The L1 vs L2 Landscape: Understanding the Design Differences, Tradeoffs, and Endgames

Actionable Insights

- Understand the available primitives and technical limitations

: Gain a deep understanding of the fundamental building blocks and the technical nuances. This knowledge is crucial for identifying how these factors can impact the functionality and scalability of your application and ultimately business, and for making informed decisions in the blockchain development process.

- Assess infrastructure and liquidity

: Evaluate the infrastructure and liquidity available on L1 and L2 solutions alongside alternative execution environments for deployment. Make projections about the state of infrastructure from first principles. Understand the actual trust and decentralization assumptions from first principles.

- Optimize for User Experience

: Optimize relentlessly for the user experience considering the ease of use, transaction speed, and overall efficiency offered by each chain. Aim to create a seamless and user-friendly environment, recognizing what is live today and what the roadmaps will lead to.

Introduction

As the landscape of blockchains continues to evolve, the decision of where to build applications has become increasingly complex for developers. This report delves into the distinctions between Layer 1 (L1) and Layer 2 (L2) solutions regarding generalized smart contract platforms, exploring the implications for user engagement, economic margins, and business growth.

Despite the diversity and advancements within the blockchain industry, the user base remains modest compared to mainstream digital platforms. For instance, crypto currently engages around 5.5 million attributable unique addresses daily compared to the roughly 2 billion daily users of platforms like Facebook.

The customers of blockchains today are developers. Blockchains sell blockspace to application developers to build novel products. Choosing where to build your application is one of the most important choices a developer faces today. From a business perspective, choosing where to build is one of the most substantial decisions, affecting downstream potential users, liquidity, and culture. Migrating systems post-launch, from a technical and social standpoint, is a difficult endeavor, albeit not impossible. Teams such as dydx have successfully migrated from Ethereum to a Cosmos appchain, whiley00ts has faced challenges migrating from Solana to Polygon and now to Ethereum.

This report is not intended to take an opinionated stance on the latest in themodular vs integrated debate. Rather, it offers a practical view for developers, potential developers, and executives to better understand the L1/L2 landscape, with concrete examples of application tradeoffs for certain designs.

Characteristics of a Blockchain

Before discussing the differences between L1s and L2s, we must first define the characteristics of a blockchain.

A blockchain is a deterministic replicated state machine. It is a complex system designed to achieve consensus and maintain a unified, secure record across a distributed network.

- Deterministic

: The processes within a blockchain are predetermined by the underlying protocol. When a blockchain's "state transition function" (STF) is executed with a given input and the current state, it will produce the same new state across all nodes. This deterministic nature ensures that – irrespective of the individual executing the function or the node processing the transaction – the outcome will be consistent. Deterministic processes guarantee reliability and predictability in the network's operations.

- Replicated

: Blockchains leverage a network of nodes, each of which maintains a copy of the entire state of the blockchain. This replication contributes to decentralization, providing resilience against data loss and tampering. Since each node holds a replica of the state, any attempt to alter the historical record would require simultaneous modifications across a majority of nodes. This feat is practically infeasible in a sufficiently large and distributed network.

- State Machine

: At its core, a blockchain is a system that manages state – snapshots of a particular point in time encapsulating all agreed-upon information, such as account balances, smart contract states, and ownership records. The blockchain transitions through transactions, which are requests to change the state from one state to the next. These transitions are governed by the rules encoded in the blockchain's protocol.

The result of these properties is a ledger that anyone can read or write to. As Balajinoted in 2019, public blockchains are global multi-client databases where every user is a root user. This enables a global view of how many assets (tokens, NFTs, etc.) each user owns and permissionless innovation around the public global shared state.

Blockchain Marketplace Participants

Blockchains facilitate a global marketplace for agreed-upon information based on decentralized trust. There are two main stakeholders in a blockchain network:

- Trust suppliers

: stakers, validators, miners (depending on if the network isPoW or PoS)

- Trust consumers

: applications (which often bundle transactions and pass the marginal operational cost onto the user)

In other words, blockchains facilitate a two-sided marketplace between buyers and sellers, just like any other economic market. As supply and demand dictate, the market prices a cost associated with facilitating trading between the two sides.

Inspired by

[EigenLayer

](https://www.eigenlayer.xyz/)

Public blockchains need fees to prevent spam and allocate scarce resources. These fees are the primary form of compensation/revenue for validators/miners to cover overhead (equipment, electricity, and other resources). While different protocols have different fee pricing mechanisms, all major public blockchains integrate gas into the protocol in some way. Someone is running some computing equipment somewhere in the world and must be compensated to be incentivized.

Defining L1s and L2s

With a better understanding of what a blockchain is and what it enables, we can formalize a definition of what an L1 and L2 is.

A comprehensive, rigorous definition of L1s and L2s and their respective connotations is a contentious debate amongst researchers and the broader crypto community. Different flavors of rollups have emerged (optimistic, zero-knowledge, enshrined) with differing implementation details and trust assumptions (validiums, volitions, sovereign, amongst others).

