Pontem Network: Move on Ethereum

1. Abstract

Pontem Network (L2/L3) is a novel optimistic rollup based on the open source OP stack, which stands out in three significant ways. Firstly, it allows for settlement/data availability on both Ethereum and other L1/L2s like Aptos and ZkSync, providing higher transaction processing speeds, redundancy in uptime and direct asynchronous/synchronous connections for liquidity to flow. Secondly, it features a Move & EVM compatible runtime that allows developers to leverage the benefits of the Move language on Ethereum while still supporting the Solidity ecosystem. Thirdly, it incorporates an app-centric economic model for gas, enabling application builders to share in network usage revenues.

2. Introduction

2.1. Multi-chain Future

The mainstream adoption of decentralized infrastructure for applications has reached a tipping point, with millions of unique addresses and billions of dollars in value transacting every day. However, technical limitations to scalability hinder widespread adoption. With Ethereum's maximum throughput at 14-20 TPS and alternative L1s like Solana experiencing uptime issues, it's evident that the infrastructure required to support billions of concurrent users using cryptocurrency is still a work in progress.

As new generation systems like Aptos come online, capable of high TPS at low relative costs, the promise of a web2-like experience on web3 is becoming closer to reality. Nonetheless, most of the value ($243B) is locked up in Ethereum as ETH and ERC20 tokens (UNI, SHIB, DAI, etc.). Therefore, the demand for scalable, secure, and efficient Layer 2 protocols has grown to meet the supply of increased activity on Ethereum. ETH liquidity is quickly growing on L2s like Arbitrum, Optimism, and ZkSync ($3B+), signaling the need for improved Layer 2 infrastructure that can serve billions of users concurrently.

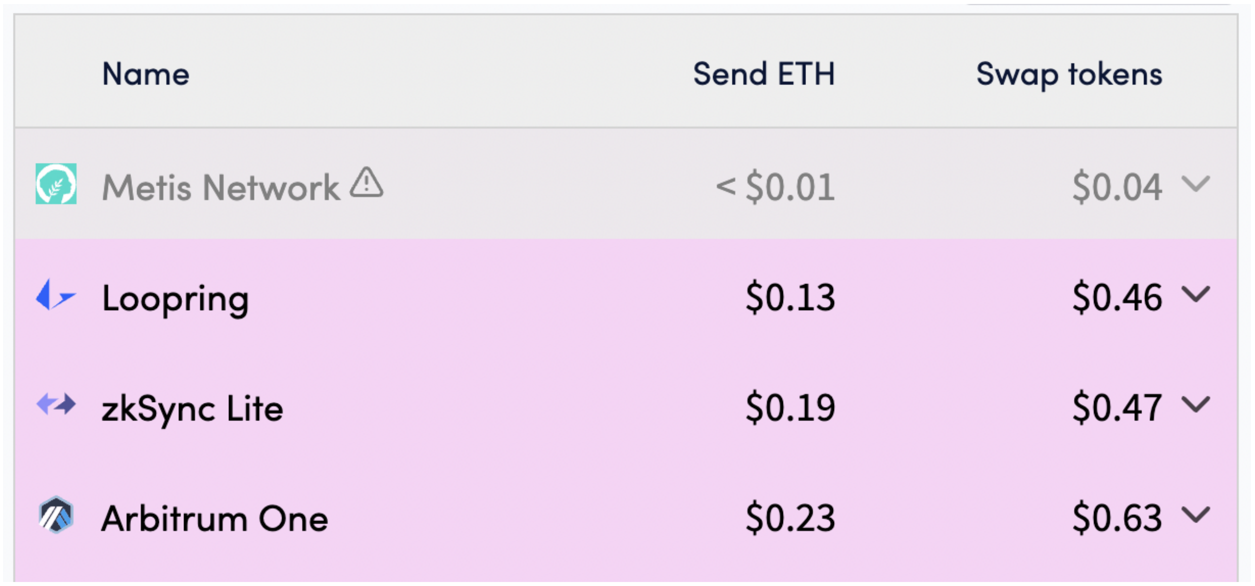
Despite the proliferation of Layer 2 solutions, such as Optimistic Rollups, zk-Rollups, and Plasma, there remains a pressing need for improved Layer 2 infrastructure that can meet the needs of millions and eventually billions of users concurrently. In order to scale horizontally, researchers posit that the L2 ecosystem on Ethereum will evolve into an ecosystem of loosely and tightly connected heterogeneous chains sharing (and competing for) security in a similar way as the Polkadot, Cosmos, and Avalanche architectures. This concept was introduced by StarkWare as 'Fractal Scaling,' a way to use proofs as a recursive structure for cost reduction.

"

L3 relates to L2 just as L2 relates to L1. L3 can be realized using validity proofs as long as the L2 is capable of supporting a Verifier smart contract. When the L2 also uses validity proofs submitted to L1, as StarkNet does, this becomes an extremely elegant recursive structure where the compression benefit of L2 proofs is multiplied by the compression benefit of L3 proofs. In other words, if each layer achieves, for example, 1000X in cost reduction, L3 can reach 1,000,000X reduction over L1 — while still retaining the security of L1.

“

With gas costs still in the $0.20 range on a random day, use cases such as high frequency trading and gas subsidized gaming transactions are still too cost prohibitive on L2s. source: https://l2fees.info/

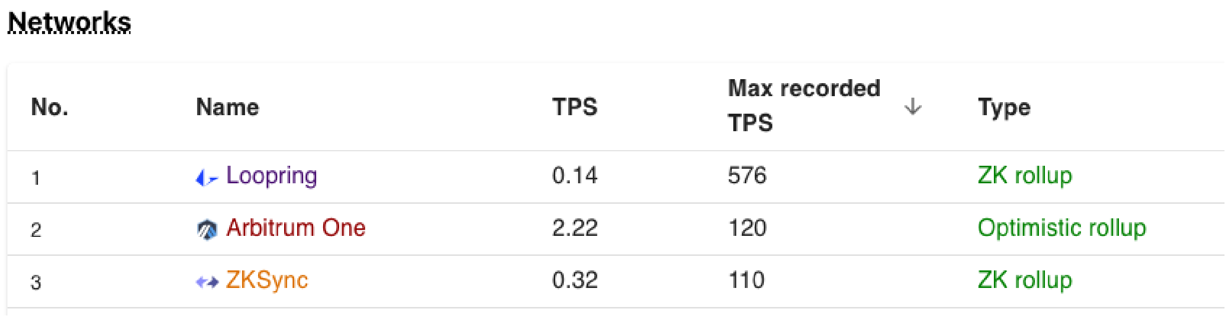


Name	Send ETH	Swap tokens
Metis Network ⚠	< $0.01	$0.04
Loopring	$0.13	$0.46
zkSync Lite	$0.19	$0.47
Arbitrum One	$0.23	$0.63

(source: https://l2fees.info/)

Optimism's Superchain explainer provides further insight into this shared vision of interconnected chains on Ethereum and the research problems that need to be addressed. Arbitrum's Orbit and ZkSync's Hyperchain visions depict a similar future.

The main limitation to Ethereum scalability is Ethereum itself. Although L2 protocols have improved, most have only reached a maximum recorded TPS of a few hundred. There is a path for lower costs, but there may still be limits to TPS due to Ethereum's maximum TPS and as mainstream adoption of public blockchains for financial settlements grows, Ethereum will fill up and be relatively expensive and slow compared to modern tech L1s. Even as protocols improve, L2s like Arbitrum predict they will only reach around ~40K TPS in their end state. Pontem L2 aims to address this limitation by scaling TPS horizontally across L1s using the bandwidth of alternative L1s like Aptos to meet network demand with Ethereum as a failsafe. To achieve this cost effectively, the majority of data is stored on the alt L1s while utilizing hashes of relative data on Ethereum. As long as the alt L1 remains operational, users can expect comparatively lower costs than other L2 solutions like Optimism. The end-state architecture will combine the TPS bandwidth of alt L1s like Aptos with Ethereum to ensure uptime and censorship resistance of Ethereum liquidity like ETH.


Networks

No.	Name	TPS	Max recorded TPS ↓	Type
1	Loopring	0.14	576	ZK rollup
2	Arbitrum One	2.22	120	Optimistic rollup
3	ZKSync	0.32	110	ZK rollup

(source: https://ethtps.info/)

Pontem L2 is designed to be compatible with existing Layer 2 solutions' future plans to establish a common shared sequencer layer and enable seamless liquidity movement and cross-layer execution. This approach ensures that Pontem L2 can easily connect with other Layer 2 protocols, such as Optimism, ZkSync or Arbitrum, in the future. Pontem L2 has drawn significant inspiration from the Optimism Stack, making future adherence to the sequencer standard for Layer 2 stacks a relatively straightforward process. For further details on these proposed ideas, readers can refer to Optimism's "The OP Stack Landscape".

Problem: Challenges in Bootstrapping Liquidity for High Performance Use Cases

New applications that require high transaction throughput, outside of Ethereum, such as App Chain solutions on Cosmos and Polkadot, face difficulties in bootstrapping liquidity efficiently. This is because the majority of the cryptocurrency value is still concentrated on Ethereum in the form of ETH, stablecoins, and ERC20 tokens. Liquidity providers are hesitant to move their liquidity to new ecosystems outside of Ethereum due to concerns about potential bugs in the core codebase, 51% attacks, and bridge security.

Solution: Ethereum L2 Rollup Protocol

Pontem L2 is a rollup protocol that leverages the robustness of multiple Layer 1 (L1) protocols, initially including Ethereum and Aptos. It operates by transmitting transaction data to a single L1, which serves as the data layer, while other L1s only receive current state hash and transactions including withdrawals are confirmed by a committee of nodes. To ensure secure data communication between the data layer and other L1s, it employs a committee architecture, similar to Arbitrum's AnyTrust. This architecture not only enables Pontem to save on transaction costs, particularly as the data layer L1 becomes more cost-efficient, but it also expands transaction bandwidth. This setup maintains the provision to verify the current state of L2 on any supported L1s, allowing anyone to upload the necessary data for verification during a fraud proof. Thus, Pontem L2 can operate on multiple L1s in a fully permissionless and transparent manner, ensuring both scalability and security.

In its existing design, Pontem L2 leverages Aptos as its data layer. Aptos stores the complete transaction history, thus simplifying the process of fraud proof verification. This eliminates the need for extra steps like data transfer that may be required in other connected L1/L2 protocols. Aptos offers significant transaction per second (TPS) capacity and cost-effective transactions, which allows for swift fraud proof verification and precise identification of fraudulent transaction operations. This setup enables other L1 protocols, such as Ethereum, to simply rely on a committee that can snapshot the current state of the L2 stored in Aptos and share it with Ethereum at predetermined intervals, such as once per day, hour, or minute. In the event that the committee reports fraudulent data or goes offline, the standard L2 fraud verification procedure can be initiated on L1 protocols, such as Ethereum, which may not contain all necessary data. The fallback procedure ensures the solution remains censorship resistant and resilient, thereby enhancing the overall security and reliability of the network.

By leveraging high-throughput chains like Aptos, Pontem L2 can effectively increase transaction bandwidth, uniting the advantages of high TPS chains with the security and liquidity of Ethereum. This approach will empower Pontem L2 to facilitate liquidity for new applications that demand high transaction throughput and enable liquidity providers to more efficiently distribute their resources across various ecosystems.

Disclaimer:

This is a theoretical concept that is subject to significant change, including the possibility of the overall concept being entirely impossible.

2.2. Move Beyond the EVM

In addition, a native Move VM on Pontem L2 will improve performance, safety and developer UX on Ethereum. The majority of current Layer 2 solutions are either EVM-compatible or equivalent. Although the EVM and Solidity were important milestones for the ecosystem in their role as the first smart contract platform, they have limitations that make them less attractive for modern blockchain application business logic and infrastructure. For example, the EVM's single thread design inherently limits parallelized execution, which means that the transaction throughput of Layer 2 solutions is not only constrained by the Layer 1's capability but also by the runtime itself.

