Move Stack Chains: A Network of High-Throughput Fast-Finality Move-based Rollups Secured by Ethereum Version 0.2.5

Movement Labs

November 21, 2024

Abstract. We introduce Move Stack Chains, a secure and scalable network of Move-based rollups secured by Ethereum, addressing the need for safer execution environments. At its core is the Move Stack, a modular framework for creating highly customizable Move Rollups.

The Movement Network, our flagship general-purpose L2 Move Rollup, showcases the capabilities of the Move Stack:

1. Move Executor: A *high-throughput* execution layer with the MoveVM, parallel execution and EVM compatibility for seamless integration with existing applications.
2. A novel *fast-finality* settlement mechanism, achieving confirmation times in seconds, by leveraging economic security from a network of validators while retaining Ethereum's security.
3. Modularity: Move Stack integrates with multiple DA services and sequencers. Developers can also opt for validity or optimistic rollup configurations to achieve traditional Ethereum security guarantees.

We also introduce the Move Arena, our sophisticated infrastructure that integrates Move Stack Chains with our set of in-house services and that enables an ecosystem of next-generation interoperable rollups. As part of Move Arena, rollups benefit from DSS, our decentralized shared sequencer network, which enables seamless cross-rollup *interoperability*, enhances censorship resistance, and eliminates single points of failure.

DSS is secured through our multi-staking mechanism, which pools economic security across Move Stack Chains and beyond, minimizes the infrastructure requirements and maximises the sovereignty of each Move Rollup.

Table of Contents

Glossary

Move	The programming language and runtime for the MoveVM.
Movement Network	General-Purpose Move-based rollup.
Move Rollup	Blueprint for Move-based rollups.
Move Stack Chains	A network of Move-based rollups.
Move Executor	The module that enables the execution of both MoveVM and EVM bytecode.
Move Stack	The stack of tools, components and adapters that are required to build and deploy custom Move Rollups.
DSS	Decentralized shared sequencer for Move Stack.
Move Arena	A framework that enables Move Rollups to gain access to DSS and the **Staked Settlement** mechanism.

1 Objectives & Motivation

Blockchain technology provides a decentralized ledger where participants can transact without relying on a central authority. The Ethereum Network [5] was the first to propose a versatile *world computer* [12], with *programmable* transactions called *smart contracts*, and the ability to implement arbitrary business logic that go beyond simple currency or assets' transfers (pioneered by the Bitcoin network [11]).

Wide adoption of the Ethereum-based technology is still hindered[1] by several limitations, such as high latency to transaction finality, low throughput (expressed in *transactions per second, TPS*), and widespread security vulnerabilities in *decentralised Applications* (dApps).

Ethereum mainnet, with its unmatched level of *Total Value Locked* (TVL), offers the highest level of crypto-economic security, which creates an unrivaled incentive to capitalise on its best-in-class security guarantees.

Several solutions have been proposed to address the above limitations of the Ethereum network. The most popular ones being *rollups*, which are Layer 2 (L2) solutions that bundle multiple L2-transactions into a single Layer 1 (L1) transaction. Note that we use L1 and Ethereum interchangeably in this paper, however, this is applicable to other L1s as well as L2s. Rollups *settle* transactions on the Ethereum mainnet, thereby inheriting its high level security. Rollups have been successful in addressing some of the scalability limitations of Ethereum, but they have not been able to fully address the security vulnerabilities of dApps, nor the latency issues.

Some of the original design choices of Ethereum, inherited by Ethereum rollups, have made it a very complex infrastructure, making it difficult to address the current limitations. For example, the EVM is not designed to prevent security vulnerabilities[2], unintended assets's duplications or re-entrancy attacks [7,8]. The *global storage* model of the EVM itself makes it hard to parallelise the execution of transactions, which severely limits the scalability of the network. However, the design choices and limitations of the Ethereum network offer a good opportunity to reflect on the current technology and see how to improve it.

Recently new paradigms have emerged for the execution layer, offering new execution environment and programming languages. An example for the latter is Move, originally developed at Facebook (Diem/Libra project), a next generation highly secure and efficient Web3 development platform, providing principled solutions to security vulnerabilities and scalability. It empowers Web3 developers with modern tools to tackle the challenges of deploying reliable, cost-effective

[1] This is also true for many networks like Solana.

[2] According to DefiLlama, hacks have cost more than $680m since the beginning of 2024.

and efficient dApps. Move and the MoveVM are used in L1 chains, such as Aptos [2], Sui [4,13], and 0L [1,10] and has demonstrated very promising results in terms of security, low latency (sub second finality) and throughput (sustained reported throughput of 30k TPS and 160k theoretical, compared to a typical 20 TPS for Ethereum).

The Move language [3] proposes a new approach to Web3 development and was designed to address the current blockchain technology limitations. Move introduces a novel programming paradigm known as *resource-oriented programming*, enabling parallel execution of transactions in the MoveVM, together with strong security guarantees using *formal verification*.

One of the main challenges for the Move community is to build an ecosystem that is crypto-economically secure, but for the time being, the L1 chains Aptos, Sui and 0L have not attained the TVL[3], liquidity and developer activity levels[4] of Ethereum yet. This is a compelling opportunity for our Move community to bring together the highly crypto-economically secure Ethereum platform and Move/MoveVM, the most technologically advanced Web3 development platform.

Our proposal is to build a network of interoperable chains to bridge the gap between two ecosystems, Move and Ethereum, where the most advanced Web3 technology meets the most crypto-economically secure L1 chain.

Our contribution. In Section 2, we introduce Movement Network, a general purpose Move-based rollup. The Movement Network architecture is extracted from the more general Move Rollup blueprint framework, described in Section 3, which is shared by all Move-based rollups in our network. Section 4 describes our novel fast-finality settlement mechanism. In Section 5, we introduce Move Arena the network of Move-based chains, and also, DSS, the shared sequencer that enables cross-chain interoperability.

2 The Movement Network

Movement Network is Movement Labs' general-purpose rollup (Figure 1). It is the first Ethereum L2 that will integrate Celestia for data availability, decentralized shared sequencing, optimistic rollup with option for dual-finality through Fast-Finality Settlement, and the Move Virtual Machine (MoveVM) for execution, which offers unparalleled transaction throughput. This integration will

[3] $60b for Ethereum vs $550m for Sui and $360m for Aptos according to DefiLlama.
[4] More than 13K Solidity devs vs 400 Move devs according to Electrical Capital.

allow developers to create high-performance, consumer-focused applications with minimal resource expenditure.