Before discussing the tradeoffs between L1s and L2s, we must define them:

Layer 1s (L1s)

: L1 blockchains are foundational networks that operate as the primary infrastructure for a cryptocurrency. These are standalone networks with their own underlying protocols, consensus mechanisms, and native tokens. L1s are often referred to as the base layer of a blockchain ecosystem because they provide the fundamental security and governance framework.

Select L1s

Ethereum pioneered the concept of modular trust when it launched in 2015. However, Ethereum was not and still is not optimized for the scale required to onboard millions of users. As a result, new scaling solutions such as rollups emerged as a promising solution to scale Ethereum.

Layer 2s (L2s)

: L2 solutions are built on top of L1 blockchains to enhance scalability and transaction throughput without altering the base layer. Rollups are one of the most promising and researched L2 solutions.

Select L2s

Rollups offload transactions from the L1, process them separately, and then record the final state back on the L1 (often in the form of a state root). This approach allows the rollup itself to execute additional transactions in a separate state environment at a lower cost than could be achieved directly on the L1. Rollups bundle or "roll up" multiple transactions into a

single transaction on the L1 to reduce load and improve efficiency. Different L2 solutions have different trust assumptions and generally

aim to inherit the security and decentralization aspects of the underlying L1, making them secure but more scalable alternatives to processing every transaction on the L1.

Celestia has formalized this definition for rollups:

Source

In practice, Ethereum has the most interesting and nominally valuable assets in crypto. Users have shown a willingness to bridge to various L2 solutions without much concern for some of the more centralizing aspects of the L2 (which are actively being developed but are often not live yet) for the much lower fees than Ethereum mainnet.

Security is a common concern when bridging assets between chains. Although many prominent rollup bridges lack rigorous proof systems and have been critiqued for potential centralized control points, they offer a seamless extension of the Ethereum experience (via a similar EVM-type experience and wallet uniformity). Rollups such as Base, Optimism, and Arbitrum maintain a user's existing Ethereum wallet and address infrastructure to the best of their ability, making transitions between the main chain and L2s much more seamless. This convenience, coupled with the dominance of ETH in DeFi and social applications, makes L2 an attractive proposition for users seeking familiarity and ease of use. Ethereum has the most popular state

that other developers want to interact with, and L2s are viewed as a natural extension of Ethereum.

This is all to say that "inheriting" certain security guarantees from Ethereum is far down the list of the reasons why a team chooses to launch as an Ethereum L2 or migrate from a standalone L1 to an L2

New developments such as Solang and Neon are bridging the gap for developers accustomed to Ethereum's Solidity, while Metamask's Snaps extends wallet support to non-EVM chains. These developments signal a future where the UX effectively abstracts many complexities across different execution environments.

While the Ethereum Foundation has underwritten much of the research around L2s, all L2s are independent entities and are not operated by the Ethereum Foundation. Permissionless innovation is not unique to the decentralized application layer, and no credibly neutral L1 has the technical ability to decide which L2s can deploy on top of a given L1.

"Typical" L1 and L2 Implementations/Architecture

To best understand the tradeoffs between L1s and L2s, it is crucial to understand their respective supply chains, relevant actors, and trust assumptions.

Lifecycle of an L1 Transaction

While each L1 has unique implementation details, all major L1s have the same relative stakeholders and transaction lifecycle. This section outlines the common transaction flow from generation to eventual inclusion.

1. Intent

: The user begins with a clear goal or action they want to achieve on the Ethereum network, such as sending ETH, interacting with a smart contract, or swapping tokens. This stage involves the user's decision-making process and understanding of what they want to accomplish, outlining the transaction's journey.

1. Transaction Generation

: Here, the transaction takes form. The user, often through a user interface like a wallet or decentralized application, inputs the necessary information, such as the recipient's address, the amount to send, or the contract method to call. This stage translates the user's intent into a structured request that the Ethereum network can process. At this juncture, the user determines the cost of the transaction, which includes the gas price and gas limit. This is highly dependent on the chain and is often abstracted away by the wallet provider. Both Ethereum and Solana implement a version of this via a base fee and a priority fee (tip), with differing implementations and levels of determinism with respect to transaction inclusion.

1. Transmission

: The user signs the transaction with their private key, a digital signature proving their authorization. Once signed, the transaction is broadcasted to the network (typically via an RPC provider integrated with the wallet), usually through the public mempool or via private orderflow auctions where it awaits processing.

1. Inclusion Confirmation

: The transaction reaches its conclusion in this last stage. The user receives confirmation that the transaction has been successfully included in a block (with varying finality levels depending on the specific protocol). Finality brings closure, allowing users to verify the outcomes, such as updated balances or contract states.

A diagram of the lifecycle of a Solana transaction, from genesis to inclusion

The Ethereum Supply Chain

Ethereum, aiming to be a credibly neutral infrastructure layer for censorship-resistant applications, has undergone significant evolution in its supply chain mechanism. Initially operating under Proof of Work (PoW), miners controlled Ethereum's transaction inclusion and ordering. This led to the emergence of priority gas auctions (PGAs), where searchers competed for block space, resulting in mempool bloat and inefficient chain usage. The introduction of Flashbots Auction, an out-of-protocol relay for MEV bundles, mitigated some centralizing effects of MEV by moving MEV markets off-chain, though it relied on the honesty of both the relay operator and miners.

