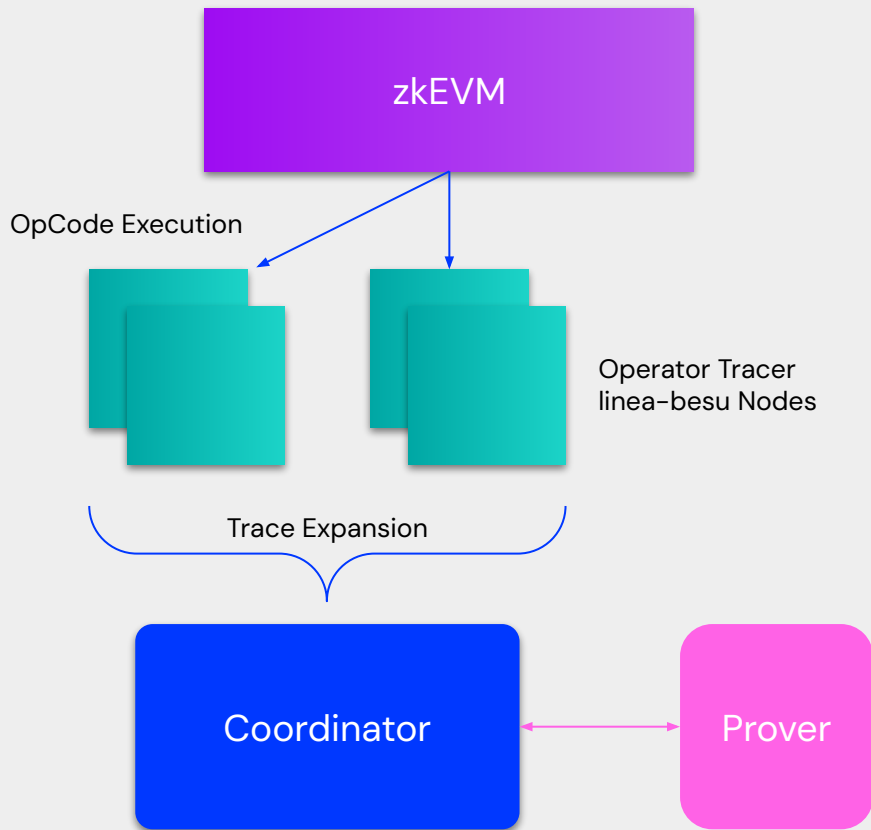


Component Double-click

# Tracer

We expand the EVM execution steps but combining trace calls against the zkEVM, and shipping their results to the prover. The prover checks the results of these RPC calls for accuracy.

A specification of constraints has been defined that allow for the proper expansion of traces and for Besu operator nodes to "check the work" of the zkEVM by providing these traces to the prover.