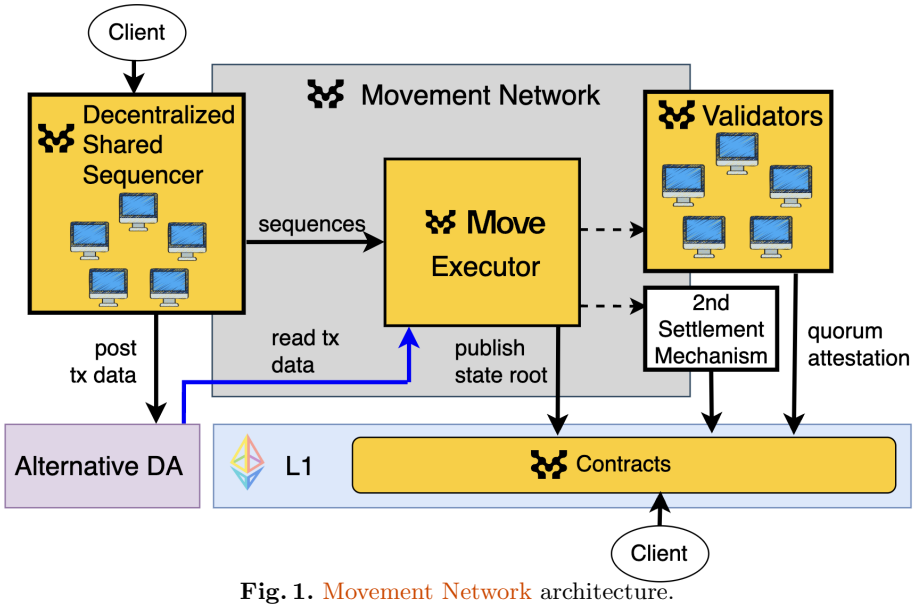

Fig. 1. Movement Network architecture.

2.1 Original components

We develop three original components, that we capitalise on in the Movement Network.

1. The Move Executor (Section 3.2), which supports both MoveVM and EVM transactions, enabling Web3 developers to deploy smart contracts in Move and EVM bytecode on a single network.
2. A *Fast-Finality Settlement* module (Section 4) which connects to a *validator network*, facilitating fast settlement finality when compared to optimistic and validity settlement mechanisms.
3. A *decentralized shared sequencer* module, DSS (Section 5) which ensures customizable transactions ordering, with templates from a set of approaches, such as fair transaction ordering for mitigation of front-running attacks and enhancement of censorship resistance.

First, Movement Network supports both MoveVM and EVM transactions. This is a unique feature of our architecture, as most rollups only support one type of transactions. This feature is critical to allow Web3 developers to onboard the Movement Network quickly. It is also a significant advantage for the Move

Arena (Section 5) as it allows developers to leverage the existing EVM dApps and extend them benefiting from the advanced features of the Move platform. For instance, standard EVM contracts like ERC-20 can be deployed on Movement Network and new and secure Move dApps can be developed to *interoperate* with them.

The Move Executor supports both MoveVM and EVM transactions, allowing Web3 developers to deploy smart contracts in both Move and EVM bytecode on the same network. It provides a unique infrastructure where Web3 developers can migrate or extend their existing EVM dApps with the more secure and efficient Move framework.

Second, we introduce a *fast settlement* mechanism (Section 4), an alternative settlement mechanism to validity and optimistic rollups. Fast-Finality Settlement relies on a set of validators who stake native tokens. The validators have to confirm the correctness of the new L2 state by forming a majority (e.g., 2/3 of the total stakes) to validate the new state.

The fast settlement mechanism offers fast-finality and also contributes to increasing the utility of the Movement Network native token.

Third, by utilizing the DSS sequencer, Movement Network builds on an alternative to sequencing marketplaces, such as Espresso, Astria, or L1-based sequencing. This is a deliberate choice to ensure the sovereignty of Movement Network (and the Move Rollup network more generally) and provide a fast, customizable and verifiable ordering of transactions.

Another consideration is the complexity of (decentralized) shared sequencing marketplaces especially when it comes to distributing rewards and penalties, which are hard problems currently lacking good solutions. A sovereign sequencer module offers a solution where fees can be collected by the L2 rather by an external component (marketplace), thus positively impacting the utility of the native token of the L2. Shared sequencing aims to provide some level of *interoperability* between different rollups and it is discussed in Section 5.2.

The DSS sequencer provides a sovereign, fast, customizable and censorship resistant ordering of transactions, enabling interoperability and increasing the utility of the native token of the Movement Network.

2.2 Original frameworks

We develop two original frameworks, that we capitalise on in Movement Network.

1. The Move Stack (Section 3.3) which enables to create customizable rollups, with the Move *Executor* at the heart.

2. The Move Arena (Section 5) which provides a framework to deploy and join the Move rollups network. A Move Rollup can be configured to connect to the various components of the Move Arena, such as fast settlement or the DSS. This permits it to tap into Move Arena's benefits, which are interoperability with other Move Rollups, Fast-finality and more.

3 The Move Rollup framework and the MoveVM

We introduce the Move Rollup (Figure 2), our general-purpose Move-based rollup schema for Ethereum-secured rollups. Move Rollup is a modular architecture where components can be configured to fulfill customers' needs with the most suitable, cost-efficient and performant components.

In Move Rollup, we offer fraud proof, ZK-proof, and a new *fast-finality* mechanism (Section 4), where a network of *validators* who have staked native tokens, validate the correctness of the new L2 state and the availability of the data, and provide ultra-fast reliable finality with high-economic security.

We show a categorization of Move Rollup configurations in Figure 5 and provide examples in Table 1.

3.1 Architecture of Move Rollup

Move Rollup is a generic architecture for creating *Move-based rollups* that are rollups using the Move Executor (Section 3.2, Figure 2, page 8).