With the transition to Proof of Stake (PoS), Ethereum's block production shifted to staked validators, introducing new dynamics and potential centralization issues. Under PoS, economies of scale in MEV extraction could lead to stake centralization, threatening Ethereum's censorship resistance.

To address concerns of stake centralization, Proposer-Builder Separation (PBS) was introduced byFlashbots. It was initially implemented as MEV-Boost, a novel regime for block production. In this system, searchers send bundles to block builders, who then build blocks and bid through block relays to the proposer. MEV-Boost moves centralizing forces to the block builder layer, requiring trust in block builders and relays. As of December 2023, over 93% of Ethereum blocks are proposed by validators running MEV-Boost.

Source: Flashbots

However, MEV-Boost and PBS are not without issues. The MEV-boost relay, run by Flashbots, enforcesOFAC compliance, an undesirable equilibrium for a fair, permissionless system. The centralization of relays poses a risk of censorship and dishonest behavior as relays become critical intermediaries between builders and validators. There is also a heavy reliance on out-of-protocol software like MEV-Boost, increasing Ethereum's vulnerability to exploits and attacks, as evidenced by incidents like the low-carb-crusader exploit. There are ongoing discussions toenshrine PBS (ePBS) into the core Ethereum protocol, borrowing much of the work underwritten by Flashbots. Builder centralization is another concern, with a few builders dominating the market, potentially leading to censorship and influence over the network. New solutions such as SUAVE aim to alleviate these concerns.

The Solana Supply Chain

Launched in 2020, Solana, a separate L1, broughtnumerous advancements to solve the scaling problem on the base layer without any additional third-party solutions such as L2s. Solana's laser focus on allowing applications to "share state" with each other enables native composability between applications with minimal work from application developers and no additional complex user operations or trust assumptions as is the case with bridging.

On Solana, a transaction consists of three main components: a set of instructions dictating the transaction's operations, a list of accounts with associated permissions that the transaction will interact with, and the necessary signatures to authenticate the transaction. Once the user confirms the transaction, it is sent to a Solana RPC server, which forwards it to the current network leader responsible for block production. This is known as Gulf Stream, one of the eight unique innovations that Solana enables.

Source: Umbra Research

Upon receipt, the leader validates the transaction and schedules it for execution. Solana's parallel execution model enables it to process multiple transactions simultaneously, enabling parallelism and "local fee markets". This is achieved through a multi-threaded scheduler where transactions are queued based on priority fees and time.

After execution, the leader records the transaction in the ledger and disseminates it across the network. The transaction's status is then updated in stages: from execution to "confirmed" upon acquiring sufficient consensus votes and finally "finalized" after being securely built upon by the required number of additional blocks (31+).

Unlike Ethereum, which empirically has much more robust guarantees for transaction ordering based on fees, Solana does not have a deterministic ordering of transactions. Although transactions offering higher priority fees are generally more likely to be placed higher in a block, and competitive state situations often necessitate higher fees, this outcome is not guaranteed. The likelihood of such occurrences is influenced by the non-deterministic nature of Solana's default scheduling algorithm, which determines transaction inclusion and positioning within a block. This incentivizes out-of-protocol activity, including searcher-validator side-deals, which incurs negative externalities to the median user.

To be sure, MEV on Ethereum is a much larger market, with over$750 million (373,701 ETH) extracted since the Merge. As a result, more time and resources have been allocated towards MEV, which is significantly more well-understood and researched with incentive-compatible mechanisms.

This is one of the core philosophical differences between Ethereum and Solana and manifests in each ecosystem's

resource allocation and culture. While Ethereum invests heavily into researching and designing incentive-compatible mechanisms, such as with PBS (with ongoing discussions around ePBS), the core Solana protocol is designed specifically to synchronize global state and defers to ecosystem teams such as Jito for specialized ordering services and guarantees. Anatoly (CEO of Solana Labs) is clear in his view that the current state of Solana is a better (temporary) equilibrium for the median user.

User Experience (UX) on L1s

From a user's perspective, submitting a transaction to a Layer 1 blockchain, such as Ethereum or Solana, is one of the most straightforward actions in crypto, typically involving the following steps:

1. Wallet Setup

: The user must first download a wallet. This can be a software wallet like MetaMask, a mobile app, or a hardware wallet for higher security.

1. Acquire the L1's native token

(ETH, SOL, etc.): Ensure your wallet has enough ETH/SOL to cover the transaction and gas fees. You can buy ETH on various centralized exchanges. Teams and protocols are working on account abstraction; one of the features of account abstraction is to pay for gas in any arbitrary token, not just the native token of that L1. Ethereum implements this via EIP-4337. Solana has native account abstraction built into the protocol, and a gasless transaction relayer called Octane.