Some of the major EVM drawbacks:

- Poor architecture workarounds: Most tokens based on the (EVM) are smart contracts that adopt the ERC20 standard. However, every individual implementation can vary, which may introduce unexpected bugs. An example of this is the challenge of accurately accounting for all tokens held by a specific address, a topic which has been discussed on StackExchange. This issue originates from the evolving standards governing token deployment.

  Even in recent times, a multitude of standards for fungible tokens have been introduced to address the shortcomings and limitations of the standard ERC-20 contract. These include ERC-223, ERC-777, and ERC-1155, each designed to tackle specific bugs and enhance token functionality.

  In contrast, blockchain networks based on the Move protocol have a well-architected default token standard which is deployed only once as part of the standard framework. This standard is used as a module for registering new tokens, ensuring uniformity in the treatment of all coins within the network. As a result, any token can be passed into functions in a manner similar to the native gas token of the network. Refer to the provided example for a more detailed understanding of this process.

- Re-entrancy exploits: The EVM currently lacks inherent protection against Re-entrancy exploits at its core level. This necessitates that most developers adhere to certain protocols, such as the `reentrancy` modifier, to guard against these vulnerabilities. However, if either the developer or the auditor overlooks this requirement, it leaves the system open to potential attacks. Despite being a well-documented and long-standing issue, these types of hacks still occur with alarming frequency.

On the other hand, the Move Virtual Machine (Move VM) is inherently designed to prevent re-entrancy attacks at its core level. This protection is attributed to its "resource" data design and strict modular system. As a result, dynamic calls to "untrusted" contracts cannot be executed, thereby eliminating the risk of double-spending the coin balance.

- Hard to fetch balances: Fetching current token or NFT balances directly is currently impossible without the use of third-party services due to the EMV's storage design. The data is stored in the smart contract storage, which lacks an API or user-friendly way to fetch it by user address or similar methods. As a result, determining token balance, NFT balance, or any other related data becomes a complex task. To navigate this, applications like wallets typically rely on third-party software that indexes the entire blockchain or ask users to provide their token address to display the token balance.

  On the other hand, the Move Virtual Machine (Move VM) makes data retrieval significantly easier. Since all data is stored under a user's account, the user's address serves as a key to this storage. Developers simply need to identify the data type and query the storage using the address and data type. A helpful API for this process is readily available 'out of the box', making the process straightforward and user-friendly.

- The `approve` function, integral to the ERC20 standard, carries significant risks: any contract can access the coins approved on your wallet at any time. For a comprehensive explanation of this issue, refer to **Metamask's detailed support article**. On the other hand, Move VM-based chains, such as Aptos, eliminate this function entirely due to their well-engineered coin standard implementation.

- In the Ethereum ecosystem, to treat the native coin (ETH) as an ERC20 token, you must first create a wrapped version (WETH) which can create the very common mistake of people sending WETH directly to the contract in an attempt to unwrap it. In contrast, the native coin of Move VM (for example, APT) always adheres to the same standard.

- The Ethereum Virtual Machine has non-human readable reproducible builds for smart contracts. It doesn't allow to verify smart contract code and metadata on chain, the bytecode it produces is based on very low level commands, it doesn't contains information of functions or variables types. In contrast, Move VM chains allow for the upload of smart contract metadata and source code directly to the chain, or even for the disassembly of it into a roughly human-readable format, because Move bytecode is replicates the Move language itself, contains variables types and functions.

- At the core level of the EVM, smart contracts cannot be updated. Contract upgradability is a crucial feature for many decentralized applications (DApps), yet developers are forced to create their own proxy/upgrade contracts for each new project in the EVM. In contrast, Move VM enables developers to configure the upgrade capability of contracts by simply adding **one line to the configuration file**. Simultaneously, it facilitates safe upgrades due to the availability of several upgrade policies and requires developers to transparently adhere to specific rules for the chosen policy.

- It is unlikely that legacy design flaws will ever be fixed in EVM, as doing so would compromise compatibility with existing on-chain EVM contracts.

To address these limitations, Pontem L2 will introduce Move as the native language for smart contracts, bringing all the benefits of Move to Ethereum developers. The Move language is intuitive and relatively expressive like Solidity and Javascript, but also inherits the safety and efficiency of the Rust language which Move is built on top of. It enhances the security of smart contract development by mitigating common issues found in Solidity, such as reentrancy errors with dynamic dispatch, unsafe code constructions, and inability to be formal verified without external tooling ("from the box").

Pontem L2 will maintain EVM compatibility for synchronous contract calls between apps that compile to EVM and Move bytecode, allowing users and developers to use existing EVM infrastructure like wallets and dev tooling. Although most applications today are built on Solidity, we anticipate more new applications entering the market to be built natively with

Move and many incumbents to migrate to Move as some large applications like Pancakeswap have already done so. In the future, now having the choice to do so, we believe app developers would prefer to only release updates to a Move codebase on Ethereum instead of Solidity, given the benefits that Move offers for smart contract efficiency and security.

Problem: Building Safe and Efficient Ethereum Apps with Move VM

Existing Layer 2 solutions that are primarily focused on Ethereum Virtual Machine (EVM) have limited their ability to incorporate advanced technologies that could offer increased performance and security. While some L2s support various smart contract languages and have even developed their own virtual machines, they often still borrow design elements from the EVM and will continue to prioritize its support. This approach overlooks the importance of improving the VM level, which could significantly enhance both throughput (TPS) and security. Supporting the ecosystem of DApp development on EVM is also essential, given that the majority of current DApps are built using Solidity as a codebase.

Solution: EVM Compatible Move VM on Pontem L2

Pontem L2 leverages the Move VM as a front end on top of an LLVM compiler to create its own optimized implementation. This design inherits the benefits of Move, including safety by default through bytecode verification, built-in formal verification, and high performance due to multi-thread parallel execution capabilities. In addition, Pontem L2's VM maintains compatibility with EVM, allowing for the seamless migration of existing smart contracts to the new platform and enables the use of widely adopted EVM products like the Metamask wallet.

By introducing the Move VM as the native language for smart contracts, Pontem L2 brings all the benefits of Move to Ethereum developers. This approach will maintain EVM compatibility for synchronous contract calls between apps. This enables users and developers to use existing EVM infrastructure like wallets and dev tooling while still leveraging the benefits of the Move VM. This shift to Move could improve the performance, safety, and developer experience of Ethereum-based DApps, making them more usable for end users.

2.3. App-centric Economics

Innovations in economic models represent a key area for exploration in layers built on top of Ethereum. Achieving a game theoretic equilibrium that incentivizes all participants to act in the best interest of the ecosystem is an ongoing challenge. The interaction of incentives among users and node operators has been tackled successfully with proof of work and proof of stake consensus mechanisms. Nevertheless, the field is still evolving, as exemplified by Ethereum's recent implementation of EIP-1559. This upgrade separates a "base fee" from miner incentives and burns it, in order to improve the efficiency of gas auctions and prevent spam on the network. Yet, while L1/L2 infrastructure incentives have been examined from the perspective of miners and validators, applications and their communities of developers and users have been overlooked.

As a result, innovative concepts for token economics and revenue generation have mainly been developed at the application level. App-chain ecosystems like dYdX have started merging L1 protocol and application-level economic models, but incentive alignment across multiple apps in an ecosystem is still a nascent area. Pontem L2 aims to fill this gap by exploring a novel economic system that shares the value of network usage with a trusted set of applications running on the protocol. By expanding the scope of incentive alignment, Pontem L2 hopes to foster a more equitable and sustainable ecosystem for DApp developers, validators and users alike.

Problem: Capturing Value from L1/L2 Usage for DApp Developers

Applications running on smart contract platforms like Ethereum drive demand for usage of the compute resources that maintain the applications running on the decentralized infrastructure of the L1/L2. However, the current economic model of these platforms charges gas fees to users to mitigate spam and compensate validators of the network of servers maintaining the network. The recent iteration of this economic model, EIP 1559, aims to accrue value to the native asset ETH used to pay for gas. While this is good for Ethereum’s monetary premium, it does not directly drive value to the applications that drive demand for usage of the network, and whose users have to pay a premium during times of high activity. Therefore, this new mechanism creates a misalignment of incentives between the L1 economic model and applications built on top.

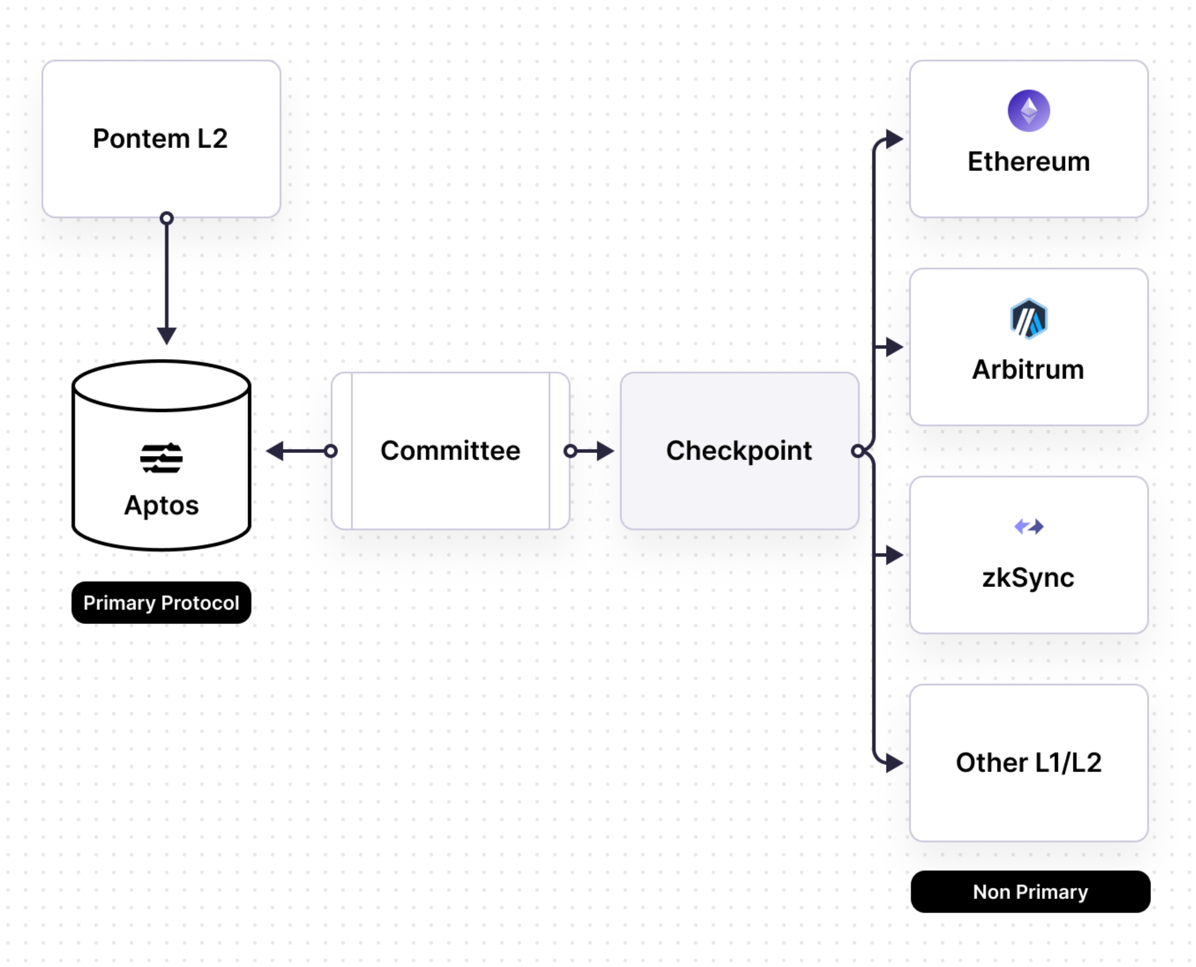
Solution: Empowering DApp Developers with Protocol Incentives

Pontem L2 aims to empower DApp developers by allowing them to earn protocol incentives, which will include a portion of the ‘base fee’ from EIP 1559 that would have been burned during gas usage, in addition to protocol emissions that would have normally only gone to validators. This architecture encourages the development of DApps within the ecosystem and shares excess revenue gathered during times of high activity. The amount of tokens DApp developers will receive is proportional to the value generated by their smart contracts. The specific emission amount and apps that qualify for rewards will be determined through governance voting, ensuring a fair distribution aligned with the community. By doing so, Pontem L2 aligns the incentives of the L1/L2 economic model with applications built on top, encouraging more developers to build on the platform and promoting a more sustainable and fair ecosystem.