The Move Rollup generic architecture has a set of core components:

- *Executor* to process transactions and generate new L2-blocks.
- *Bridge contracts* on L1 for asset deposits and withdrawals between L1 and the rollup.
- Connection to a *Sequencer* to order transactions.
- Connection to a *Data Availability* (DA) service to ensure transaction data accessibility to the settlement mechanism.
- Connection to a *Settlement Mechanism*: to verify transaction execution correctness.

Move Rollup can be used to create Move-based rollups, for instance the Movement Network (Section 2) is an instance of this generic architecture.

The lifecycle of a transaction within a Move Rollup is as follows:

1. A transaction tx is submitted to the mempool (client, top of Figure 2).
2. The *sequencer* extracts a *batch* b of transactions from the mempool, including tx, and orders them. The sequencer publishes the transactions data of b to the DA service (L1 or an alternative DA).
3. The *executor* processes the transactions. This results in a new L2 state (and a short commitment of it, know as a *state root* s) that is also published to L1 in the bridge contract.

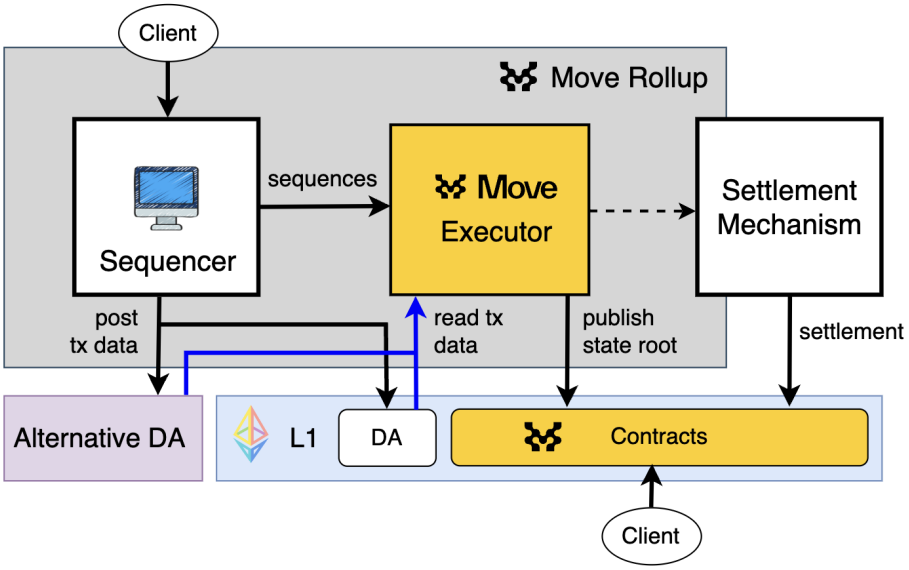

Fig. 2. The generic Move Rollup architecture with the Move Executor. Components in Yellow are fixed in the architecture whereas components in White are customizable.

4. The *settlement* of the transaction *tx* happens when the L1 validating bridge contract verifies or approves the new state. This can be done with ZK-proofs, by passing the challenge period successfully in optimistic rollups, or when the quorum certificate is validated in Fast-Finality Settlement.

3.2 The Move Executor

The execution layer of all Move Rollups, such as Movement Network, is the MoveVM. The Move Stack provides an execution module, Move Executor, that can execute MoveVM bytecode and EVM bytecode. This module is at the heart of our architecture and not configurable.

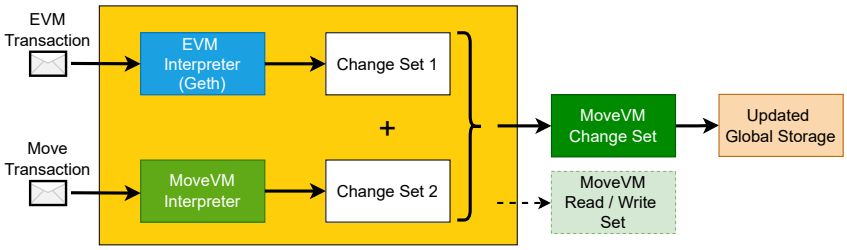

Fig. 3. The Move Executor.

The Move Executor module supports the execution of both MoveVM and EVM bytecode on the same chain.

Figure 3 gives a high-level view of the Move Executor module. Transactions are triaged from the mempool according to their types, Move or EVM. A corresponding VM (MoveVM or Geth) executes a transaction. In MoveVM this results in a *change set* that is later applied to the global storage. In Geth a transaction can be executed and modify the global storage, but it can also be *traced* instead to produce a change set that can be applied to the global storage. This provides a way to get a common format for the update of the global storage by both Move and Ethereum transactions. Another nice feature is that read/writes sets can also be extracted with Geth and can be used seamlessly in *BlockSTM*, the MoveVM's built-in parallel execution engine.

The Move Executor re-uses existing EVM interpreters and integrates seamlessly with MoveVM to benefit from its parallel execution engine, thereby providing a *parallel* EVM. Moreover, using an existing EVM interpreter under the hood ensures that Move Rollup is EVM-equivalent and that executed EVM bytecode executed has the same behaviour as on L1.

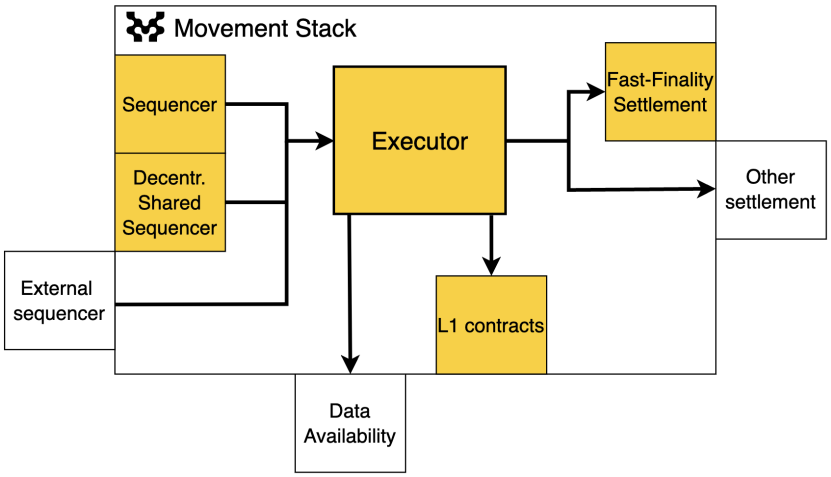

Fig. 4. The Move Stack provides a set of components (Yellow boxes) along with adaptors (White boxes). To create a rollup instance from the Move Rollup blueprint a component is selected (i.e. configured) from the available options.