1. Enter Transaction Details

: Access your wallet and choose the option to send ETH or tokens. Enter the recipient's Ethereum address, the amount of ETH or number of tokens you want to send, and any additional data if required (for smart contract interactions). If interacting with an application, the transaction is usually pre-packaged for you by the application developer.

1. Set Gas Price and Limit

: The wallet will set these automatically or allow you to choose. The exact gas implementation will differ depending on which L1 you are interacting with.

1. Sign and Submit

: Sign the transaction with your wallet (this process varies depending on the wallet). Your wallet will then broadcast the transaction to the network (usually via an RPC node).

1. Confirmation

: Once your transaction is included in a block and added to the blockchain, it's confirmed. Most wallets will notify you of the transaction's completion, and you can also track it using a blockchain explorer like Etherscan or Solscan.

User Experience (UX) on L2s

There are distinct UX differences when interacting with an L2. This is how a new user would onboard onto Optimism (one of the largest Ethereum L2s):

1. Download a wallet

: If you don't have one, download and set up a wallet. Users typically use the same wallet across EVM ecosystems such as Metamask.

1. Add the Optimism network

:

1. Deposit ETH into Optimism
2. For large L2s, this can often be done natively inside the wallet.
3. Alternatively, you can visit the official Optimism bridge.

4. Choose the amount of ETH to bridge to Optimism and confirm the transaction.
5. Switch to Optimism Network in Your Wallet

: Open your wallet and select "Optimism" from the network dropdown menu.

1. Interact with Optimism-compatible applications and front-ends

: Choose “Optimism” for applications that support the Optimism network.

UX advancements have been made in L2s, specifically around native assets and onboarding. Circle (which operates USDC) has native versions of USDC on most major L2s, eliminating the additional counterparty risk typically associated with bridges and wrapped tokens. Additionally, Coinbase’s distribution advantage has allowed for tight integrations between the Coinbase platform and Base – an Ethereum L2 of which they are the sole sequencer operator.

L2 Components

After bridging funds to a given L2, which can be obscure and complicated depending on which L2 you’re using, the user experience while transacting on an L2 is often similar to an L1. While L2s in the past have primarily utilized Ethereum for different aspects of the blockchain stack, many L2s today are leaning toward a “modular” approach. This modular approach allows a mix-and-matching of different ecosystems and infrastructures, separating the key components of an L2: execution

, settlement

, and data availability

. Previously, L2s utilized Ethereum for everything except execution, but this is rapidly changing with new settlement and data availability solutions. L1s, on the other hand, offer each of these components in an integrated, opinionated manner with no ability for developers to choose a desired architecture without governance approval.

Execution

The goal of execution is to run a state transition function over the pre-state to yield the post-state. Different virtual machines (VMs) like Ethereum Virtual Machine (EVM), Solana's Sealevel (SVM), or others like MoveVM and FuelVM are employed as standards to govern these state transitions. Each VM offers unique features affecting compatibility, developer tools availability, and efficiency in processing transactions. They create the environment for executing smart contracts and protocols.

Interacting with the execution layer is where application developers actually deploy code, transactions occur, and users interact with the blockchain’s state. This layer is designed to process transactions rapidly while delegating the responsibilities of security and data availability to other specialized layers.

Source

There is no requirement for the L2 virtual machine to match the L1 virtual machine. Although this has historically been the case in the past (most L2s are EVM or EVM-compatible rollups), more teams are exploring this design space with different virtual machines “settling” to Ethereum (Eclipse, Movement Labs).

Settlement

The word “settlement” is viewed by many to be anoverloaded term at this point. When understanding settlement, it is important to distinguish between the point of view of the “rollup” and the “bridge”. Bridges often adhere to a 1-week finalization process, but the bridge is not the rollup. From the rollup’s perspective, finalization occurs after checking the state transition and the relevant data has been posted to a data availability layer.

Settlement layers serve several key functions for rollups:

- Verification and Dispute Management

: They act as platforms where rollups can post their proofs for external validation, which is crucial for optimistic rollups that depend on interactive fault proofs.

- Bridging Hub

: By acting as a central point, settlement layers enable rollups to interconnect, eliminating the need for individual bridges between each rollup.

- Source of Liquidity

: The liquidity present on the settlement layer can be accessed by all associated rollups, facilitating smoother transactions and interactions.

As mentioned, Ethereum has the most interesting and nominally valuable assets in crypto – people want to interact with the state on Ethereum. Ethereum is being used as a "settlement layer" because users have shown a strong preference for interacting with an enshrined bridge to a separate chain (the Ethereum L2 they are interacting with). L1 to L1 bridges have not been viewed by the market in the same way, likely due to previous hacks and a more complicated UX (L1 to L2 bridging can often be tightly integrated within a single wallet). The UX is rapidly improving with wallets such as Phantom supporting Ethereum and Solana, as well as Metamask releasing Snaps. In more formal terminology, users are comfortable with interacting with the enshrined bridge that a given rollup defines.