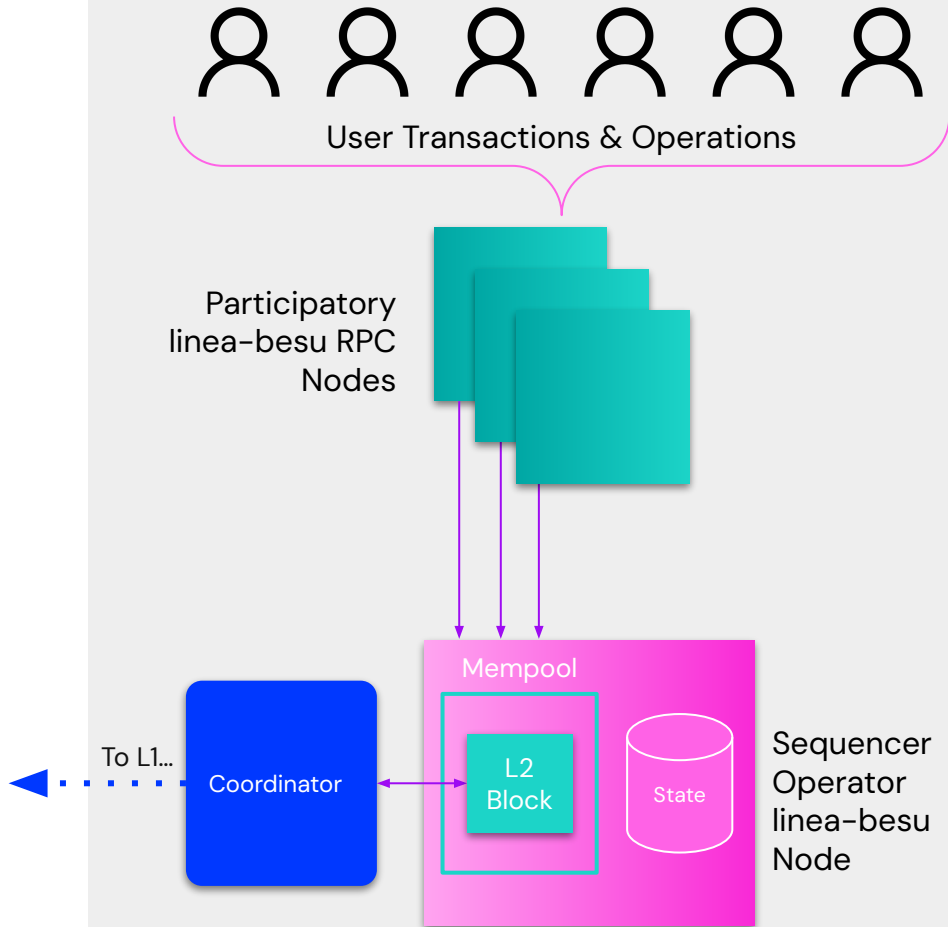


[Component Double-click](#)

# Sequencer

Participatory RPC nodes receive transactions from users to place in the Linea mempool.

The Sequencer, a modified Besu node, uses conditional logic to package transactions into L2 blocks, that are executed elsewhere. The coordinator and sequencer work together to ensure proper logic is applied. In the sequencer and at the RPC layer are where we extend Linea for account abstraction.