3.3 Move Stack

The Move Stack provides support to create and deploy Move Rollups. Developers are empowered to quickly spin up new Move Rollups in the network, by selecting suitable components from the provided options in the Move Stack, see Figure 4.

The (configurable) components of the Move Stack include:

- *Sequencer:* A rollup can opt-in for the default DSS, decentralized shared sequencing service, but is provided with a default self-reliant sequencing mechanism.
- *Data Availability:* We plan to support Ethereum EIP-4844 blobs, and major DA solutions (e.g., 0G, Avail, Celestia, EigenDA, Near).
- *Settlement mechanisms:* Optimistic (fraud proof), ZK (validity proof), Fast-finality (attestations).

Rollup	Type	Sequencer	DA	Settlement
Movement Network	General Purpose	DSS	Celestia	Optimistic & Fast-finality rollup
Gamechain	Gaming	Centralized	EigenDA	Optimistic rollup
Finchain	DeFi	DSS	Ethereum	Fast-finality rollup
Tokenchain	DeFi	DSS	Near	ZK rollup
Duckchain	Art	Centralized	EigenDA	Optimistic rollup

Table 1. Example of Move Rollup configurations

Table 1 illustrates a diverse range of Move Rollups within the Move Arena (Section 5), showcasing the extensive customization possibilities based on specific requirements. Moreover, using Move Stack to deploy a Move Rollup promotes standardization across crucial infrastructure components, including wallet software, developer APIs, and block explorers. This standardization enhances interoperability and significantly improves the developer and user experience across the Move Arena ecosystem.

4 Fast-Finality Settlement

The modularity of the Move Rollup framework enables that the chain can be secured through a novel *staking* mechanism. This staking mechanism provides *fast-finality*[5] with high crypto-economic security.

[5] *Finality* is the time for a transaction to be confirmed and become practically irreversible.

Since our Fast-Finality Settlement approach is similar to how Ethereum works in terms of security, we will recall basic concepts relevant to this context, specifically Ethereum's security model (Section 4.1) and the security of ZK and optimistic rollups (Section 4.2). We then introduce the concept of Post-confirmations, a simple mechanism for implementing Fast-Finality Settlement (Section 4.3). We compare the level of security provided by Fast-Finality Settlement to optimistic and validity rollups, see also Figure 5. Finally, we propose a combination of traditional rollup settlements with the Fast-Finality Settlement, called dual-finality, to provide both Ethereum security and economically protected fast-finality guarantees (Section 4.5).

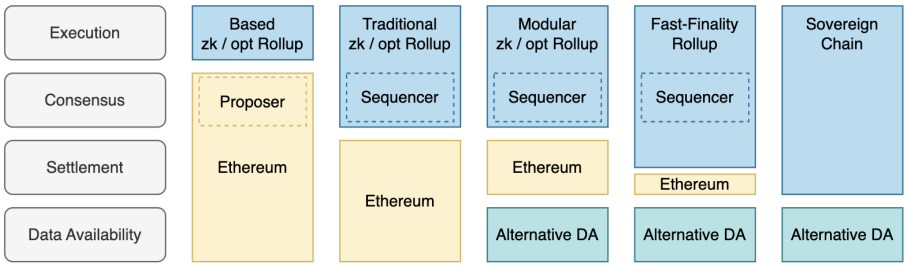

Fig. 5. Categorisation of rollups. Move Rollups are configurable and can take on any of the above forms. The Movement Network is a dual-finality rollup with optimistic approach and Fast-Finality Settlement.

4.1 Ethereum settlement and security

Ethereum's consensus is a *proof of stake* (PoS) protocol, and validators have to stake some assets (32 ETH) to be incentivized to attest honestly about the status of a state transition. A validator that would be Byzantine (malicious) bears the risk[6] of being slashed of their stake. On Ethereum mainnet (L1), a state transition (creation of a new block) is *final* once it has received *enough attestations* from the validators. Enough stake is usually understood as 2/3 of the total stake – *a super-majority* – of all the validators. As a result, under the assumption that less than 1/3 of the validators are malicious,[7] if more than 2/3 of the validators have attested for a state transition, it must be correct as at least one the validators in this 2/3 is not Byzantine (it is honest).
The *security* provided by the PoS mechanism is two-fold:

- *liveness*: in order to prevent a super-majority to attest a correct state transition, an adversary would have to control more than 1/3 of validators. This

[6] There are two slashing conditions on Ethereum, and if a validator is caught violating one of them, they can be slashed.
[7] And each validator stakes the same amount.

is considered *infeasible* when the total stake in the system is large (the probability that this happens is negligible.)

- *safety*: in order to force an incorrect state transition (e.g., double spending) an attacker would need to control 2/3 of the validators. Similarly to the previous point, this is considered infeasible given a large enough stake.

Ethereum security: The *level of security*, i.e., the liveness of the Ethereum network and the safety (correctness of a state transition), increases with the total stake in the system. The higher the total skake, the more secure the network is. The level of security provided by the Ethereum network is commonly referred to as *Ethereum security*.

4.2 Security of validity and optimistic rollups

There are two main types of rollups, *validity (ZK) rollups* and *optimistic rollups*. Both settle on a Layer 1 (e.g., Ethereum mainnet) but use different settlement mechanisms.

In a ZK-rollup, settlement happens when the ZK-proof of the state transition is *accepted*. This is done by submitting a *verification* transaction to the L1 *verifier* contract. Since the verifier is implemented as a contract on L1, the security level of the verification phase is Ethereum's security. Under the assumption that the ZK-proof system (proof generation and verifier contract) is correct, the ZK-proof is accepted *if and only if* the state transition is correct, hence

ZK-rollup security: The level of security of a ZK-rollup is the same as Ethereum's security: a ZK-rollup *inherits* Ethereum's security.