User behavior is rapidly changing. Wormhole has moved over $500M in volume over the last 30 days, closing the gap to the Optimism 30-day bridge volume of $825M.

One can argue that the L1-L1 bridges have had a hard time due to path dependency, with large hacks such as the Wormhole hack in early 2022. While there are unique engineering trust assumptions enabled by sharing DA, the actual implementations of L1-L2 and L1-L1 bridges are often secured by a multisig.

The main takeaway here is that rollups "settle to Ethereum" (or more accurately, finalize an enshrined bridge to Ethereum) for access to Ethereum state

for the express purpose of users and liquidity. Settlement itself is not a profitable endeavor for Ethereum, asrollups such as Optimism pay less than $50 per day in ETH to Ethereum for settlement. Over 80% of rollup costs go towards publishing data to Ethereum L1 blocks as calldata. This is why we see so much competition around data availability solutions today. In total, L2s make up approximately ~6% of Ethereum L1 fees (using Ethereum for data availability).

L2s make up approximately ~6% of total Ethereum L1 fees

Data Availability (DA)

Data availability in blockchains refers to the assurance that all data in a block is actually published and accessible to the network. This concept is crucial for the functioning and security of blockchains, especially when scaling them. Typically, a block consists of a header and the transaction data. Full nodes download and validate every transaction, while light clients, which are less resource-intensive, rely on only the block header and assume that the transactions are valid. The data availability problem arises when a block producer publishes a block header but withholds some or all of the transaction data.

EigenDA's design

Although it has not happened yet, data withholding attacks pose a threat to blockchain networks. In such an attack, a malicious block producer deliberately withholds a portion of transaction data from a new block. This hinders other network participants from validating the block's transactions. The consequences of data withholding can range from network disruption to more severe issues like enabling fraudulent transactions. The severity and impact of these attacks vary based on the blockchain's structure (Layer 1 or Layer 2) and whether data availability is managed on-chain or off-chain. The risk underscores the importance of robust mechanisms for ensuring data availability and the necessity of effective strategies to mitigate such attacks.

Solving the data availability problem involves methods such as all full nodesdownloading all the data, which can be resource-intensive, or innovative approaches like data availability proofs and data availability sampling (DAS). These proofs use techniques such as erasure coding, allowing clients to verify data availability by downloading only a small part of the block.

A true "Ethereum rollup" – according to the most up-to-date terminology – requires using Ethereum for data availability. This means the rollup must post transaction data to the Ethereum L1 as calldata to be considered a rollup. Ethereum is currently working on implementing EIP-4844 which will create a separate market for blobspace

(transaction data) and allow data blobs to be posted at a further discounted rate. Even when 4844 is live, it will only increase data availability by a factor of ~7, which translates to around 100 TPS (transactions per second) across all rollups. For context, Solana's demand for blockspace sits at around 500 TPS and has demonstrated the ability to handleover 2000 TPS. Furthermore, Solana is designed to scale with hardware (Moore's Law) and software advancements such as Firedancer.

Scroll

Though data availability may exhibit some small network effects, competition will likely drive the cost to post data to a data availability solution very close to the marginal cost of operation.

DA ensures all necessary transaction data is accessible for reconstructing the state of an L2 (over a predefined time period). In the short term, validiums and volitions utilize separate data availability layers, while Ethereum's long-term roadmap includes an improved data availability solution through upgrades such as EIP-4844. Alternative data availability solutions (e.g., Polygon Avail, Celestia, and EigenDA) are expected to play significant roles and further explore different DA designs. However, it is important to note that these data availability solutions are not yet live, with the exception of Celestia which just launched their mainnet but have no rollups posting data yet. They are all built on novel systems and are not yet proven to work at scale in production, with a wide spectrum of unique security and trust assumptions. These solutions offer an interim fix until Ethereum's comprehensive data availability layer becomes operational, and might demonstrate network effects even

after Ethereum integration.

Lifecycle of an L2 Transaction

The lifecycle of a transaction on Arbitrum, a Layer 2 protocol, involves several stages from initiation to confirmation and finalization:

Source

1. Sequencer Receives Transaction

: Transactions start with the Sequencer, either received directly/off-chain for L2 transactions or from L1 via the Delayed Inbox for deposits.

1. Sequencer Orders Transaction Off-Chain

: The Sequencer orders the transaction in its off-chain Inbox and executes it using the Arbitrum Nitro VM, providing an "instant" receipt to the client. This phase's finality depends on trusting the Sequencer.

1. Sequencer Posts Transaction in a Batch On-Chain

: The Sequencer posts batches of L2 transactions onto L1 at regular intervals including the relevant data as calldata. If Sequencer fails to include a transaction, the client can force it to be included after a delay (i.e., forced inclusion). At this stage, if there is at least one well-behaved active Arbitrum validator, the transaction's finality is equivalent to an Ethereum transaction.

1. Validator Asserts Rollup Block (RBlock) Including Transaction

: Validators assert the state of the L2 chain on L1 at regular intervals. RBlock assertions include claims about the state of the Outbox for L2 to L1 messages.If Challenged

: In case of a dispute, validators engage in a detailed verification process down to a single operation, refereed by contracts on L1.