The concept described is not new and is now a heavily discussed EIP proposed by the Ethereum community called ‘Generalized CSR Protocol’ as a standard for implementing this across EVM environments. See EIP-6969 and section “3.2. App Centric Economic Model”.

In order to enable this, Pontem L2 introduces an innovative ownership model for DApp developers via NFTs, providing a fair reward system through gas fee generation and emission rewards. The central component of this model is a fragmented NFT that grants complete control over the DApp, including upgrade abilities, revenue generation, and the power to pause or resume transactions. These NFT fractions can also be utilized for governance purposes. The implementation of this model is streamlined through the Move VM and Aptos framework.

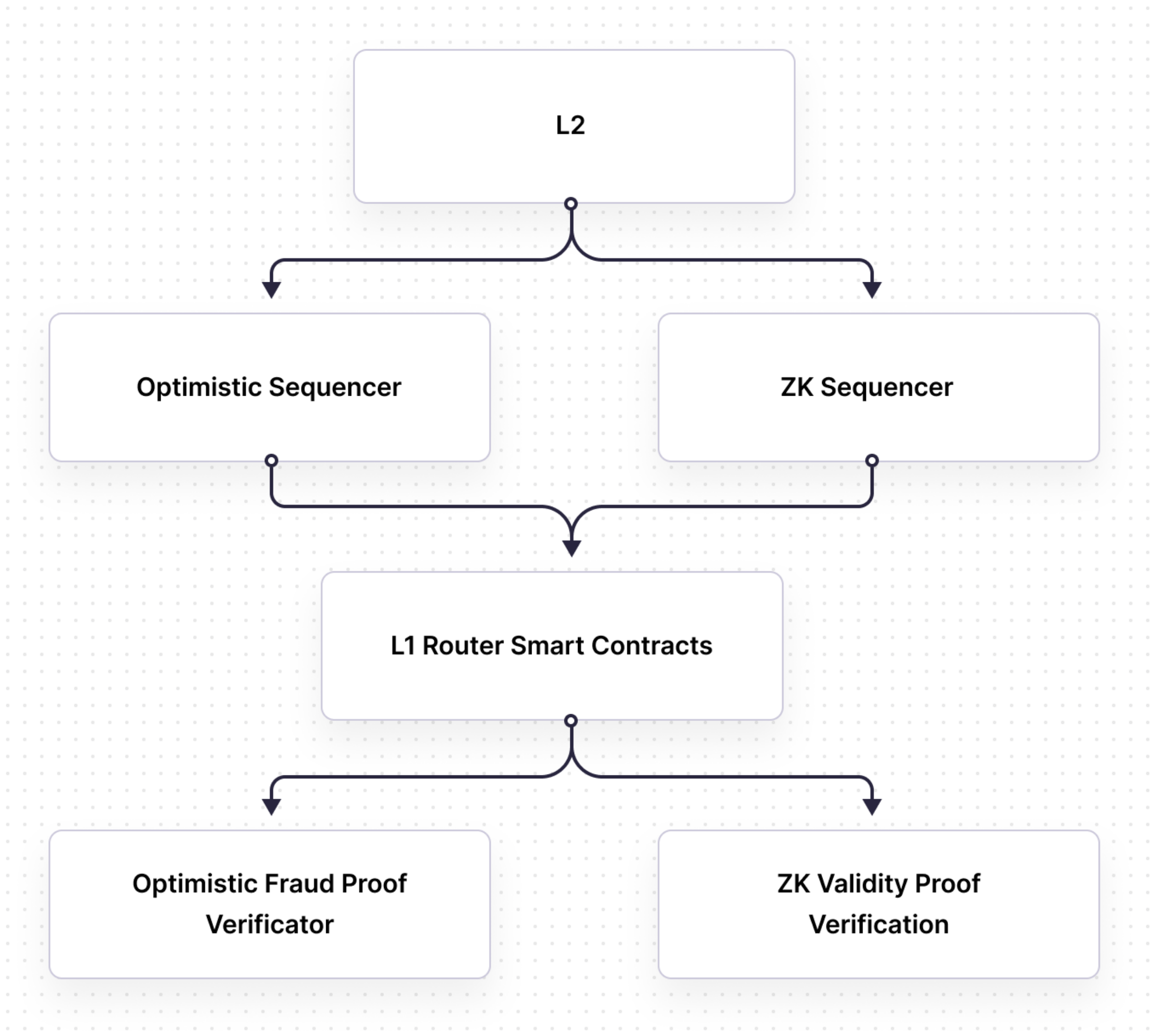
3. Design



Pontem L2 Multilayer

3.1. Architecture

Pontem is an optimistic rollup that leverages the security of Ethereum and plans to extend its security to multiple Layer 1 and Layer 2 protocols. Built with a modular architecture and upgradable components, Pontem L2 ensures flexibility and adaptability to meet the evolving needs and innovations within the blockchain ecosystem. Similar to Optimism Bedrock, this design approach allows for seamless integration of new fraud proof types and support for multiple networks, while also providing the ability to introduce future enhancements and optimizations. Upgrades and parameter changes will be governed by a DAO consisting of Pontem community members, utilizing a token voting mechanism.


L2
Optimistic Sequencer
ZK Sequencer
L1 Router Smart Contracts
Optimistic Fraud Proof Verificator
ZK Validity Proof Verification

Modular proof system

Initially, Pontem will utilize standard interactive fraud proofs, but the ultimate goal is to introduce modularity in the proof system, enabling support for not only fraud proofs but also ZK validity proofs. The L2 will operate using an attestation-based approach and later transition to permissionless optimistic settlement, similar to Optimism and Arbitrum.

Utilizing multiple protocols for data availability, Pontem L2 employs a committee consensus design, originally inspired by Arbitrum's AnyTrust model. The committee comprises respected industry entities and individuals who maintain significant token stakes at risk in the event of fraudulent activity. For efficient relaying across multiple Layer 1 and Layer 2 solutions, the system selects the most cost-effective and speedy protocol as the primary one. This primary protocol stores all necessary data for L2 verification and recovery, while other protocols receive checkpoint signatures from the committee. This design effectively minimizes transaction costs and enhances transaction bandwidth.

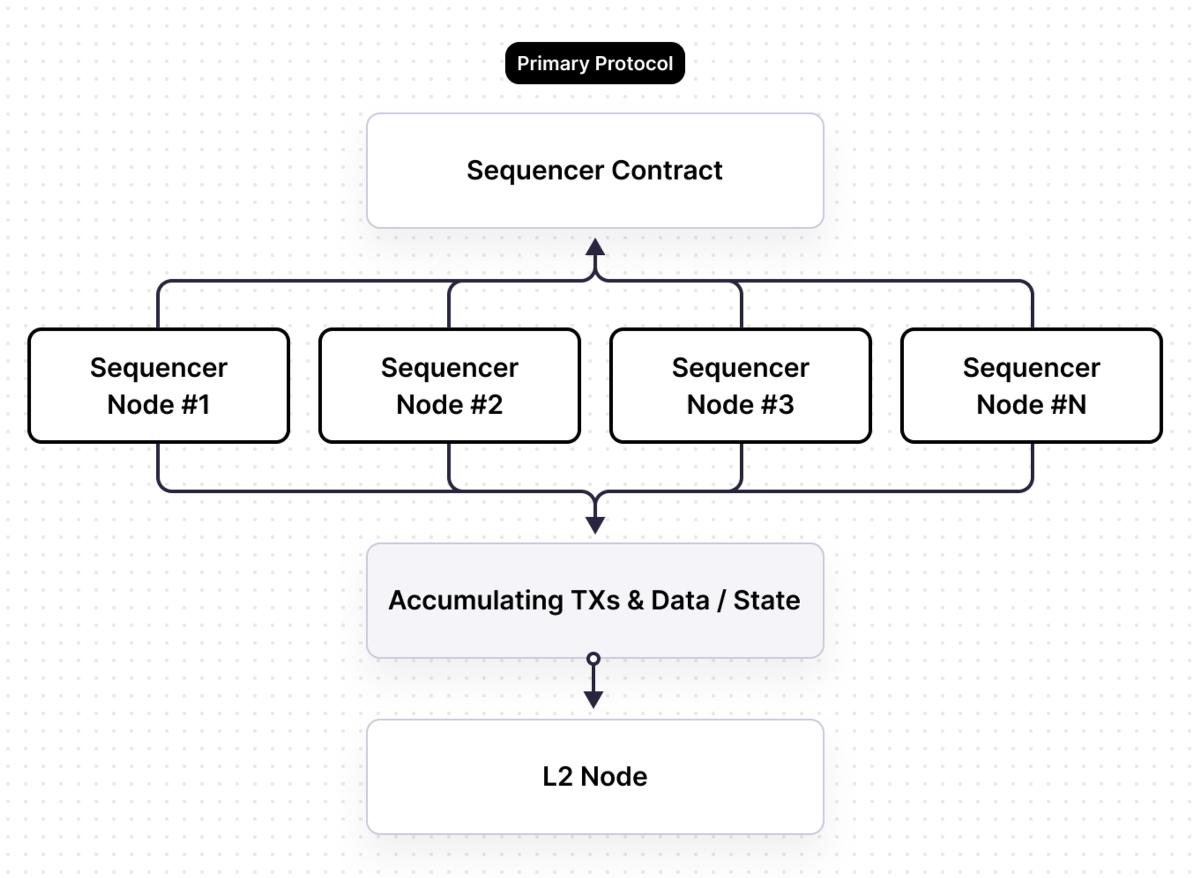
Despite the committee requiring additional trust, Pontem L2 does not sacrifice its decentralized nature or become entirely trust-dependent. If the committee were to act maliciously or experience downtime, guards could report the committee and provide proof of fraudulent activities, leading to the potential slashing of the committee's stake. In such cases, the L2 would automatically revert to the standard process, collecting all L2 data and submitting it to non-primary protocols. This would inevitably raise transaction costs until the committee resumes normal functionality, at which point non-primary protocols can revert to storing only checkpoint data.

The committee comprises N members and employs the formula N+1-K to determine the quorum, where K represents the expected number of honest members. For instance, with a 30-member committee, assuming at least 4 honest members, 27 signatures would be required to validate any data.

This proposed model can significantly decrease withdrawal periods and L2 finality, reducing the average time from seven days to potentially as little as a few hours or even minutes.

For a detailed technical explanation of the committee design, please refer to the relevant section “4.1. Multiple L1 / L2 Settlement”.

3.1.1 Decentralized Sequencer



Sequencers

In the initial version (v1) of Pontem L2, a centralized sequencer will be utilized, with plans to transition to a decentralized sequencer in later stages. The protocol will employ a standard Byzantine Fault Tolerant (BFT n/2+1) consensus mechanism, leveraging a permissionless set of sequencers who can participate in the sequencing process to earn gas fees and mining rewards. This design choice ensures security and decentralization while incentivizing validator participation during the bootstrapping phase of the protocol.