In an optimistic rollup, finality of transactions – after submitting the data and state commitments to Layer 1 – is achieved at the end of a time window, called the *challenge period*. It follows that security is *conditional*: the settlement happens if at the end of the challenge period (usually 7 days), no *disputes* have been *successful*. A dispute is a way of challenging a state transition. Validators can *raise* a dispute against a state transition if they think that it has been computed incorrectly. A *trusted dispute resolution* mechanism resolves the challenge: if the challenge is successful, the submitter of the incorrect state transition is slashed. Otherwise the challenging validator is slashed. Assuming at least one honest validator (e.g., *watchtower*) re-executes each L2 state transition, it is impractical for the L2 to submit an incorrect new state. The level of security depends on *where* the dispute is settled. If it settles on Ethereum mainnet via a contract (e.g., as in [14]), and the contract that resolves the dispute is trusted (no bugs) then the security of the dispute resolution is Ethereum's security.

Optimistic rollup security: The security of the dispute resolution mechanism of an optimistic rollup can inherit Ethereum's security. But if no validator checks a state transition before the end of the challenge period, then the level of security is zero.

Finality of a state transition (i.e., finality of a transaction) on Ethereum mainnet is in the order of 12 minutes. On average the time to generate a ZK-proof is in the order of 10-15 minutes, and hence the finality of a transaction on a ZK-rollup is expected to be in the order of 20-25 minutes. For optimistic rollups, the standard challenging period is 1 week. In both cases, the time it takes to finalize a transaction can be prohibitively large for some, if not many applications.

4.3 Security of Fast-Finality Settlement

As discussed in the previous sections, validity (zero-knowledge proof, ZKP) and optimistic (fraud-proof, FP) rollups can finalize transactions with Ethereum security within approximately 30 minutes and 1 week, respectively. However, until a transaction is finalized, assurance about its validity and result (success or failure) is limited. This can be a limiting factor for many types of DeFi applications. An intermediate level of economic security but with fast-finality guarantees, can be provided via Fast-Finality Settlement.

Fast-Finality Settlement provides security through a Proof of Stake (PoS) protocol. In a PoS protocol, validators stake some assets (e.g., in native L2 tokens) to be incentivized to attest honestly about the status of an L2 state transition. If they are dishonest (they accept incorrect state transitions or reject correct state transitions) their stakes can be slashed. If they are honest validators, they are rewarded for their activity. A network of *validators* can then provide fast and economically backed confirmations of correctly executed blocks. More precisely, the role of a validator is to confirm that the execution of a state transition is correct.[8]

A state transition (that corresponds to the execution of a set of transactions) is *L2-final* (irreversible) on a Move Rollup that utilizes *only* Fast-Finality Settlement when *enough* validators have confirmed the correctness of the state transition. For the sake of simplicity, we assume all the validators stake the same amount and *enough* means more than 2/3 of the validators. The entire stake value is called *L2-Stake*. Figure 6 illustrates the process of Fast-Finality Settlement, and the time to (L1-/L2-)finality of a transaction.

Overall, the user can now obtain quickly guarantees about the result of their transactions, and can decide whether this is good enough to assume irreversibility of the transaction or to wait for L1 finality (with a ZKP or a FP) to get Ethereum's level of security.

[8] w.r.t. the semantics of the execution layer i.e. MoveVM.

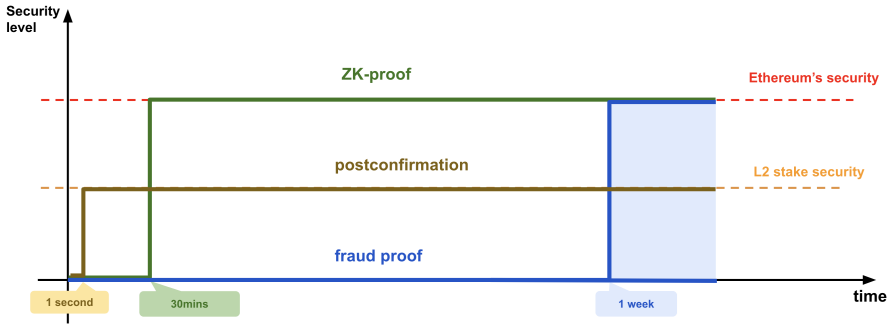

Fig. 6. Security levels and time to finality. The time to finality does not include L1 (Ethereum) average finalization time which is 13 minutes. Times displayed are indicative and may vary.

Comparison with optimistic and ZK-rollups. The security of an optimistic rollup inherits Ethereum's security *under the condition* that an honest validator raises a dispute for each incorrect state transition. However, at present, optimistic rollups limit the list of challengers to reduce the risk of delay attacks in which an adversary could open as many disputes as they are willing to forfeit bonds for. [9] All of the above impose significant trust assumptions onto the user. Moreover, slashing penalties may not be economically large, compared to total stake protection as is the case in the validator network.

In contrast to ZK-rollups, a Move Rollup that employs only Fast-Finality Settlement does not require expensive proof generation equipment. However, the most significant improvement delivered by Fast-Finality Settlement is the reduction in latency compared to both optimistic and ZK-rollups. Since attestations can be delivered in the order of seconds, we can provide fast-finality guarantees and substantially improve user experience. This compares to order of minutes in the ZK-rollup setting and days in the optimistic setting.

The Fast-Finality Settlement is instrumental to interoperability and atomic cross-rollups transactions where a fast settlement time is required. Both optimistic and ZK-rollups lack in that respect. Hopefully, ZK-proof technology that permits real-time proving with specialised hardware, will be widely available in the near future, however, it is not clear at what point in time this is the case. Regarding optimistic rollups, they have an inherent requirement of extensive challenge periods (up to a week) to account for social engineering and attack vectors. In contrast, fast-finsality rollups can provide finality guarantees within seconds.

[9] https://docs.arbitrum.io/how-arbitrum-works/bold/gentle-introduction. Accessed on 2024-07-10.

A Fast-finality Move Rollup rollup can be more secure than an optimistic rollup and has faster finality than a ZK-rollup. If the total stake of the validator network is greater or equal to Ethereum validators' total stake, then the Fast-finality Move Rollup rollup even reaches Ethereum's economic security level. The overall security of the Fast-Finality Settlement approach depends on the total stake of validators. The staking, rewarding and verification steps inherit Ethereum security.

We discuss a generalization of the staking mechanism, multi-asset staking, in Section 5.3.