1. RBlock Confirmed on L1

: After dispute resolution and a sufficient time window, the RBlock is confirmed on L1, updating the Outbox root.

This process ensures that once an L1 transaction recording a batch is confirmed, the L2 transaction within it has the same level of finality as a regular Ethereum transaction. If the transaction includes L2 to L1 messages, they become executable on L1 only after RBlock confirmation.

Because rollups exist outside of Ethereum (not to be confused with the fact that many rollups today are built on the EVM or are EVM-equivalent), there are additional considerations and involved parties in the transaction supply chain. The following characteristics of Arbitrum and Optimism, the dominant rollups on Ethereum today:

- Centralized Sequencer

: Arbitrum and Optimism, the largest optimistic rollups for Ethereum, currently operate with a single centralized Sequencer. This Sequencer is responsible for ordering all transactions. Future plans involve decentralizing this role with potentially community-voted L2 nodes, but details and timelines are still unclear. Users trust the sequencer not to sell order flow, reorder transactions, or otherwise extract MEV. Fault proofs are on the roadmap but are still not live for permissionless challenges.

- Trust Assumption

: Finality at this stage depends on trusting the Sequencer. The Sequencer's role is limited to ordering or delaying transactions, not altering them. Arbitrum allows transactions delayed over 24 hours to be force-included, while Optimism automatically includes L1 transactions in the next available L2 block.

- L1 Re-orgs

: If an L1 reorg occurs, the policy for L2 block continuation and handling is not explicitly detailed. This leaves some uncertainty in transaction finality during L1 reorgs.

- Finality

: Sequencers batch L2 transactions and post them on Ethereum as calldata, achieving “hard finality” equivalent to the finality of the Ethereum block in which they're included. The L2 state tree’s root, representing the rollup’s latest state, is then updated on L1, ensuring irreversible state changes.

- Challenge Period and Withdrawal: Validators on Arbitrum can challenge state roots, initiating a dispute resolution

process. If a state commitment remains unchallenged during the designated period, it becomes final. Post-challenge period, withdrawals from L2 to L1 based on the final state are allowed.

- Bridging: Bridging protocols consider different finality levels, from instant receipts under “soft finality” to batched calldata under “hard finality”. Bridges can choose faster messaging based on client needs or adhere to stricter finality standards, depending on the transaction type and risk assessment.

Where L2s Make Sense

L2s are stateful systems. Each additional L2 fragments state across the differing L2s. While atomic composability is sacrificed (which may or may not be overrated), deploying to an L2 makes sense for some applications. Furthermore, choosing the correct L2 is very important even if they run on a popular execution environment such as the EVM. Different L2s have vastly different user experiences, integrations, and distribution.

Without getting too technical, L2s can broadly be separated into application-specific rollups

and general-purpose rollups

.

Application-Specific Rollups

Application-specific rollups, unlike L1s and other previously mentioned L2s, do not share blockspace with other applications. Instead, they are the dedicated blockchains for a specific application (or set of applications). As a result, you are responsible for running all of the corresponding infrastructure.

While some teams are vertically integrated and effectively run their infrastructure (dydx), this is a massive engineering undertaking and generally only successfully performed by teams with prior product-market fit (PMF). In the case of dydx, they have product-market fit, a strong community, and enough resources to do so (after finding PMF on Ethereum and later on an Ethereum L2). Other projects such as Maker are exploring designs such as an SVM app-chain. In practice, most teams do not run their own hardware; instead, they rely on rollup-as-a-service (RaaS) providers that aim to fill this void, such as Caldera and Conduit.

It is an open question if application-specific chains will be able to generate enough revenue to be profitable in the long run. In these cases, it is important to understand the counterparty risk and cost structure of RaaS providers. RaaS providers often charge a fixed fee (often between $5,000 and $100,000, depending on the application size) and a percentage of the sequencer profits. Most RaaS providers leverage well-accepted standards such as the OP Stack to avoid “lock-in”, meaning that even if the RaaS provider goes under, you can still run your own hardware, as long as you can overcome the short-term liveness failure.

Because gaming is a more specialized use case, novel implementations and protocols are being actively developed such as Argus’s World Engine. On December 5th, Dark Frontiers launched as an execution shard on top of Argus’s EVM L2 for fully on-chain games, unlike other on-chain games that exist today that only post simple game items on-chain. While yet to be seen, this seems to live somewhere between application-specific rollups and general-purpose rollups.

General-Purpose Rollups

Being an isolated environment with less economic capital at stake, rollups are somewhat incentivized to experiment with certain layers in the stack, such as with execution and consensus level to offer a differentiated product. However, most rollups are off-the-shelf versions of existing VMs (with the exception of Fuel). This is because they are incentivized to make it as easy as possible for existing applications to deploy to their L2. This has pushed teams innovating at the execution and consensus level towards launching as their own L1, such as Monad.

More experimentation at the execution layer may be on the horizon as the number of rollups far exceeds a reasonable equilibrium, especially for the rollups struggling to attract users and liquidity.