During the transition to a decentralized sequencer, users will have the opportunity to stake PONT, the native gas and governance token of the protocol, to participate in the decentralized sequencing process. They will be responsible for running their own nodes, ordering transactions, and sharing proofs with the L1. Joining the decentralized sequencer will require a specific minimum amount of PONT coins, similar to other Proof of Stake networks. Only the top stakers will have the opportunity to participate in the leader selection process for the sequencer, while other users will be able to delegate their stakes.

Initially, when the sequencer is operated solely by the Pontem team, gas fees collected will be used to cover L1 gas and L2 server costs, as well as to implement a buyback and burning mechanism for PONT coins, if earnings are sufficient to cover node execution server expenses. This approach will be further explained in the “3.3. Gas” section, providing a more comprehensive understanding of the native coin, gas strategies, and economic aspects of Pontem.

Please note that the aforementioned details are specific to the initial stage of Pontem L2 and may evolve as the protocol progresses.

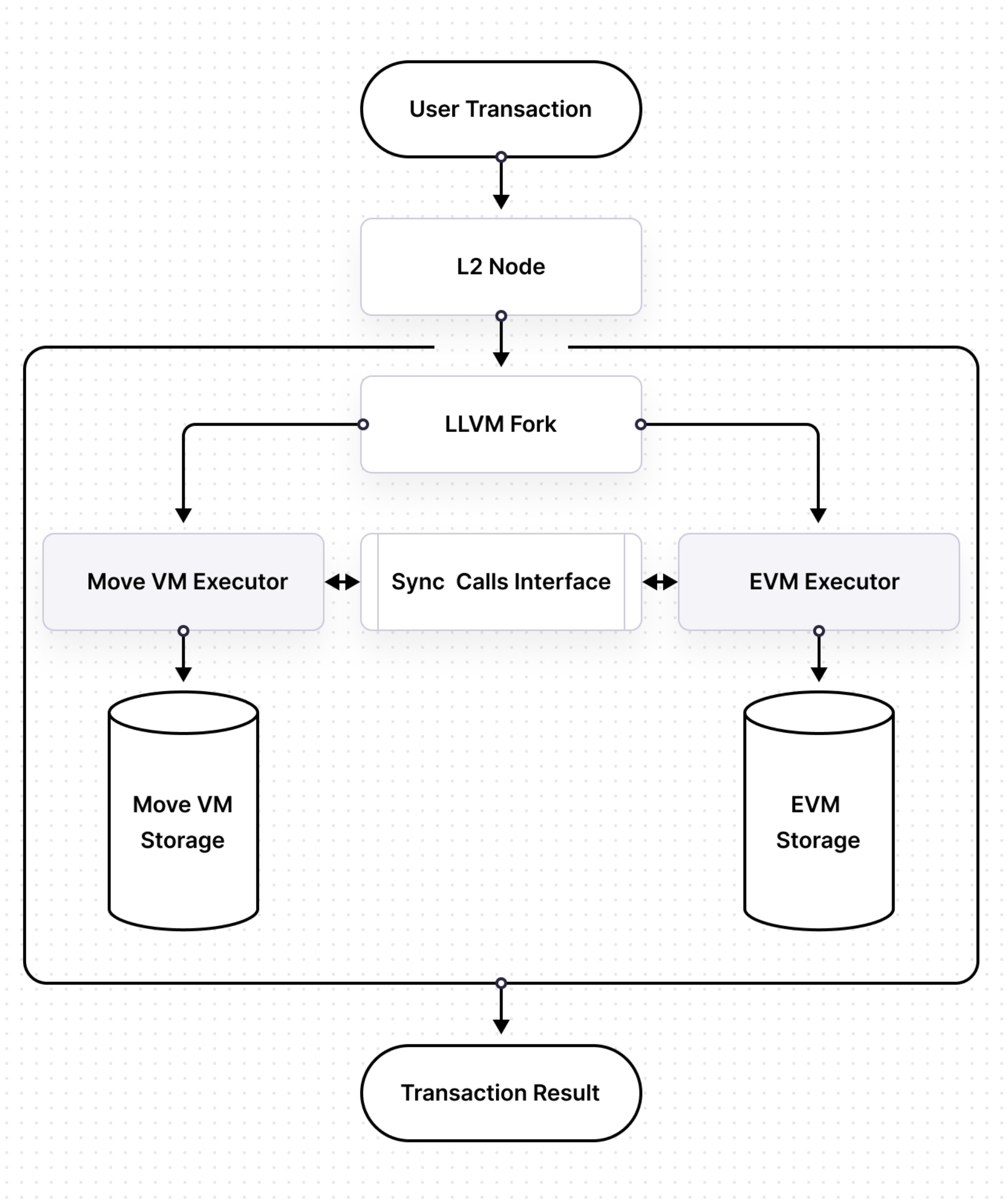
Decentralized sequencers is not a part committee. While some sequencers can be both it doesn’t mean that being sequencer allow entity or individual to participate as part of committee.

3.1.2 Execution Environment

The Move Virtual Machine (VM) represents a significant advancement in the development of secure and high-performance decentralized applications (DApps). By incorporating parallelized execution, the Move VM achieves remarkable scalability, with the ability to theoretically reach 100,000 transactions per second (TPS), greatly enhancing DApp performance. From a security perspective, the Move VM introduces innovative features like an atomic resources model, bytecode verification, and formal verification. Originally introduced by Meta, one of the world's largest technology companies, the Move VM built on Rust was purpose-built for smart contract development and verification.

Currently, the Move VM is being successfully utilized in production environments by Aptos and Sui, demonstrating its reliability within the decentralized ecosystem. The Move ecosystem is experiencing rapid growth and is projected to continue expanding, with numerous funded projects recently launched on Aptos and a similar number anticipated to debut on Sui. The community surrounding Move comprises approximately 1 million individuals, including 10,000 to 20,000 daily active users and thousands of developers. In terms of development tools and frameworks, Move has already reached a level of maturity comparable to Solidity. For instance, Move Playground offers a compelling alternative to Remix.

While we acknowledge the Ethereum Virtual Machine (EVM) as the first Turing-complete virtual machine designed for decentralized ledger technology, and recognize its widespread adoption as the foundation for smart contracts worldwide, we also understand the importance of enhancing the developer and user experience by letting them use tools they are used to. Consequently, we prioritize EVM compatibility from the outset and aim for long-term EVM equivalence. To this end, we propose a seamless integration of the Move VM and EVM within our execution layer, enabling both virtual machines to coexist harmoniously and operate efficiently in our proposed environment.



Execution Layer

After devoting over a year to tackling this challenge, initially through our EVM ↔ Move VM compiler and subsequently by integrating Move VM into ZkSync, we have amassed significant expertise in addressing this problem. The Pontem L2 will leverage LLVM as its primary execution layer in the background, facilitating the execution of Move VM (Aptos framework) as well as EVM bytecode.

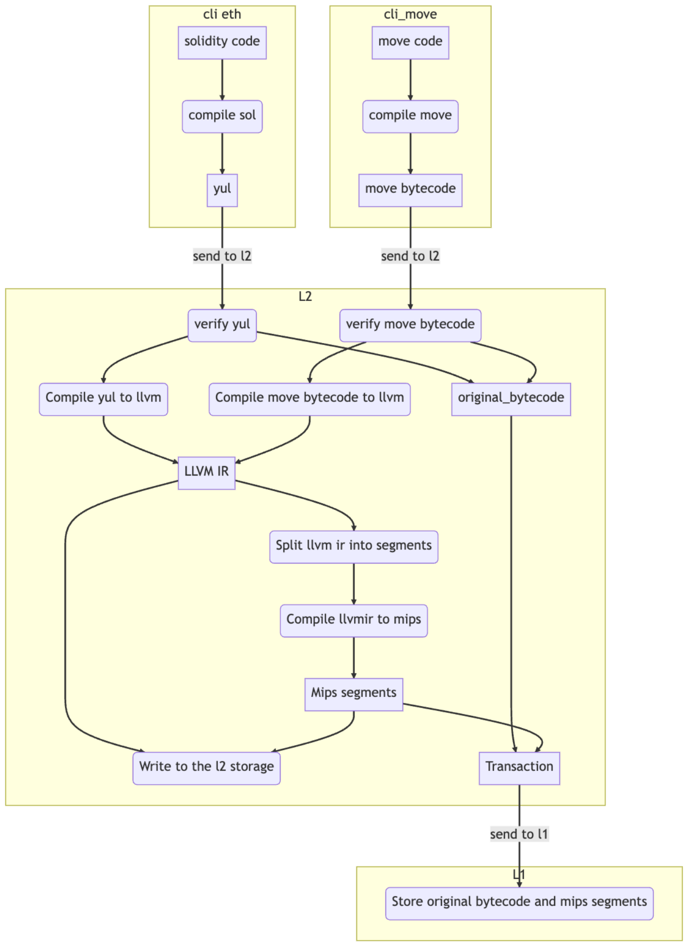
This approach generates Optimism-like MIPS instructions, which can be interactively verified by the L1/L2. It enables the coexistence of both virtual machines while supporting parallelized execution and a bytecode verifier for Move VM. Our long-term objective is to achieve EVM equivalence, which entails isolating the EVM environment.

To ensure maximum security, synchronous calls between EVM and Move VM will occur through a specialized interface designed explicitly for this purpose. For instance, in Solidity, it may be deemed acceptable to pass structures and access structure fields in third-party contracts, but such practices are not considered safe in Move. In such cases, a high-level interface can be utilized to facilitate calls between applications using low-level native functions that handle call data, module names/addresses, and method names.

This approach guarantees secure execution and adherence to the rules of both VMs whenever feasible. Consequently, Pontem L2 will enable DApps like Liquidswap (developed in Move) to seamlessly interact with Ethereum-based DApps like AAVE, granting users access to loans for swapping or adding liquidity.

In the proposed design, storage space and memory will be isolated between VMs, ensuring that Move VM storage cannot be accessed directly (without using the call interface) from the EVM, and vice versa. Due to the differing native cryptographic primitives in the Move VM (Aptos) and the EVM, such as hash functions and digital signature functions, fraud proof verification may pose challenges on an L1 which does not support specific functions. For instance, while Aptos has a native implementation of `secp256k1`, EVM does not. This could result in difficulty verifying a Move VM smart contract that employs this function on an EVM-based L1 or L2.

To address this, our proposed solution is to initially maintain a minimal set of cryptographic primitives and functions compatible with both the Move VM and EVM, and later scale this up using smart contract-based solutions. For example, `secp256k1` has already been implemented in Solidity and can be used during fraud proof verification. Similar methods can be applied for other primitives. A potential drawback of this approach is that developers may need to modify their smart contracts to utilize supported cryptographic primitives if the original ones are not supported. However, most smart contracts do not heavily rely on cryptography: many of the cryptographic primitives implemented in Aptos are used for internal processes, such as validating keys and managing multisignatures with `multied25519`.