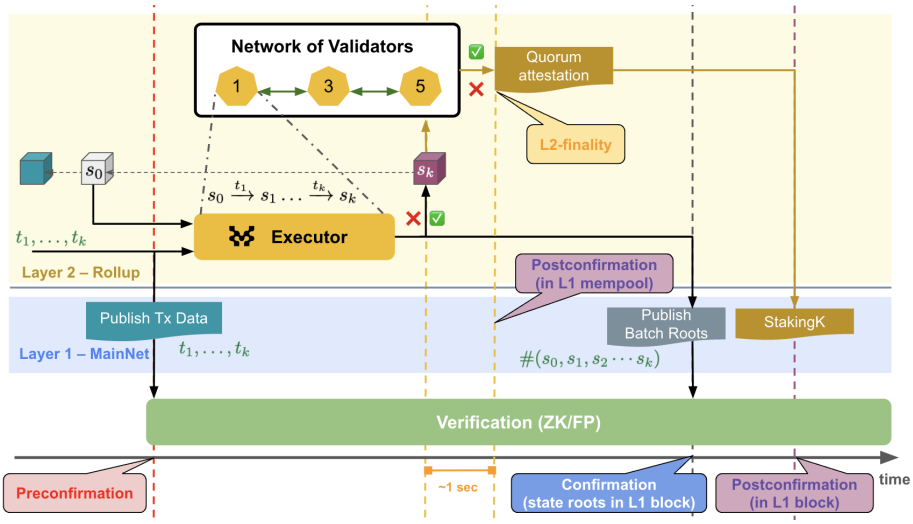

Fig. 7. Confirmation stages of a transaction. Preconfirmation are promises that a transaction will be included (or even executed) in the next block(s). In contrast, Postconfirmations offer guarantees, backed economically by the L2 stake of validators, that the new state (after the executor has processed a transaction) is correct.

4.4 Postconfirmation

Postconfirmation is an implementation of Fast-Finality Settlement.

First we provide a defintion of *confirmation* on L1, that differs to that of L1-finalization. (We do not deem it useful to define L1-confirmation the same as L1-finality). We say a transaction is *confirmed* on L1, when the state has a committment in an L1 block. We note that a confirmed transaction can still be reverted if the block that contains the transaction is orphaned.

Postconfirmations are different to preconfirmations. They provide a guarantee that the new block is correct, not only that a transaction will be included

(or executed). It is also not a replacement for the complex execution tickets mechanism, as it does not provide a way to influence the creation of a block, but rather to report on the correct execution of the transactions in a block. In Figure 7 we depict the confirmation stages of a transaction.

Postconfirmations are fast because they are delivered right after the execution of a block of transactions when a new block is comitted, see Figure 7. Postconfirmations themselves may be confirmed (i.e. included in a L1 block) together with confirmation of the state.

The mechanism is as follows:

- A set of validators stake some assets in a *trusted* L1 contract `StakingK`.
- For a given state transition of a Move Rollup, the validators broadcast their signed attestations (either approving or rejecting), and at the same time collect the signed attestations from others.
- When an validator has collected attestations representing more than 2/3 of the total stake, they submit them to contract `StakingK`.
- The contract `StakingK` verifies that the attestation signatures are valid, unique, and account for more than 2/3 of the stake. The state transition becomes final.

Due to cryptographically protected signatures, a Byzantine validator cannot forge/tamper with the signed attestations. Assuming less than 1/3 of the validators are Byzantine, and due to the 2/3 majority requirement, there cannot be a malicious actor who could submit enough attestations supporting an erroneous state transition. Moreover, during synchronous periods liveness is preserved, as 2/3 of stake are also sufficient to complete the attestation process.

The verification that the threshold of 2/3 of the validators have confirmed a state transition is performed by contract `StakingK`. As a result the verification step inherits Ethereum security. The staking/slashing/rewarding functions are also executed on L1 with the same security level.

Aggregating Attestations. To make the attestation process more efficient, we can require the validators to run an L1 light client. They have access to the state of `StakingK` and can determine themselves how many validators are *active*. An active validator is a validator that has not been slashed of their stakes.

The validators can broadcast their votes for a new block to the entire validators' network. The validators can record and aggregate signed attestations. When one of the validators has determined that the 2/3 super-majority is reached, they can send the aggregated (signed) attestations to the `StakingK` contract. This reduces the number of L1 transactions needed to record the attestations.

We should also note that the validators record both positive and negative attestations and send both types of attestations to the `StakingK`. Once a 2/3 super-majority is reached, the `StakingK` contract can slash the validators that attested negatively, as they are dishonest (under the assumption that at most 1/3 of validators are Byzantine).

As validators rely on a recent state of `StakingK` contract (to compute what the 2/3 majority threshold is), we have to prevent validators from withdrawing their stakes too quickly. This is to ensure that a dishonest validator cannot wrongly/dishonestly attest and then withdraw their stakes before being slashed. We can lock the stakes for a pre-defined amount of time (a few epochs).

It does not matter whether an honest or dishonest validator posts the aggregated signatures. Assuming signatures cannot be tampered with, a dishonest validator can only withhold some signatures or not posting anything to the L1, but cannot forge an invalid set of attestations.

4.5 Dual-finality

While Fast-Finality Settlement provides a rapid and economically robust form of transaction finality, it can be further strengthened by integrating it with the proven security guarantees of optimistic and ZK-rollups. By layering these approaches, we can offer a dual-layer security model (i.e. *Dual-finality*) that leverages the strengths of both systems.

In this combined approach, Fast-Finality Settlement delivers a finality *level* that is backed by the economic security of staked validators, ensuring a rapid confirmation of transactions. The system can also invoke the traditional finality mechanisms of optimistic or ZK-rollups, which provide the additional security benefits derived from Ethereum's mainnet, albeit with the typical latency associated with these methods.

This dual-layered finality model operates as follows:

- **Fast-Finality Layer**: Validators within the Fast-Finality Settlement framework rapidly confirm the correctness of state transitions, providing an initial layer of finality that is economically secure and swift, enhancing user experience by reducing waiting times.
- **Optimistic/ZK-Finality Layer**: After the Fast-finality is established, the transaction data is also processed through a secondary finality mechanism —either optimistic or ZK-rollup— on the Ethereum mainnet. This ensures that even if the Fast-Finality mechanism is compromised (e.g., due to a significant, albeit improbable, collusion of validators), the transaction still benefits from Ethereum's robust security guarantees.