Current State of General-Purpose Rollups

General-purpose rollups such as Arbitrum, Optimism, and Base have exhibited strong network effects and have attracted numerous existing and novel applications to launch on the L2s such as GMX and friend.tech. The presence of Ethereum’s native assets and state on these L2s – often using Ethereum to denominate their gas token – ensures that users and developers have access to a rich and mature ecosystem. This access is vital for applications seeking liquidity and with existing Ethereum-based protocols and assets. Regarding user experience, platforms like Coinbase have integrated native wallet and payment support for Base (where they operate the sole sequencer), making the interaction seamless and user-friendly.

When considering various L2s, it is important to consider alternative execution environments to deploy to, liquidity, and the L2 infrastructure itself. Factors like the cost structure of data availability, potential future changes in network fees, and the evolving landscape of infrastructure providers should inform the decision-making process. Application developers should

understand the cost structure for users both now and make reasonable projections about the future based on factors, particularly the cost of data availability. For developers considering L2 deployment, it's crucial to understand the nuances of systems, paying particular attention to the execution environment of choice and alternative deployment opportunities.

Novel research led by the Ethereum Foundation such as with L1-sequenced "based rollups" means that the rollups will continue to improve with respect to liveness, decentralization, and user experience. Decentralized sequencers – or at least a multiple sequencer rotation for liveness – are still in the research phase but offer promising advancements in ensuring reliability and reducing the current state of sequencer risk. Today, users trust the sequencer run by the L2 core team to fairly order transactions and not maliciously extract MEV by re-ordering transactions. Users can also submit transactions directly to the L1 via forced inclusion in case of sequencer censorship.

On the other hand, the rollups that exist today are missing some key features that are actively being developed according to their roadmaps. For example, none of the major optimistic rollups offer permissionless fault proofs and are often gated behind feature upgradeable contracts via multisigs. Additionally, many rollups lack escape hatches for users to withdraw funds under a malicious sequencer.

The General-Purpose Rollup Endgame

A general-purpose rollup endgame with a "beefy" sequencer (or set of sequencers) is one possible outcome. In a world where Ethereum retains its position as the chain with the most interesting state and users continue to strongly prefer L1 to L2 bridges, a general-purpose rollup L2(s) can offer significant execution advantages for a much better user experience.

The L1 would effectively only exist to serve the desired properties that the rollup requires, eliminating L1 re-org risk to the L2, and allowing the L2 to optimize for the dimensions that end users care about: latency

and updating state as efficiently as possible, typically in the form of better prices

. Ethereum – the L1 of choice – still plays a role as social consensus still derives value from Ethereum state for memetic reasons (credibly decentralized parties involved, fair launch, etc.). The L2 exists as a foreign entity to the L1.

The rollup(s) will exhibit the following features and properties:

- Permissionless innovation

– anyone can build and deploy new applications

- Censorship resistance

– via forced inclusion on the L1

- Data availability

– either directly on the L1 or a separate DA layer

- Validity

– fault or zk-proofs

- Parallelized execution environment

– transactions are processed in an extremely efficient manner

In this endgame, there are a few (2?) rollups that have meaningful users and volume. The marginal application chooses to deploy to this rollup because the cost to share blockspace (amortized to as close to zero as possible) is outweighed by sheer access to users. Revealed user preferences indicate a high priority of the simplest UX. If we ever reach this scenario, crypto has effectively re-invented traditional finance with multiple key improvements: permissionless innovation, censorship resistance, non-custodial, and transparency.

Only a small set of actors has write access to traditional finance markets; NYSE and other exchanges are tightly regulated and permissioned markets that can censor individuals as laws and regulations dictate.

In this scenario, latency and co-location with sequencers would still exist as in traditional finance with order book servers. Currently, permissioned access dictated by politics and path dependency has determined which entities have the right to co-locate in NYSE's servers. Citadel's decades of expertise optimizing infrastructure and algorithms that provide end users better prices still applies in a permissionless setting. In this rollup endgame, individuals and teams would instead compete against every other talented individual in the world strictly on performance, resulting in better prices and end-user experiences.

While this all sounds good in theory, the state of L2s today paints a much different picture. Not only are the only sequencer(s) run by the core team – a single liveness point of failure – they often gate the most important actions whether that be the bridge contract or permissioned fault proofs. While this is on the roadmap for many L2s to decentralize and

remove the multisig, this is not the state of affairs today.

Where L1s Make Sense

The concept of shared state on Layer 1 blockchains, like Solana and Ethereum, enables permissionless composability, meaning that applications can interact with each other in a trustless and secure manner. One way this is useful is for marketing strategies such as targeted airdrops. For example, by leveraging the shared state, a project can identify and reward specific NFT communities, enabling highly focused and effective reward mechanisms. Pyth, a multi-chain oracle for data, recently airdropped $PYTH on Solana to anyone who has interacted with their protocol across 40+ blockchains. By analyzing ownership patterns and community engagement within the blockchain, projects can tailor their rewards, such as exclusive tokens or access to new services, to incentivize and engage with precise segments of the blockchain community, thus creating more impactful and resonant marketing initiatives.