Low level deployment flow

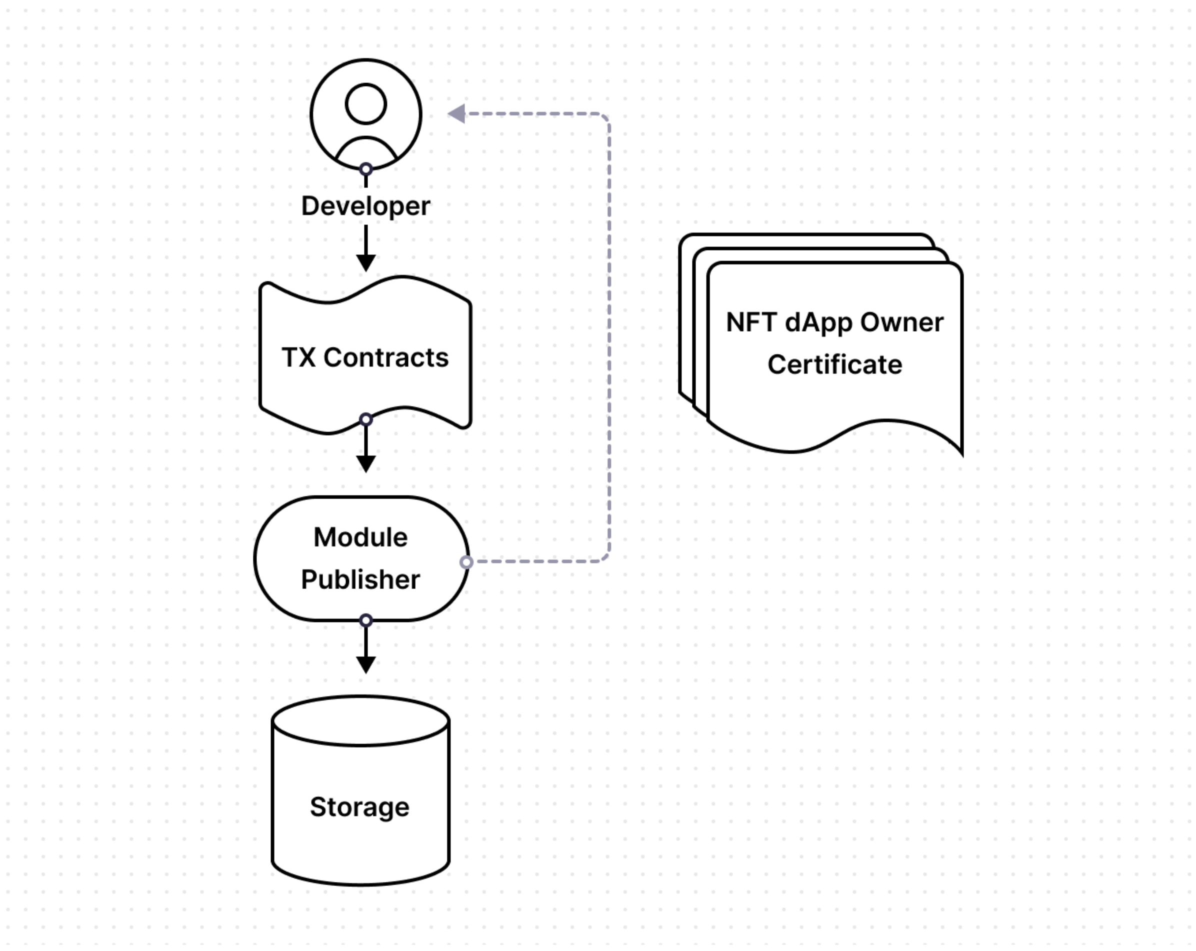
3.2. App Centric Economic Model

Currently, there are **12,495 DApps** spread across various blockchain networks, with 2,500 DApps boasting at least one active user in the last 24 hours. Notably, at the beginning of 2023, Uniswap alone attracted an average of approximately **30,000 daily active users**. DApps play a pivotal role in driving user adoption and adding value to the underlying protocols they are built upon. Imagine Ethereum without the likes of Uniswap, Curve, Aave, or Compound; it would lose its essence and utility as an application platform. The current standard economic model often leads to an **unfair distribution of rewards for DApp developers**, with the majority of profits, including transaction fees, token price growth, and liquidity, flowing into the protocols (L1/L2).

To generate revenue, DApps commonly resort to launching their own governance tokens. However, these tokens often have limited utility and fail to adequately represent the value and usage of the DApp itself, as they are not essential for utilizing the DApp. For instance, consider the UNI token created by Uniswap, which primarily serves as a governance token. While it holds potential, it does not significantly capture the true market value of Uniswap usage. Moreover, this practice can create market pressures, as developers often rely on selling their pre-mined tokens to cover development expenses.

To address this issue, Pontem L2 draws inspiration from the pioneering **Contract Secured Revenue** (CSR) model initially proposed by Canto and currently proposed as EIP-6969. This innovative approach seeks to establish a more equitable reward distribution system for DApp developers and their communities within the decentralized ecosystem.

3.2.1. CSR NFTs



Issuing of CSR certificate

Pontem L2 introduces a novel way for DApp builders to leverage the value of their applications by introducing an ownership model through NFTs that represent DApp ownership. This groundbreaking approach ensures that DApp developers and their communities are rewarded fairly through a combination of gas fee generation and emission rewards, promoting a more equitable distribution of rewards. At the heart of this model lies a digital certificate, known as a fragmented NFT, which grants developers and communities complete control over the DApp.

The fragmented NFT serves as the key to DApp ownership, encompassing critical aspects such as upgrade capabilities, revenue and fee generation, as well as the power to pause or resume transaction flows within the application. NFT fractions can even be used for governance.

The implementation of this logic is simplified through the use of Move VM and the Aptos framework. Delving into the specifics, the Layer 2 (L2) technology supports both Move VM and EVM atop LLVM. This allows us to maintain a unified gas tracker for both virtual machines. This tracker will be enhanced to track all contracts involved in transaction calls to accurately determine the gas usage for each one.

Simultaneously, the publishing of smart contracts will be facilitated with a modified `code` module derived from the Aptos framework for both Move VM and EVM. As a part of this process, a valid CSR certificate will be issued to the deployer. Importantly, from an end-user perspective, there will be no changes in the way users sign and publish transactions, ensuring EVM compatibility is preserved.

The strengths of the outlined approach lie in its ability to maintain consistent logic for both VMs at the core level, thereby promoting code reusability across both VMs.

Subsequently, the DApp ownership NFT can be assigned to a smart contract governed by a DAO, sold to other parties, used as collateral for loans, or locked within other DeFi applications. This flexibility empowers DApp owners to efficiently manage and utilize their ownership NFT to their advantage.

3.2.2. DApp Revenue

DApp developers within the Pontem ecosystem will be rewarded through a combination of gas fee generation (base fee) and emission rewards, promoting a fair and inclusive distribution of rewards. Each interaction with a contract will allocate a portion of the user-paid gas fees to the DApp builders, providing them with a direct share of the transaction revenue. Additionally, selected DApps in the ecosystem will receive a portion of the emission rewards.

To participate in the reward system, DApp builders can list their DApps in the governance system. Upon approval from the community, they will receive a portion of the rewards designated for DApp communities. It is important to note that upgradeable code requires developers to go through a pre-approval process, which may involve verification from centralized parties to ensure the developers are acting in the best interest of their communities and preventing scams.

The Contract Secured Revenue (CSR) model offers DApp builders a versatile solution, enabling them to create an ecosystem centered around shared ownership of the DApp through NFTs. CSR NFTs can be locked in smart contracts or used as collateral, empowering developers to launch a DAO that facilitates voting on proposals and determines how revenue should be allocated. The fragmented nature of CSR NFTs allows community members or users to utilize fragments for voting, eliminating the need for launching new governance tokens. This inclusive approach ensures that revenue can be distributed not only to DApp developers but also to DAOs, communities, or even utilized as collateral for DeFi products, creating a flexible and inclusive financial ecosystem. Developers also have the flexibility to add additional revenue streams from their protocols to NFT holders through smart contracts.

As a result, DApp builders, communities, and DAOs can benefit from three revenue streams:

1) Revenue share of the fees generated by the L2.
2) Emission rewards.
3) Traditional DApp monetizations such as tokens and staking.

Revenue sharing and emission rewards occur with all smart contracts participating in the transaction, ensuring that fees are fairly distributed among all involved parties. DApps involved in generating transaction fees through participation in transaction internal calls will be rewarded accordingly. The rewards earned by users can be received in the Layer 1 coin (APT/ETH) or Pontem's native coin, "PONT," depending on the preference. However, emission rewards will always be paid in PONT. The logic behind the distribution of fees is straightforward and bears similarity to EIP-1559. However, unlike EIP-1559 where the base fee is completely burned, here it is distributed among all contracts that participated in the transaction call. This fee can then be withdrawn by the holder(s) of the CSR NFT. The fee distribution could potentially be the entire amount that would have been burned, or it could be a specified percentage.

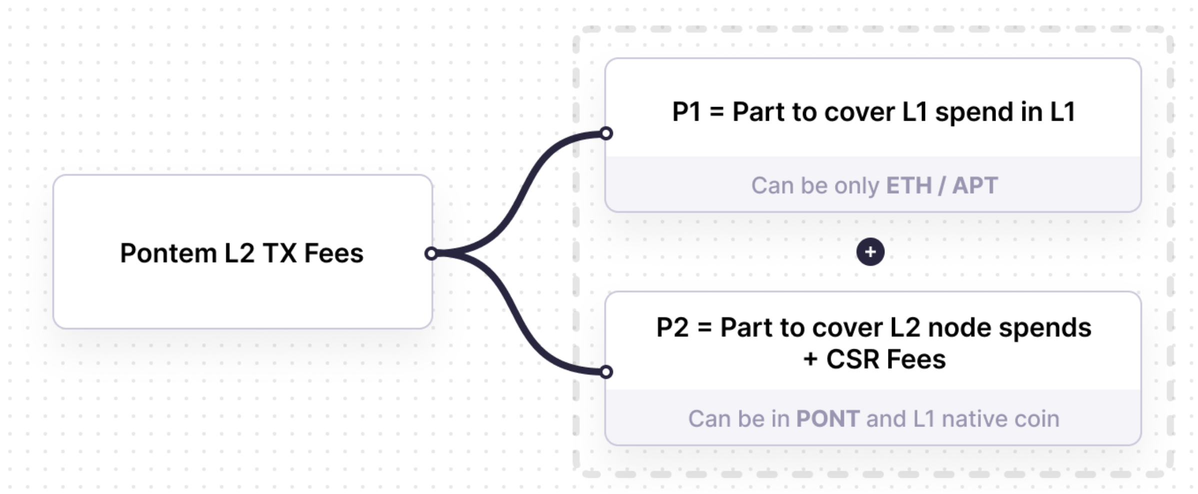
By adopting an application-centric approach, Pontem L2 cultivates a healthy ecosystem where users, developers, and validator (sequencer) nodes are all appropriately rewarded. DApp builders and users holding a fraction of an NFT can earn rewards for their contributions, whether through development, liquidity provision, usage, or other valuable actions defined by DApp developers. They can also influence the future evolution of the DApp through governance or receive rebates on fees spent within the DApp. At the same time, sequencer nodes continue to be rewarded for executing transactions and submitting proofs.

The comprehensive reward structure encourages economic growth in DApps, making up for the value previously accrued from tokens being burned. The model allows for a portion of fees to be burned while the ratio of fees burned versus distributed to DApp builders can be adjusted through DAO controlled parameters, allowing the community to experiment with a fair distribution that reaches a Schelling point. This flexibility ensures the sustainable growth of the network over time.

The rewards earned by developers, communities, or DAOs can be utilized to operate their own sequencer nodes, contributing to the decentralization and active participation within the ecosystem. This holistic reward system incentivizes all stakeholders to contribute to the growth and development of the Pontem L2 ecosystem.

3.3. Gas

Pontem L2 adopts a gas-based approach similar to Ethereum and Aptos, incorporating gas limits and gas prices to determine transaction fees and prioritize transactions. If we get into details to implement CSR the L2 going to use implementation similar to EIP-1559 but not exactly the same. As L2s need an gas to cover fees on L1s we can’t distribute whole “base fee” to CSR holders, but instead of it the protocol will distribute part of fee or whole amount going to sequencer nodes (P2). The percent can be configured by governance.