By combining these mechanisms, the system offers a highly secure and efficient transaction finality process:

- **Improved User Experience**: Users benefit from the fast and economically secure finality provided by the Fast-Finality Settlement, without sacrificing the long-term security provided by Ethereum's settlement layer.
- **Flexibility and Resilience**: This approach allows the system to adapt to various security needs, offering a balanced trade-off between speed and security based on the specific requirements of different applications.
- **Enhanced Security**: The integration of two independent finality mechanisms significantly reduces the probability of a successful attack, as an

adversary would need to compromise both the validator network and the secondary finality process.

In conclusion, this dual-layered finality approach provides the best of both worlds: the swift and economically backed assurances of Fast-Finality Settlement, combined with the well-established security of optimistic or ZK-rollups. This hybrid model is particularly advantageous for applications requiring both immediate transaction confirmation and the highest possible security standards.

5 The Move Arena

The Move Arena is an advanced blockchain infrastructure designed to seamlessly integrate with our suite of in-house services and support a network of interoperable rollups. This infrastructure facilitates the creation of a dynamic ecosystem where various rollups can operate efficiently and interact with one another. Its designed to meet the diverse needs of modern blockchain applications, offering enhanced interoperability, security, and resource efficiency.
Move Arena is built upon several core components that enhance its functionality and interoperability, see Figure 8.

- Move Stack Chains: A framework for deploying and managing application-specific rollups.
- DSS: A decentralized shared sequencer network that ensures seamless cross-rollup interoperability and enhances network security.
- **Validator Network:** A Proof-of-Stake based attestation system to ensure Fast-finality and strong economic security for Fast-Finality Settlement, see Section 4.
- **Multi-Asset Staking:** Allows stakers to use multiple assets for staking, increasing flexibility and economic security.

In the following sections we explore the concepts of Move Stack Chains, DSS and the Validator Network.

5.1 Move Stack Chains: a network of application-specific chains

Application-specific chains are becoming the norm in the blockchain world. This is driven by the fact that applications like DeFi, gaming or supply chain applications have different requirements for latency and throughput. Privacy or proprietary requirements may also need to isolate a chain and its dApps from others. As a result, app-specific chains are proliferating across L1 networks like Avalanche, Cosmos, and Polkadot.

We can take advantage of the modularity of our architecture (Section 3) to cater for specific needs, while at the same time providing cross-chain interoperability and shared liquidity. This is achieved by creating a network of Move Rollups, called Move Stack Chains, and integrating this network into our novel platform, the Move Arena (Figure 8). By sharing the same modular architecture,

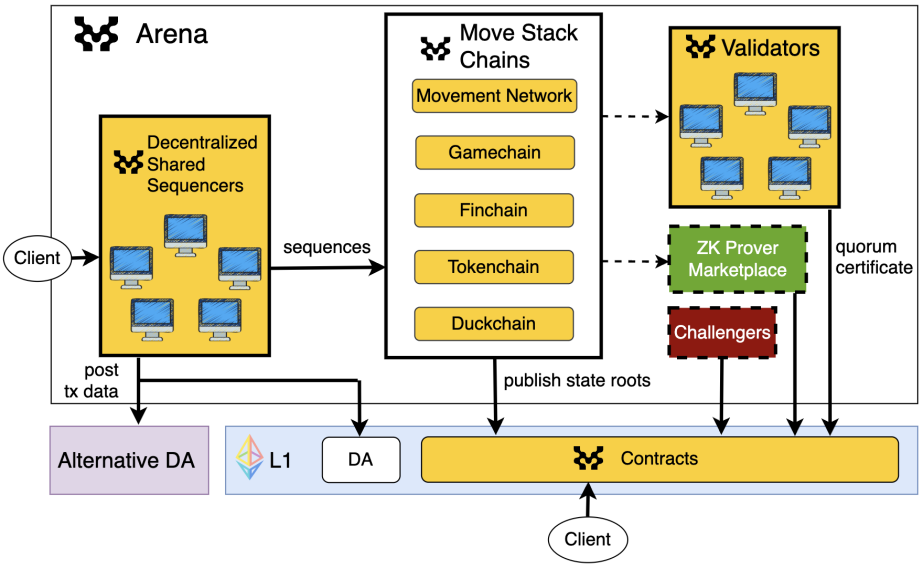

Fig. 8. The Move Arena: an unique blockchain infrastructure to cater for Move Stack Chains - a network of interoperable Move Rollups.

the chains in Move Stack Chains are equipped with increased interoperability, can share the same bridge and Data Availability layer, and can profit from the fast settlement supplied in Move Arena.

This design choice is consistent with the other L2 ecosystems, including Optimism Superchain, Arbitrum Orbit, Polygon Supernets, zkSync Elastic Chain or Starknet appchains (layer 3).

The Move Arena offers a cost-efficient and secure way of deploying new application-specific Move Rollups. Moreover, by being part of the Move Stack Chains (rollup network) these chains are provisioned with cross-chain interoperability, as well as shared liquidity between them.

In the sequel we outline the key features of our Move Arena. We discuss DSS (Section 5.2), our innovative shared sequencer solution and the concept of *multi-staking* (Section 5.3), which enhances the security and economic efficiency of our network.

5.2 DSS: Decentralized Shared Sequencer

DSS serves as a decentralized and shared sequencer for the Move Arena, diverging from the centralized sequencers commonly used in most rollups. This decentralized design enhances network robustness by eliminating single points

of failure, promotes fairness and censorship resistance in transaction ordering, and allows permissionless participation [9].

To achieve consensus on transactions ordering, we employ a highly scalable, performant Byzantine Fault Tolerant (BFT) protocol.

Centralized sequencers may offer faster preconfirmations than decentralized sequencers due to their centralized nature. However, being built with a highly scalable BFT consensus mechanisms and an efficient mempool mechanism, DSS can provide fast preconfirmations with only marginal increase in times, with the additional benefit of being economically backed, rather than based on trust. In regards to throughput a centralised sequencer could have less throughput limitations, however significant advances have been made on modern BFT protocols and which enable throughput levels that more than satisfy requirements. Finally, users may value interoperability features over latency questions.