Shared state uniquely enables these on-chain attributions for rewards and actions. Jupiter, a unified trading aggregator that guarantees users the best possible prices, is an example of an application that is only possible on a chain with shared state such as Solana.

A sub-$3 transaction that routes via 3 hops – on separate trading venues that Jupiter does not operate – always guarantees users the best price.

In combination with DeFi, payments are one of the most important use cases for crypto, allowing users to send over arbitrary amounts of value without extractive middlemen like Western Union. Payments encompass both peer-to-peer transactions like Venmo and business-to-consumer transactions like online shopping. Importantly, payments should be so cheap and fast that users do not have to consider gas costs when sending money. On Solana, over half of USDC transfers are less than $5, which is achievable only on extremely low-gas blockchains such as Solana.

Payments on Solana

This allows new businesses to be built on top such asTiplink, which allows people to send money using a URL with no wallet setup required (you can even sign up with your Google credentials). Shopify has integrated with Solana Pay to enable shoppers to check out by paying in USDC on Solana. Visa chose to integrate with Solana to settle USDC for global merchant acquirers Worldpay and Nuvei. Code has used durable nonces to create a pseudo-asynchronous execution environment to build an extremely fast payment notification system and seamless user experience. While some people call this an L2, it does not have fault proofs and is purely optimized to create a better end-user experience.

The reason Solana does not have L2s is that transactions are already so cheap and natively separate and price state within a given block. Solana has been optimized from the beginning to bring synchronized state as fast as physics allows and to bring the cost of a transaction as close to the marginal cost of operation. On Ethereum, transactions are much more expensive in part due to the single-threaded EVM, state can only be priced separately when there is another instance of the virtual machine running, which is what L2s do. Solana and other parallel VM blockchains enable local fee markets, where different pieces of state have different fees based on how contentious a particular piece of state is. A localized state hotspot can be managed without escalating contention or fees across the entire blockchain. In practice, this means that an NFT mint or an airdrop has little externalities to your separate application.

Unlike L2s, the endgame for Solana is not on the roadmap – it's here today. While Solana is being actively maintained and improved via SIMDs, it is a functional, credibly neutral asset ledger that requires no additional trust in multisigs, alignment, or vibes.

What is the Best Solution for Me?

Well, it depends on what you are optimizing for. As a general rule, an application developer or business owner should spend the minimum necessary time on blockchain infrastructure maintenance and spend the most time iterating on your product to create the best user experience. The question is how much is necessary for you

?

When choosing a specific chain, the primitives available on that chain should be carefully studied such as with applications such as Drip that have over 60M NFTs minted and 500,000 collectors.

What you're looking to optimize for will ultimately narrow your choice of chain. Based on the above summary of the various L1s and L2s, these are some dimensions to consider:

1. On-chain users
2. On-chain usage (if you're optimizing for a specific on-chain function such as trading, then you'll want to examine the chains that have a low-fee environment – like Solana, Sui, or Optimism. You will want to understand the technical

tradeoffs and limitations of shorter block times, larger block sizes, etc.)

3. Execution time
4. On-chain liquidity
5. On-chain whales/liquidity (e.g., the EVM ecosystem tends to have more liquidity due to the prior era of token launches but this is rapidly changing)
6. Access to specific on-chain communities (artists, traders, etc.)
7. Grants and VC availability (newer ecosystems have more grant capital but lesser VC capital)
8. Technical complexity and overhead of mgmt (e.g., Code only needs 2 engineers to run their application Solana vs. dydx that needs over 30 engineers and researchers)
9. Programmability and sovereignty in chain rules (e.g., idiosyncratic trade settlement rules)
10. Access to institutional players (e.g., Coinbase on Base and Visa and Shopify on Solana)
11. Composability with other ecosystem applications (e.g. DeFi, payment apps)

Deploying to specific rollup constructions can make sense for certain applications that understand and uniquely leverage the aforementioned properties. For most applications, Solana provides the necessary primitives for the application to succeed. Its unique design decision has enabled it to be fast, cheap, and performant, and has been live in production for many years.

Conclusion

Understanding and navigating the blockchain landscape, particularly the choice between Layer 1 (L1) and Layer 2 (L2) platforms, requires a nuanced understanding of the trade-offs involved. The choice between L1 and L2 is more than a technical decision—it's a critical business strategy that influences scalability, security, and the overall viability of a project. Refactoring an entire codebase will further incur more technical debt and decrease your probability of building a successful application.

The role of the user experience in blockchain adoption cannot be overstated. As the technology matures, the focus should increasingly shift towards creating intuitive, seamless, and user-friendly interfaces that can bridge the gap between complex blockchain infrastructure and the end user. While different applications benefit from different levels of flexibility, most application developers should choose a platform that allows them to focus the maximum amount of time on building the best product possible.

Thanks to Akshay, 0xIchigo, and Jon Charbonneau for review and comments.