# Private Network Evolution

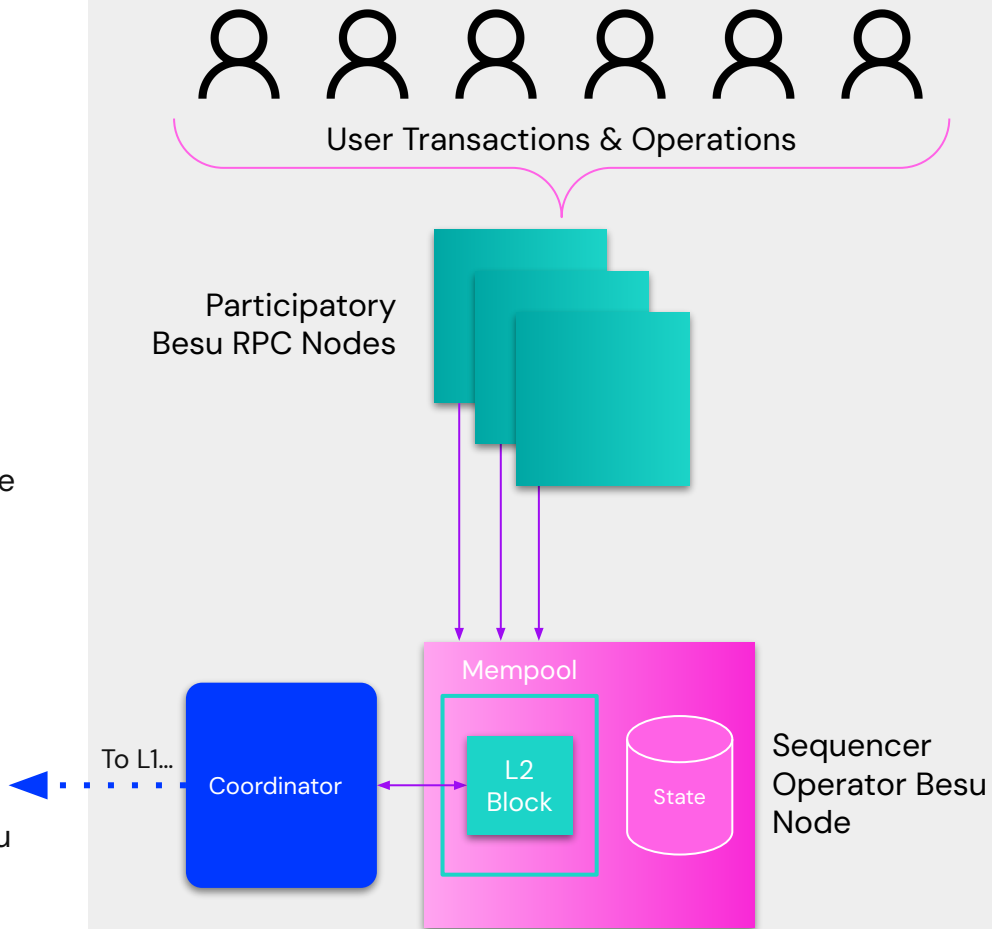
Consortium Evolution

# Decentralized & Single Sequencing

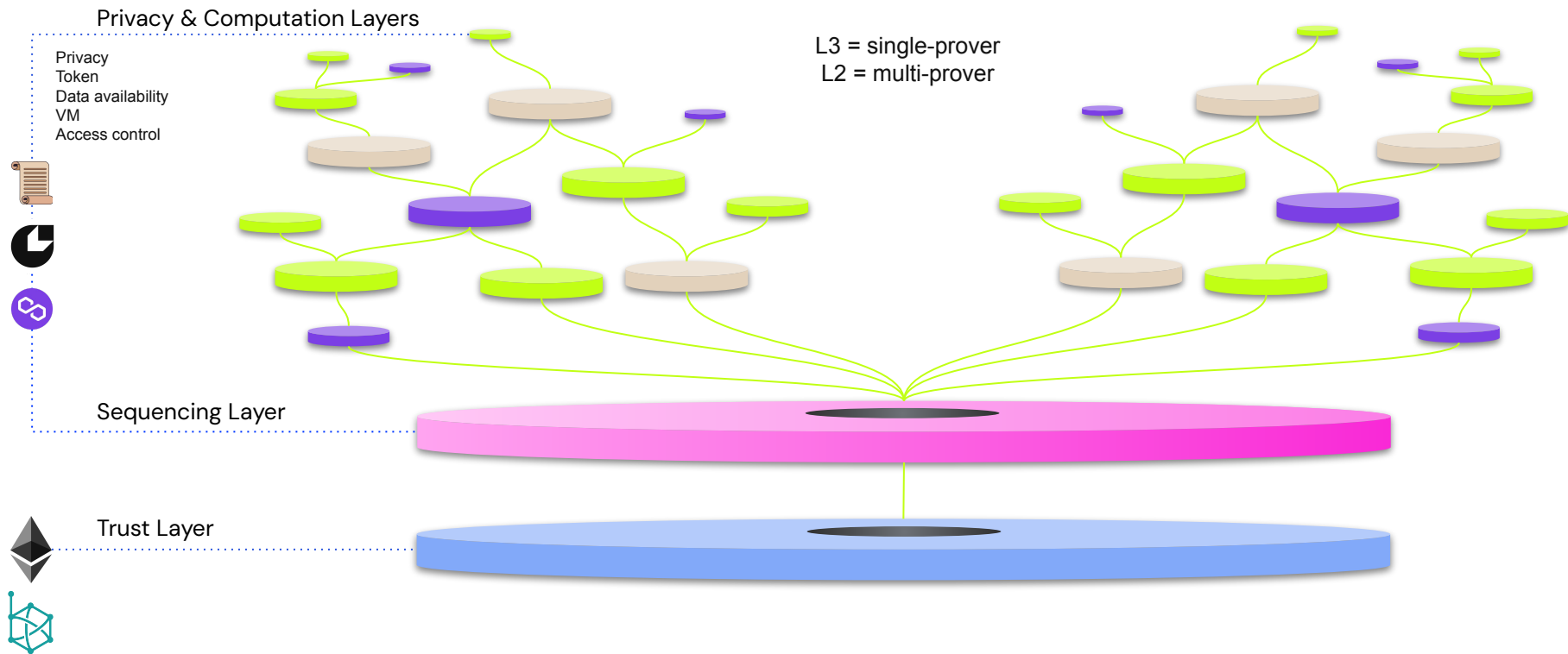
Consortiums exist to agree upon and order transactions on-chain and to validate the incoming and outgoing state of each block.