Pontem L2 TX Fees
P1 = Part to cover L1 spend in L1
Can be only ETH / APT
+
P2 = Part to cover L2 node spends + CSR Fees
Can be in PONT and L1 native coin

How fees split

When discussing Layer 2s and transaction fees, it is crucial to address the expenses associated with the sequencer on Layer 1. A portion of the fees paid by users on Layer 2 must be sufficient to cover the costs incurred by the sequencer for executing smart contract calls on Layer 1. The remaining fees are allocated to the sequencer node for transaction ordering and execution. This balance ensures the sustainability of Layer 2 sequencers and maintains efficient network operation.

While optimizing the Layer 2 network gas fees (P2) can involve various strategies, the first part (P1) involving Layer 1 expenses is less flexible. It is vital to always have adequate funding to cover the costs on Layer 1, not only for the current sequencer but also for future decentralized sequencers. Sufficient funding for Layer 1 expenses is crucial to ensure network stability and the smooth operation of both centralized and decentralized sequencers.

Pontem L2 is capable of supporting multiple Layer 1 (L1) networks. In its initial version, it's designed to integrate with Ethereum and Aptos. Consequently, users must pay a portion (P1) of the fees across these two networks. The majority of these fees are processed in the primary network, APT, with a smaller amount allocated to Ethereum to enable state checkpointing. To simplify the user experience, L2 allows for fee payment in different currencies, which are then automatically exchanged for the native currency on the DEX.

Considering that most users prefer to use ETH for their fees, a portion of ETH should be automatically converted into APT at the time of transaction processing. However, due to potential liquidity issues on the DEX, L2 also provides the option to pay fees exclusively in APT, a mixture of APT and ETH, or even solely in ETH if this option is activated by governance. This functionality may be particularly useful in the early stages when liquidity is not fully established.

We are also considering strategies to ensure an initial robust liquidity level for the ETH/APT pairing, followed by a permanent lock of LP coins. This ensures a minimum liquidity level indefinitely, thereby offering an intriguing economic incentive and arbitrage opportunities for bots.

Subsequent iterations of L2 will enable users to pay a part (P2) of the fees in Pontem L2's native coin, PONT, alongside the native coins of Layer 1. It's crucial to understand that this only covers node expenses and not Layer 1 costs. This P2 portion can also be paid in various coins, granted sufficient liquidity exists on the native DEX for conversion.

The protocol will seamlessly auto-swap any pre-approved coins, including whitelisted DApp coins/tokens, into a coin suitable for fee payment. This process, conducted via a concentrated liquidity DEX, simplifies the fee payment process, elevating the overall user experience. Should it become impossible to convert at the expected rate due to slippage, the transaction will default to using available PONT or Layer 1 coin(s) to cover gas fees. Governance can also manage liquidity failsafes to prevent swaps in low-liquidity situations.

If liquidity is insufficient, users may need to resort to paying fees in the native gas coin of Layer 1, given the high exchange costs. To further enhance the user experience, Pontem L2 will enable fee payments directly from the sender's account or through a decentralized exchange (DEX) protocol. When transactions are paid in PONT or any other unrelated coin to the primary L1 network, sufficient liquidity must exist on the native DEX.

Additionally, Pontem L2 offers DApps the ability to cover users' transaction fees, allocating a portion of their generated revenue for this purpose. This flexibility allows DApp developers to strategically determine when to cover fees through their code, enabling them to meet specific requirements. For instance, a DEX developer could decide to cover fees only if the swap amount surpasses a predefined threshold, thereby providing a high degree of control.

3.3.1 Gas Fee Flow

Pontem L2 introduces a flexible gas model at the protocol level, combining the standard gas approach with innovative mechanisms that empower DApp owners to determine the currency in which users pay gas fees through DEX conversion. While we anticipate that most fees will be paid in our native coins (ETH, APT or PONT), DApps can offer users the option to pay fees in their specific DApp coin or select from a set of whitelisted coins. The DApp coins must be tradeable on a DEX with the native coins required for settlement. This approach ensures adaptability in fee payment methods while preserving the central role of L2/L1 native coins in the ecosystem.

For DApp developers, the implementation of user gas payment can be illustrated with the following pseudo-code (please note that this is not the final version, but a demonstration of the developer's perspective):

```
public entry fun do_something() {
   let gas_payment: Coin<NativeCoin> = treasury::extract_gas_for_do_something(MAX_GAS);
   // After this we don't charge sender account for fees.
```

```
    gas_processor::cover(gas_payment);
}
```

To simplify the process and avoid excessive complexity, the coin swap will occur during the transaction processing stage, ensuring that the trade can cover the gas requirements. Consequently, DApp developers will always work with the appropriate native gas coin.

In the proposed model, we have identified a potential bottleneck that we are currently researching. Essentially, every transaction in any chain must have a minimum amount of gas provided for transaction verification before execution, including public key and signature verification, nonce, etc. In our approach, we also allocate a portion of gas for performing a basic swap in Liquidswap v1. While we strive to minimize gas usage in this context, we need to ensure that the chain cannot be spammed with transactions from accounts unable to cover fees in a custom coin due to low liquidity.

To address this concern, we plan to implement account creation on the chain as a prerequisite, similar to other networks like Aptos. Accounts will be required to hold a small amount of L1 (ETH/APT) or PONT coin, which helps prevent spam attacks. In cases where a swap cannot be performed, the L1 or PONT coin can be used as a minimal transaction fee.

3.3.2. Native Decentralized Exchange (DEX)

Liquidswap v1, which uses Concentrated Liquidity, will be utilized to facilitate the trading of non-gas coins for gas coins (PONT or L1 coins). We expect the performance of the Move-based version of Liquidswap v1 (Concentrated Liquidity) to be suitable for conducting coin swaps during the transaction pre-execution step. However, in the event that this performance is insufficient, we have plans to implement a semi-native version of Liquidswap v1. This alternative version would execute the most computationally intensive parts, such as bins/math, as an integral component of the L2 core code, ensuring faster and more efficient operations.

If there is user demand for additional features like a Centralized Limit Order Book (CLOB), it can be incorporated into the native DEX protocol in the future. Our goal is to establish sufficient liquidity in PONT and other L1 coins such as ETH and APT to facilitate gas payments and reward distributions. With this model, users can theoretically pay for gas in any token that has adequate liquidity, such as USDC or DApp governance tokens. Furthermore, liquidity from other ecosystem projects like CLOBs and other Automated Market Making Systems (AMMs) can be utilized with routers to ensure ample liquidity and achieve the best price execution for users. In the case of insufficient liquidity, the protocol will default to paying gas in native L1/L2 tokens such as ETH or APT.

3.3.3. Native PONT Coin

The PONT coin holds significant importance within the Pontem L2 ecosystem, serving multiple crucial functions:

- It will be utilized in conjunction with the L1 native coin to cover transaction fees, facilitating a seamless and efficient transaction process for users.
- With the implementation of a Proof-of-Stake (PoS) model for decentralized sequencers, these sequencers will be required to stake their PONT coins either through self-staking or by receiving delegated coins as part of their stake.
- Staked PONT coins will hold influence in L2 governance, granting holders voting rights and enabling active participation in decision-making processes.
- The PONT coin will be available in both the ERC-20 standard and the Move framework coin standard. This dual compatibility ensures seamless integration with both the Ethereum Virtual Machine (EVM) and the Move Virtual Machine (Move VM), allowing for broader adoption and interoperability within the ecosystem.

3.4. Governance

Pontem Governance will be established through a voting mechanism that enables community participation in decision-making processes. Similar to other chains, regular improvement proposals will be put forward for parameter switching, core module upgrades, and other important aspects. The governance framework will also oversee an on-chain treasury and crucial parameters such as inflation emissions, which serve as incentives for sequencers, fraud protection nodes, builders, and liquidity providers as discussed earlier in this paper. The allocation of total emission to each segment will be determined by the governance participants.

The Pontem L2 community as a whole will form the basis of governance participation. Any individual holding tokens can become a voter, with the option to delegate their tokens to a sequencer or even become a sequencer themselves through self-staking. Voting power will be proportionate to the stake held by each participant, ensuring a fair representation of interests. To ensure the effectiveness of the voting process, a quorum barrier will be in place. If the token amount participating falls below the quorum, the proposal will not proceed to the voting stage. A simple majority of "Yes" votes is required for a proposal to be approved and executed.

3.4.1. Voting on DApps

Voting on DApp rewards will be conducted on a regular basis, focusing on a predefined list of pre-screened DApps. Voters will have the opportunity to select DApps from the list and determine the percentage allocation of rewards among them. Once the voting process is successfully completed, the selected DApps will begin receiving their rewards. Currently, discussions are underway to determine whether DApps will receive vested rewards or stakes in native coins as their rewards.

In cases where the voting process fails to reach the required quorum, the rewards will be rolled over to the following month. A new voting round will commence immediately, providing voters with sufficient time to cast their votes.

3.4.2. Treasury

A specific portion of user transaction fees and emission will be directed towards a dedicated treasury under the control of the governance framework. This treasury will serve as an incentive for the community's efforts in building, improving, and securing the protocol. Through mechanisms like retroactive public goods funding, individuals and groups within the community will be rewarded from the treasury. This approach aligns with the standard practices adopted by other L1/L2 networks and supports the overall growth and development of the Pontem ecosystem.

4. Theoretical Concepts Under Research

Please note that the following information in Section 4 represents our current understanding and plans, which are subject to further research and development. The concepts described below are theoretical and may change as we progress.

It is important to emphasize that no system can guarantee 100% protection against hacks. However, our goal is to minimize the risk of fund losses through the implementation of the following mechanisms.

4.1. Multiple L1 / L2 Settlement

Pontem L2 is an innovative Layer 2 (L2) protocol, initially engineered to integrate effortlessly with an array of L1/L2 protocols.

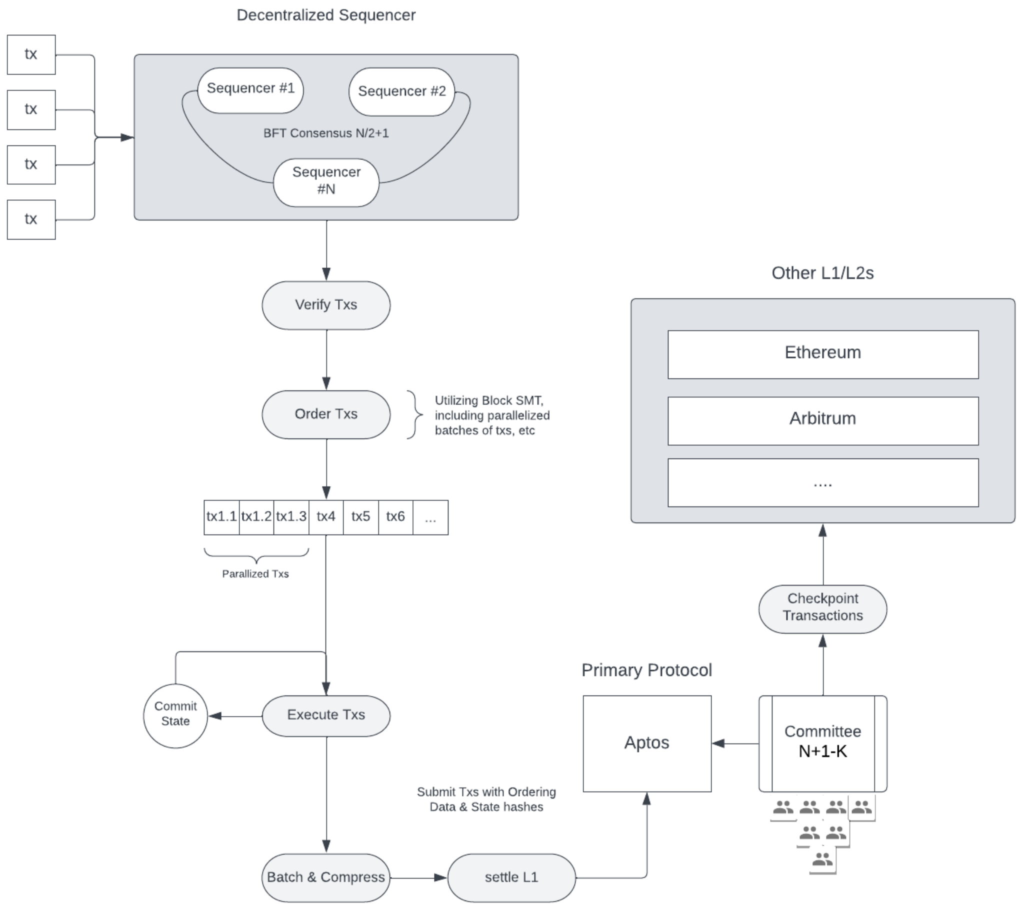
The original design of Pontem L2 addresses several issues common to existing L2 protocols:

- It scales the Transaction Per Second (TPS) bandwidth and reduces transaction fees.
- It ensures zero downtime.
- It facilitates native liquidity from various sources, including all supported L1 protocols.
- It maintains a trustless environment.