A distinguishing feature of DSS is its *shared* architecture across all Move Rollups. This shared sequencer approach is pivotal in enabling seamless interoperability within the Move Arena ecosystem. By utilizing a common sequencing layer, DSS facilitates cross-chain atomic swaps and pooled liquidity across Move Rollups, significantly enhancing the network's overall security, functionality and efficiency.

The sequencers are responsible for posting transaction data to the DA service chosen by each rollup. To mitigate data withholding attacks, we implement slashing mechanisms for non-compliant sequencers. While DSS manages transaction ordering consensus, the Move Executor, powered by MoveVM, handles transaction execution. This separation of concerns optimizes network efficiency and security, laying the foundations for future innovations such as privacy enhancements or opt-in censorship capabilities.

Unlike other shared sequencer solutions, our stewardship of both the DSS and the Move Rollup framework allows for deeper integration and optimization.

Our shared infrastructure approach not only reduces infrastructure burden for individual rollups but also creates a unified ecosystem where assets and liquidity can flow freely between Move Rollups, enhancing overall user experience and network utility. The result is a highly interoperable and scalable L2 solution that combines the benefits of MoveVM.

5.3 Multi-asset staking

Our DSS decentralized shared sequencer uses a Proof of Stake (PoS) mechanism. The Fast-Finality Settlement also uses PoS to incentivize validators to be honest when attesting for new blocks. PoS, proven effective in ecosystems like Ethereum, requires candidates to stake native tokens, demonstrating commitment and adding capability to resist attacks. *Single-asset* staking requires stakers to stake in a fixed crypto currency which means that they may have

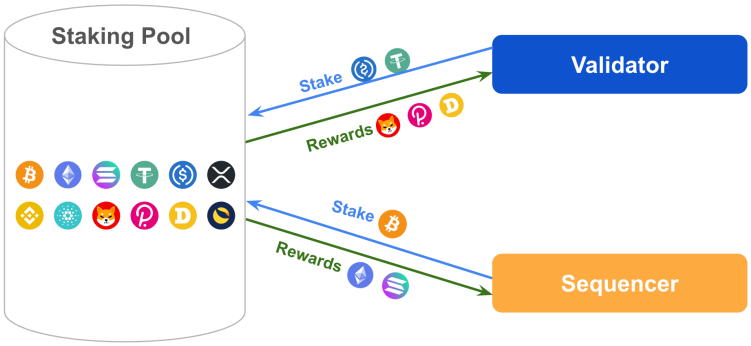

Fig. 9. Multi-asset staking.

to swap assets before staking if they don't hold the token used in the staking protocol. This can be hurdle for stakers. This is why we will enable *multi-asset* staking, which is PoS that allows stakers to stake and get rewards in multiple assets (Figure 9, Image by myriammir on Freepik).

> This is why we will enable *multi-asset* staking, which is PoS that allows stakers to stake and get rewards in multiple assets.

Multi-asset staking is convenient for stakers but comes with some challenges for the operator of the network:

- the staking pool is composed of several assets the prices of which may fluctuate,
- a PoS protocol usually relies on a *super-majority* of 2/3 of the total stake to finalize a decision (an ordering in the sequencer, a confirmation of a new block through the Fast-Finality Settlement).
- as a result of the previous two points, some stakers may obtain unreasonable power and a small fraction of them may control the 2/3 super-majority, which negatively impact crypto security.

One solution to mitigate the problem is to use a (staking) *pool token*. Stakers stake arbitrary assets and are awarded pool tokens. When new stakers stake (resp. unstake) some assets, pool tokens can be minted (resp. burnt) and some re-balancing strategies [6] and liquidity curves choices have to be applied to manage the staking pool.

The implementation of secure strategies that protect our stakers (e.g. from impermanent loss) is an active research topic.

A critical feature in our multi-staking approach is the ability to stake without operating a node. This mechanism, called *Delegation*, maximises the amount of staked value and, therefore, boosts the economic security substantially.

References

1. 0L: 0l network. https://0l.network/, accessed: 2024-01-12
2. Aptos: https://aptosfoundation.org/, accessed: 2024-01-12
3. Blackshear, S., Cheng, E., Dill, D.L., Gao, V., Maurer, B., Nowacki, T., Pott, A., Qadeer, S., Rain, D.R., Sezer, S., et al.: Move: A language with programmable resources. Libra Assoc p. 1 (2019)
4. Blackshear, S., Chursin, A., Danezis, G., Kichidis, A., Kokoris-Kogias, L., Li, X., Logan, M., Menon, A., Nowacki, T., Sonnino, A., Williams, B., Zhang, L.: Sui lutris: A blockchain combining broadcast and consensus (2023)
5. Buterin, V., et al.: A next-generation smart contract and decentralized application platform. white paper **3**(37), 2–1 (2014)
6. Forgy, E., Lau, L.: A family of multi-asset automated market makers (2022), https://arxiv.org/abs/2111.08115
7. Ivanov, N., Li, C., Sun, Z., Cao, Z., Luo, X., Yan, Q.: Security threat mitigation for smart contracts: A survey. arXiv preprint arXiv:2302.07347 (2023)
8. Liu, C., Liu, H., Cao, Z., Chen, Z., Chen, B., Roscoe, B.: Reguard: finding reentrancy bugs in smart contracts. In: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings. pp. 65–68 (2018)
9. Motepalli, S., Freitas, L., Livshits, B.: Sok: Decentralized sequencers for rollups. arXiv preprint arXiv:2310.03616 (2023)
10. Motepalli, S., Jacobsen, H.A.: Decentralizing permissioned blockchain with delay towers. arXiv preprint arXiv:2203.09714 (2022)
11. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Cryptography Mailing List (2008)
12. Ogundeji, O.: Vitalik buterin sets milestones on ethereum's route to be the 'world computer'. https://cointelegraph.com/news/vitalik-buterin-sets-milestones-on-ethereums-route-to-be-the-world-computer (2016), accessed: 2024-01-11
13. Sui: https://sui.io/, accessed: 2024-01-12
14. Ye, Z., Misra, U., Song, D.: Specular: Towards Trust-minimized Blockchain Execution Scalability with EVM-native Fraud Proofs. CoRR **abs/2212.05219** (2022), https://doi.org/10.48550/arXiv.2212.05219