Besu can serve as the L1 trust model *and* as the L2/L3 sequencing model. These use-cases work together.

Inherit the security properties of whatever trust layer you already use, with the custom privacy and sequencing needed for your org built on top.



## Layer 2 Pattern is Modular & Reusable

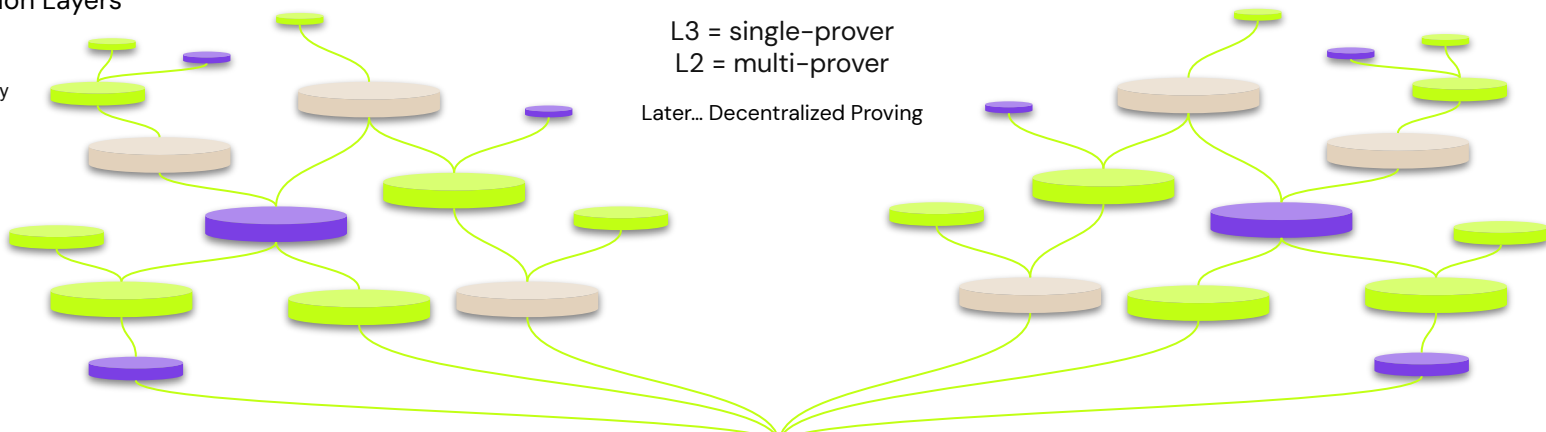



## Consortiums at Scale



### Privacy & Computation Layers

- Org 1  
Privacy  
Token  
Data availability  
VM
- Org 2  
Access control
- Org 3