Implementing a protocol like Pontem L2 presents its own unique challenges, primarily because most existing L2 solutions operate solely on one L1 protocol, typically Ethereum. These offer limited benchmarks for our innovative multi-L1/L2 research. The solution would involve posting all produced data—state hashes, transaction data, storage hashes—across all L1/L2 protocols and establishing L2 finality on top of them. However, this approach would be less competitive due to high transaction costs, with the sequencer needing to duplicate data on each layer. The method would also be considered "heavy," involving replication of a significant amount of data that already exists on one chain, and wouldn't effectively solve TPS issues as it would still be dependent on the bandwidth of all supported L1 protocols.

Pontem L2 offers a resolution to these challenges by designating one of the supported, cost-effective, and high-performance L1 protocols as the **primary**. This means that the **primary** protocol would store all necessary L2 data and initiate the fraud verification process without requiring additional steps, such as uploading missing data. Meanwhile, the other non-primary L1/L2 protocols would function as backups to the primary L1, largely relying on a **committee**. In scenarios where the committee doesn't function correctly, the L2 can switch to standard L2 mechanisms for fraud proofs and data availability of individual L1 chains.

This approach reduces transaction costs as only one cost-effective primary protocol stores the necessary L2 data. Meanwhile, non-primary protocols receive simplified "checkpoint"-like transactions confirmed by the **committee**. Because the primary protocol can process a large number of transactions per second—for instance, Aptos can handle approximately 100k TPS—this setup significantly enhances the L2's TPS bandwidth while still maintaining backups on chains like Ethereum that have high historical uptime in case of a primary protocol outage.



4.1.1. Committee

Initially, the committee will comprise known industry entities and individuals. By submitting checkpoint transactions to all other supported L1/L2s, the committee members stand to earn a portion of the fees. However, they also run the risk of being penalized or "slashed" if a committee node fails to respond or submits malicious data.

Each checkpoint transaction will carry a list of committee members' signatures as evidence that they have signed the data using their keys. The committee consists of N members and uses the formula N+1-K to establish the quorum, where K is the expected number of honest members. For instance, in a committee of 30 members, assuming a minimum of 4 honest members, 27 signatures would be necessary to validate any data.

Checkpoint transactions are processed at set intervals, whether hourly or daily. A checkpoint transaction might look like this (please note, this is not the final design):

```
{
  nonce: "N",
  period: "N-N"
  stateHash: "0x....",
  accumulatedTransactionsHash: "0x....",
  accumulatedStorageHash: "0x....",
  withdrawlsTransactions: [
      "0x....",
      "0x....",
      "0x....",
  },
  committee_changes: [
    ....
  ],
  ...
  signatures: [
      "0x.....",
      "0x.....",
      "0x.....",
  ],
}
```

Checkpoint transactions will include hashes of accumulated transactions, state, and storage for a specific period. It's also important to mark withdrawal transactions that must be incorporated into checkpoint transactions for processing.

Each checkpoint transaction will include a latency period before being finalized. This is because L2 should not wholly rely on the committee's trust. Therefore, L2 incorporates a window during which guards or any user can report a fraud proof against the committee. The mechanism mirrors other fraud proof models but additionally requires data upload for verification. Once the latency period concludes, withdrawal processing commences, and the state reported by the checkpoint is confirmed.

Initially, the committee will earn a share of transaction fees along with sequencer nodes and part of the emission rewards. The first step towards configuring committee members would involve using a multisignature, which can later be replaced by the committee itself and governance. The exact mechanism is still under exploration, but conceptually, changes to the committee list nodes could be achieved through a voting process involving both governance and committee members. Changes would only be enacted if the proposal achieves consensus from both groups.

The committee's smart contract should be implemented on top of L2, and all changes to the committee's state occur there via proposals. If a proposal to alter the committee is executed, the changes should also be incorporated into the checkpoint transaction. This would update the committee's state across all backup L1/L2 protocols.

4.1.2. Slashing

If the committee submits fraudulent information, it can be checked through a process similar to the fraud verification proofs used in other L2s. In such a case, a guard (essentially any user) would need to provide data to initiate fraud proof verification. This data could be copied and transferred from the primary L2 using a transaction. The transaction could be sent during the confirmation period for the checkpoint transaction submitted by the committee. If fraudulent action is confirmed, the offending committee members would be penalized by losing a portion of their stake, which was locked in when they joined the committee. This does not mean that staking is a prerequisite for becoming a committee member. However, it indicates that in the future, staking might be required to function as a committee member. During the initial phase of L2's launch, staking would not be necessary, but with the protocol's growth and the introduction of the PONT coin, staking could be mandated.

The same consequences could arise if the committee fails to function properly. However, this requires careful consideration as a few non-performing committee members (N+1-K) could adversely affect the others.

The solution described is highly experimental and currently undergoing extensive research.

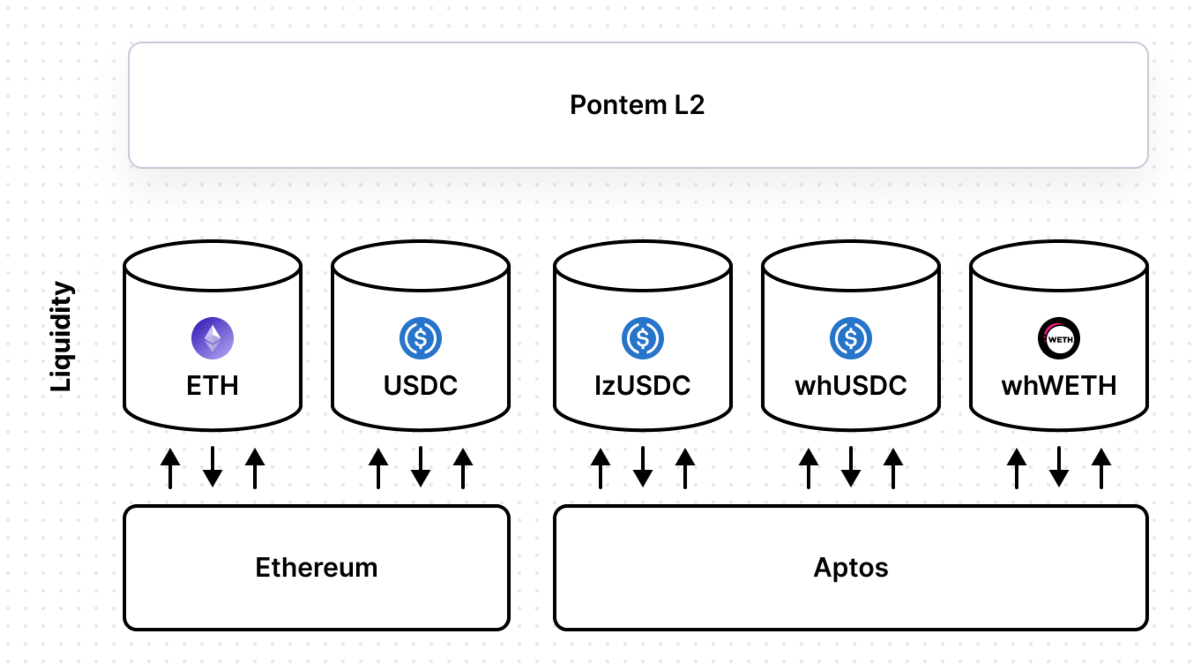
4.1.3. Failsafe mechanism

In situations where the committee is malfunctioning - either due to malicious actions, inability to reach a consensus, or total shutdown - the L2 protocol can continue functioning autonomously. This condition can be detected after a certain period; for instance, if the committee doesn't submit anything to the chain for a full day, the protocol could default to a regular operating model. In this model, transaction data and hashes of the LW would be submitted to all backup protocols, ensuring unstoppable operation.

If the primary network, such as Aptos, fails, the committee should refrain from broadcasting checkpoint transactions. Consequently, the regular fraud proof mechanism would activate on Ethereum, allowing it to become the primary network. The committee could potentially expedite the primary protocol switch by sending a specific transaction, eliminating the need to wait for the predetermined period. However, this solution is currently under discussion due to potential risk factors.

When the committee fails, transaction fees should automatically increase since sequencers would need to distribute data across both blockchains simultaneously. Once the committee resumes its operations and becomes active again, fees can be readjusted to their normal levels.

4.1.4. Deposits & Withdrawals



Liquidity Distribution

During the deposit process, tokens are locked in a smart contract based on the originating protocol and stored there until a withdrawal is initiated. Given that Pontem L2 can operate in conjunction with multiple supported L1 protocols and allow deposits and withdrawals across various networks, it mandates that all coins deposited from the originating protocol be withdrawn only to that same protocol.

Each deposit is given a unique asset ID, which contains information about the source chain and related contract details, such as the address and module name. If one were to deposit ETH or USDC from Ethereum to the L2, all deposited coins could only be withdrawn back to the original chain, in this case, Ethereum. A deposit made from Aptos, containing a wrapped USDC coin such as LayerZero or Wormhole, wouldn't match a native USDC deposit made from Ethereum. This is done to minimize the risk of potential hacks from wrapped coin bridges. However, it does introduce the issue of fragmented liquidity, which can be addressed at the application level by DeFi solutions like 3pool from Curve.

In the case of USDC, if both the origin and destination chains support the native coin representation, solutions like Circle Cross Chain Transfer CCTP can be utilized. A withdrawal operation could trigger CCTP on one of the supported protocols instead of using locked deposits of the destination protocol.

In Pontem L2, the deposit period usually ranges from 1 to 10 minutes for successful coin deposits. The withdrawal period on Pontem L2 can vary significantly, ranging from several minutes up to 7 days, depending on the current load of the primary network and the state of the committee. Since Pontem L2 uses Aptos as the primary protocol, and Aptos features high TPS and is less affected by gas DoS/DDoS attacks, the L2 can theoretically release withdrawals on Aptos within a few hours, or even minutes, once the protocol has been sufficiently "battle-tested". For other backup protocols that the L2 relies on, it would typically trust the committee's checkpoint transactions and release withdrawals based on them after a pre-configured delay, again ranging from several minutes up to 7 days. This delay allows for the initiation of a slashing process for the committee in case it is not operating correctly.

If the committee becomes unavailable, the withdrawal time could be extended to a period long enough for verification on both Ethereum and Aptos chains. In such a case, the standard L2 verification backup mechanism would be launched, ensuring withdrawals can still be processed.

4.2. Ethereum Compatibility

Ensuring compatibility with existing wallets and APIs of supported backup protocols is a primary objective for Pontem L2. The L2 node is designed to incorporate support for both Ethereum and Aptos APIs. At the same time, it facilitates the seamless operation of existing wallets such as Metamask and Trust Wallet. This includes functionality like sending Move VM-based transactions from Metamask and creating accounts utilizing Ethereum cryptography.