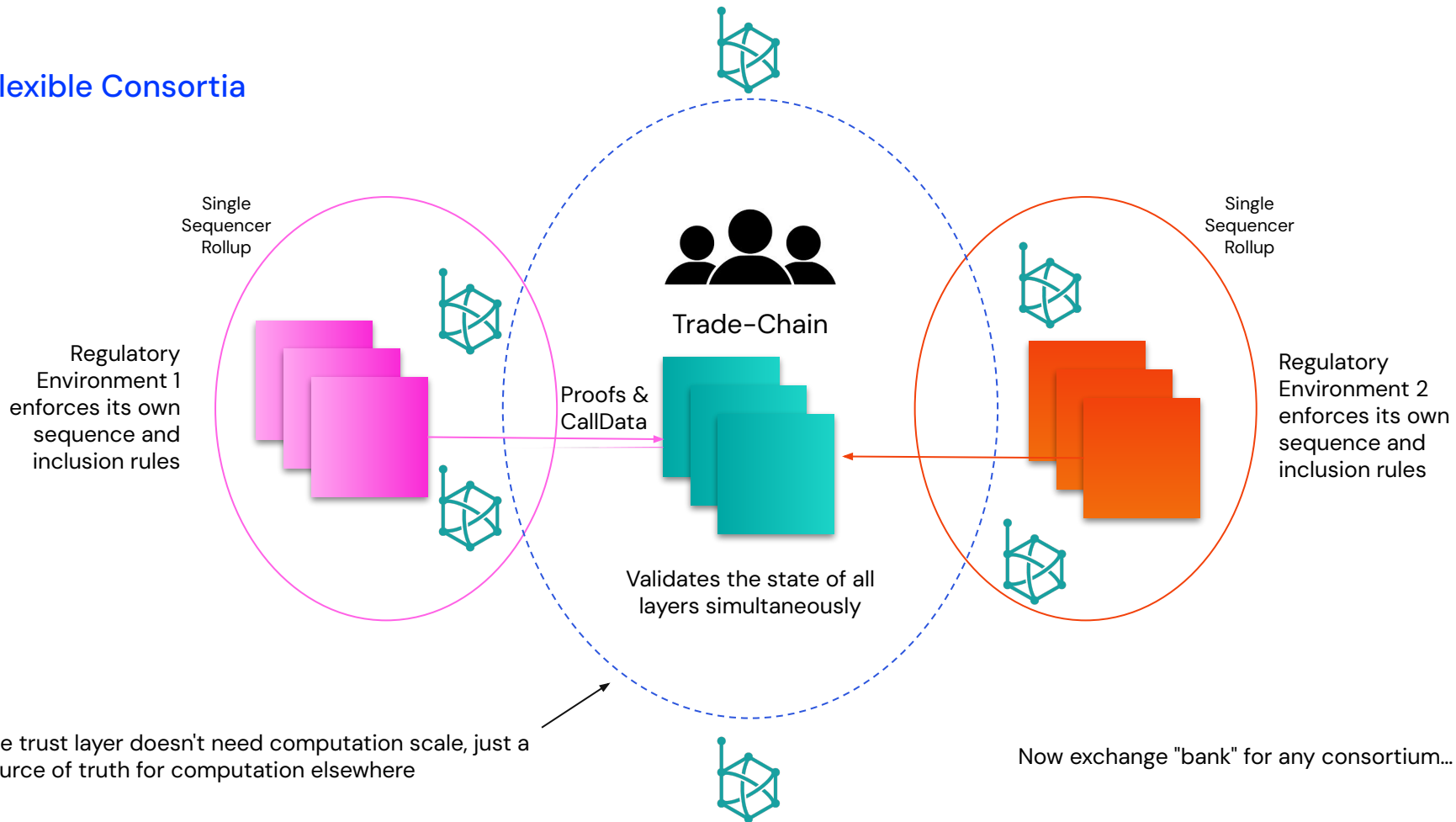
L3 = single-prover  
L2 = multi-prover  
Later... Decentralized Proving



 Sequencing Layer  
For example...

 Trust Layer  
 Existing Besu L1 Consortium  
or trust layer

## Flexible Consortia



## Wrapping Up

- Bonsai everywhere...
  - We want all Besu networks running the most performant technology (Bonsai and snap sync)
  - We will support all use-cases in Bonsai shortly (private networks, archive)
- Besu everywhere...
  - Java-Ethereum at all layers in a format you're familiar with
  - Networks that are portable
- Plug-ins everywhere...
  - Renewed focus on plug-ins next year to avoid costly Besu forks and configurations for your needs
  - Templates and revitalized docs on the way... Linea as the focus
- Ethereum everywhere...
  - Public networks with custom properties will take over the space cheaper and at scale
  - Besu fits into this narrative on whatever trust layer you use
  - EVM Everywhere™

# Thank You!