Pontem L2 leverages the LLVM compiler as its foundation in order to provide support for both Move VM and EVM, offering compatibility with popular solutions such as the Metamask wallet and various blockchain explorers, libraries, and development tools like Truffle and Hardhat.

Ensuring compatibility with Ethereum-like RPCs and cryptography is a key objective of our project. To achieve this, we have planned implementation of Ethereum-like RPCs at the native level, following the successful approach adopted by projects like Moonbeam and Optimism. This approach allows for smooth integration with existing infrastructure and tools, making it easier for developers and users to adopt Pontem L2.

While initially Move VM and transaction cryptography may not align directly with the defaults proposed by EVM, our goal is to establish cryptographic compatibility between Move VM and EVM by introducing additional native functions and interfaces. This strategy will bridge the gap between the two virtual machines, promoting interoperability and seamless integration for developers and users.

4.2.1. Wallet/Address Compatibility

Addressing the issue of compatibility between different cryptographic primitives, specifically Aptos's ed25519 signatures and Ethereum's ECDSA signatures, is an integral component of the Pontem L2 architecture. This is achieved through the implementation of on-chain accounts as a basic abstraction that can be managed by various private keys. For reference, one can examine the Aptos **account** module from the framework.

This module contains functions that enable changing of the public key without altering the addresses, essentially allowing for account management from both Aptos-based and Ethereum-based wallets. For instance, an account from Metamask would contain only the ECDSA public key, but it's still feasible to associate an ed25519 key with the same account if needed, and vice versa. The user merely needs to submit a specific transaction and visit a designated portal which generates the transaction and assists in sending it to the network.

Address compatibility presents another challenge since Ethereum wallets are represented as the last 20 bytes of the public key, whereas Aptos uses a 32-byte account address. A proposed solution for the L2 is to allow one account to have dual addresses: one complete and another truncated. For example, if a user creates a wallet from Metamask using an ECDSA public key, a 20-byte address will be utilized as a truncated address. However, to maintain compatibility with Aptos, the full address will contain the actual 32 bytes, with the remaining bytes padded with zeros. The same principle can be applied in reverse, by truncating the Aptos address to 20 bytes. Further, account abstraction could permit having two linked addresses, one derived from an ECDSA public key and another from an ed25519 public key.

4.2.2. Pontem Wallet for EVM

Pontem Wallet has established itself as one of the leading wallets for Aptos, boasting an impressive user base of 200,000 active users. Built on Aptos Move, the wallet is soon set to expand its compatibility to include existing EVM chains like Ethereum. However, our research indicates that a native Move wallet can offer an even more enhanced user experience.

One notable advantage of using a Move-based wallet is the seamless display of NFTs and coins. Unlike EVM-based chains, a Move wallet doesn't rely on additional APIs due to the efficient integration between Aptos REST API and storage in the Move VM. This results in a richer user experience, as wallets are not dependent on third-party APIs. Additionally, the Move VM provides the unique capability for users to preview all the changes a transaction will make (write set) before actually submitting it to the network. This feature, which may not be available in Ethereum-related wallets in the near future, adds an extra layer of transparency and confidence for users.

Taking these factors into consideration, Pontem Wallet is committed to supporting EVM-based chains in the near future. By doing so, we aim to offer users a superior experience for Move VM-related transactions. Leveraging automatic endpoint switching, the wallet will seamlessly enable interactions with applications that require both EVM and Move VM-based interactions, providing a streamlined and versatile experience for our users.

4.2.3. RPC API

Support for the default Eth RPC API is essential for sending transactions from Ethereum-based wallets on the L2 network. Although full support may not be necessary, basic functionalities like transaction sending and event fetching should operate effectively.

Taking the `eth_sendTransaction` API used in Metamask as a reference, we need to support the following transaction format:

```
params: [
  {
    from: '0x...',
    to: '0x...',
    gas: '0x..',
    gasPrice: '0x..',
    value: '0x..',
    data:
      '0x...',
  },
];

window.ethereum
  .request({
    method: 'eth_sendTransaction',
    params,
  })
  .then((result) => {
    // The result varies by RPC method.
    // For example, this method returns a transaction hash hexadecimal string upon success.
  })
  .catch((error) => {
    // If the request fails, the Promise rejects with an error.
  });
```

We can adopt the following approach to ensure compatibility with Move-based transactions:

- Given our address capabilities, the `from` field can reflect the actual user sending the transaction.
- The `to` field could be hardcoded to `0x1` or `0x0`, or it can use the module deployer address.
- Gas-related fields are fully compatible.
- The `value` field could be processed at the API level and used to bypass native coin withdrawal from the sender's account to the called function argument.
- The `data` field would contain essential call data, including the module address, name, method to call, generics, and arguments.

A similar approach can be applied from the Aptos side to call EVM-based transactions. The entry function is represented as follows:

```
new EntryFunction(module_name: ModuleId, function_name: Identifier, ty_args: Seq<TypeTag>, args: Seq<Uint8Array>): EntryFunction
```

Given that type arguments can be disregarded for EVM-based contract calls, the remaining elements can be used compatibly.

4.3. Hack Recovery Mechanism

Developing and using DeFi applications comes with inherent risks, as smart contracts often handle significant amounts of value and may not undergo comprehensive research or testing. These contracts may also rely on third-party contracts, such as oracles, and may only undergo basic security audits. This lack of security exposes them to potential hacks, where attackers can exploit vulnerabilities and steal substantial amounts of funds.

By viewing DeFi protocols as decentralized organizations, we can draw inspiration from existing protocols that offer enhanced security measures. For instance, these protocols often impose lengthy withdrawal periods, dissuading hackers from transferring funds to more decentralized chains or centralized exchanges. Withdrawals from Aptos to Ethereum on LayerZero, for example, require days, as do withdrawals from Optimism to L1. These delays serve multiple purposes, not only preventing hacks but also addressing other concerns. In the event of a major hack on Layer 1, the protocol can pause all bridges and validators, make necessary code changes to recover from the attack, and then resume operations. The recent BSC hack demonstrated a similar capability, albeit in a relatively centralized manner due to the limited number of validators.

Given these considerations, DeFi applications can adopt a similar approach, particularly in their early stages. The architecture of Pontem L2 enables emergency pausing of DApps and provides low-level interfaces at the smart contract level, allowing DApp developers to build applications with enhanced security measures and the ability to address issues promptly in case of an exploit.

It is important to note that this approach is not mandatory and should be used at the discretion of developers. Furthermore, the described approach can be disabled over time by reducing the delay to zero once a developer or DAO determines that the DApp has undergone sufficient battle testing in a production environment and has matured to become more resistant to censorship. It is theoretically possible for this hack prevention mechanism to be exploited for rolling back a DApp through governance, hence the need for careful consideration.

Note that this mechanism is implemented only at the application level so the core L2 protocol won't be able to mitigate hacks in the aforementioned manner. This is done to ensure censorship resistance at the core protocol level while enabling individual apps the flexibility to integrate semi-permissioned features out of the box.

4.3.1. Practical implementation

Considering that Pontem L2 implements the CSR NFTs approach, as described in section ”3.2.1. CSR NFTs”, the entire ownership of a DApp is controlled by a digital certificate, specifically an NFT. This NFT not only governs contract upgrades but also facilitates revenue collection. The CSR NFT can be managed by both the developer and locked in a contract overseen by a DAO, ensuring a fully decentralized approach.

The Move VM, which is already implemented in production-ready platforms like Aptos, allows for contract upgrades. There are three options for contract upgrades: immutable, compatible, and arbitrary. Immutable contracts do not support upgrades, while arbitrary contracts cannot be linked to immutable or compatible contracts.

On the other hand, compatible contracts offer the ability to perform upgrades. Developers can introduce new structures (resources), new methods, and modify method bodies, while ensuring compatibility with other contracts.

In Pontem L2, we introduce a novel method for pausing contracts published under CSR NFTs. By supplying the CSR NFTs as an argument to a specific function, contract creators have the option to pause the entire DApp and its associated withdrawals.

Pausing DApps is a straightforward process. Developers can simply check if the method belongs to a CSR NFT that is currently on the blacklist, thereby preventing any execution of the DApp, including interactions with third-party contracts.

4.3.2. Delayed withdrawals

As you may be aware, all operations involving coins/tokens, including native Aptos coins, are handled within the Aptos core framework. The existing coin module allows for the direct storage of coins within a DApp resource, as demonstrated by the LiquidityPool resource from Liquidswap:

```
struct LiquidityPool<phantom X, phantom Y, phantom Curve> has key {
      coin_x_reserve: Coin<X>,
      coin_y_reserve: Coin<Y>,
      ...
}
```

However, we propose a different approach to prevent the direct storage of Coins within a DApp resource and instead introduce a new interface. The following pseudocode represents a potential implementation (please note that this is pseudocode and not a final implementation):

```
struct CoinStorage<phantom T> {
  coins: Coin<T>,

  // Like one hour.
  withdraw_delay: u64,
  pending_withdrawls: <address, Coin<T>>,
}

// Create new storage.
public fun new_coin_storage<T>(account: &signer, csr_nft: &CSRNFT): CoinStorage<T> {
 ...
}

// Deposit into storage.
public fun deposit_into_storage<T>(storage: &mut CoinStorage<T>, coin: Coin<T>) {
  ...
}

// Delayed withdraw from storage.
public fun withdraw_request<T>(account: &signer, amount: u64): WithdrawRequest {
 ...
}

// Emergency withdraw to solve issues.
public fun withdraw_with_csr<T>(account: &signer, amount: u64): Coin<T> {
 ...
}

// Indeed withdraw.
public fun finish_withdraw<T>(account: &signer, request: WithdrawRequest): Coin<T> {
 ...
}
```

In this approach, the withdrawal time is incorporated, and pending withdrawals are accumulated in a separate structure, preventing direct withdrawals from the DApp itself. It's important to note that this design is not final and may undergo further refinement.

Implementing this approach would require developers to design their DApps differently, while simultaneously reducing the costs associated with risks. Once developers are confident that their DApp has undergone thorough security audits, such as multiple audits or services like Immunefi, they can set the withdrawal delay to zero, enabling immediate withdrawals.

Using the example of the Liquidswap codebase, swaps could still take place, but users would receive their funds only after a specified delay, such as an hour or two. This would help mitigate the possibility of hacks, particularly during the initial stages of launch (e.g., the first few weeks).

Since this concept is currently under review, there is room for exploring variations and potential improvements.

Additionally, adopting this approach would necessitate off-chain monitoring services to consistently verify the DApp's current state and ensure its proper functioning. This monitoring process is similar to how centralized exchanges maintain their own nodes to monitor the state of their reserves. External services like **Quantstamp** offer "24/7 Security Monitoring Software" specifically designed for DApps, providing continuous oversight and protection.

If you're interested and wish to delve deeper into our research, please don't hesitate to contact us. We're more than happy to provide you with additional materials and insights.

5. References

1. Alchemy Web3 Development Report - https://www.alchemy.com/blog/web3-developer-report-q3-2022
2. What's Optimistic Rollups Challenge Period - https://kelvinfichter.com/pages/thoughts/challenge-periods/
3. Danksharding - https://ethereum.org/en/roadmap/danksharding/
4. Transaction fees on L2 - https://community.optimism.io/docs/developers/build/transaction-fees/#
5. Decentralized sequencers: where do we go next? **-** https://www.alexbeckett.xyz/decentralized-sequencers-where-do-we-go-next/
6. Fraud proofs and virtual machines - https://medium.com/@cpbuckland88/fraud-proofs-and-virtual-machines-2826a3412099
7. Inside AnyTrust - https://developer.arbitrum.io/inside-anytrust
8. Introducing AnyTrust - https://medium.com/offchainlabs/introducing-anytrust-chains-cheaper-faster-l2-chains-with-minimal-trust-assumptions-31def59eb8d7