Rollups Are L1s (& L2s)

Originally posted to Mirror.

Thank you Kelvin Fichter for the awesome discussions getting philosophical on rollups, and for review of this post. Also thank you for the cartoons and starting the bridge ≠ rollup crusade lately.

Thank you Mustafa Al-Bassam, Christopher Goes, James Prestwich, Nick White, & Cem Özer for feedback and related conversations as well.

And a huge thanks to Toghrul Maharramov for review and fun back and forths! Especially since we disagreed on a bunch of stuff here

Editor's Note - The original post started with "The prevalent understanding of rollups is a lie." While I strongly believe in the rest of my points throughout the post, I don't want that to get lost here. I often write in jest, and I misjudged that this opening with the meme-y title would be taken quite literally by many. I acknowledge this would be confusing for the casual reader, as I also redefine some terms throughout in a manner which I have found useful for my own mental models.

My key point is that there are a lot of misconceptions out there that social consensus and "sovereignty" only apply to traditional "sovereign rollups" or other L1s. I'm highlighting that this very much still applies for smart contract rollups too. While in practice these contentious hard forks (e.g., bridge and social consensus diverging) will be incredibly rare to ever be exercised, the fundamental existence of it is critical.

For example, it's incredibly unlikely that Ethereum would disagree with Circle in a hard fork, but it's absolutely critical that it can. In the same way Ethereum is not "defined by" Circle, I push back on framing that "the rollup is defined by the bridge" because I think it's reductive and can be quite misleading.

It's also gotten near impossible to keep up (even for myself) as new definitions of"technical sovereignty" and "social sovereignty" have arisen even prior to my piece. It's confusing for anyone that there are now plans for so-called "sovereign rollups" with these validating bridges to the base layer, because they're architected very differently and the technology keeps changing.

With all that in mind, I hope you find this piece and thefollow-up post helpful.

Introduction

I believe the prevalent understanding of rollups needs to be meaningfully reconsidered. As I noted in my "Rollups Aren't Real" post:

There's an interesting argument that there is no such thing as a globally canonical chain (only bridges deciding which chain to consider canonical), good counter arguments, and other great related Twitter threads on thetradeoffs all rollups make between sovereignty and (automatic) composability.

Kelvin and Toghrul have also been going full PvP on this topic lately:

- Kelvin - [How Rollups *actually*

work a.k.a. ZK Rollups aren't real](https://youtu.be/NKQz9jU0ftg)

- Toghrul - Rollups Through the Prism of Validating Bridges or how rollups*actually actually* work

- Kelvin - Rollups Aren't Real a.k.a. There is no such thing as a Rollup blockchain

Well, I've decided to join the fight. Someone who built half the stuff we use just told me that "everyone is going to hate your argument because it doesn't pump anyone's bags." Should be fun.

This is how rollups *actually actually actually* work.

Blockchains 101

I'll very briefly start from the absolute basics to build up the correct mental model. Blockchains are "replicated state machines":

- Replicated

- Many nodes hold a copy of the blockchain's state.

- State

- The state is like a snapshot of all the stuff we agree on today (e.g., what addresses exist, who holds what amount of money, etc.).

- Machine

- The blockchain does stuff to modify the state over time. The whole "world computer" thing.

So, blockchains are just a fancy tool for everybody to agree on some state of the world and update it. Everyone applies the "state transition function" (STF) to update the state. The STF is:

- A mathematical function (representable as a program)
- Takes the current state + some input → generates new state (e.g., take the current Ethereum state → execute some transactions → calculate the new Ethereum state)
- f(Input, S_current) = S_next

Importantly, the STF is deterministic

- every node will always get the same result if they run it over the same input and current state.

Rollups 101

Throughout this report, I will use the following definition for rollups:

"Rollups are blockchains that post their blocks to another blockchain, and inherit the consensus and data availability (DA) of that blockchain."

Putting things together, a rollup comes from:

1. Input Data

- An ordered array of input data posted to another blockchain (the DA layer)

1. Function

- Agreed upon rollup node software is run deterministically over the input data

1. Output Data

- Running the function over the inputs produces deterministic outputs (the rollup blockchain) which gives us the current state

That's it. The rest is all ~implementation details~.

Sovereign Rollups vs. Smart Contract Rollups

Now let's get more specific on those implementation details.

As a devil's advocate

, I'll first present the traditional wisdom that you've likely seen by now. In the following sections, I'll provide my counterargument to the points I'm about to describe. It appears to be a contentious topic, so I want to provide both sides of the story:

What determines the “canonical chain” for a classic Ethereum smart contract rollup (e.g., Optimism)

— Jon Charbonneau (@jon_charb) May 24, 2023

The classic rollup fundamentalists

state the following:

Smart Contract Rollups (a.k.a. Classic Rollups)

Today's smart contract rollups (a.k.a. classic rollups) post their transaction data to Ethereum. This solidifies their data availability (DA) and ordering. You get these guarantees from consensus, which rollups outsource to their DA layer.

1) Data Availability

- DA is needed to ensure safety and recoverability of the chain state. If you don't know what the inputs are, you can't compute over

them.

- Optimistic rollups (ORUs) - You can't create a fault proof to challenge an invalid state transition if the data is missing.
- ZK-rollups (ZKRs) - A proof ensures that a state transition is valid, but we still can't recreate the rollup's state if we don't have the data behind it (i.e., funds are frozen).

2) Data Ordering

- Data ordering

is also critical. We just don't mention it often because we implicitly get it from the way that DA layers work. Running the same function over different orderings of input data would produce different outputs. If the DA layer just made available the data for a bunch of unordered trades, that wouldn't be very helpful in computing the result.

Additionally, the fundamentalists will tell you that classic rollups use Ethereum for "settlement."

3) Settlement

- These rollups deploy L1 smart contracts to Ethereum which run onchain light clients of the rollup to:

- Accept validity proofs or arbitrate fault proofs
- Provide a validating trust-minimized bridge between Ethereum and the rollup, giving them trust-minimized composability with the L1
- Determine the "canonical" rollup chain (i.e., enforce the fork-choice rule), thus "defining" the rollup

Standard light clients trust an honest majority of the chain's consensus, but rollup light clients are special because they check the validity of the rollup's state transition

. Even if a dishonest sequencer submits a block with an invalid state transition (e.g., printing a bunch of money), the rollup contract will reject it. Proofs protect the rollup:

Sovereign Rollups

Sovereign rollups also rely on a base layer (e.g., Ethereum or Celestia) for DA and ordering. However, they do not

have an L1 smart contract which determines their canonical chain. Rollup nodes decide this for themselves - they check for DA on the L1, then they locally verify the fork-choice rule. For example, proofs could be sent around the p2p layer for light clients to check (instead of posting them to a smart contract). They don't need a settlement layer - they just do their own settlement. The rollup's users decide which fork to follow.

Comparison

The fundamentalists argue that [sovereign rollups then inherit some

security from the L1](https://www.youtube.com/watch?v=GlxSP_ABE4Y&t=205s). Smart contract rollups additionally inherit censorship resistance and validity guarantees from the L1, where:

- Censorship Resistance

- Within predefined time, a user-originated transaction is guaranteed to be inserted into the canonical

chain. The argument here is that sovereign rollups can use the DA layer to force transaction inclusion, but they don't guarantee inclusion in the canonical

rollup (since the base layer doesn't determine the canonical rollup). Smart contract rollups guarantee inclusion in the canonical rollup since the base layer chooses it. (I'll explain shortly why I disagree with this).

- Validity

- Execution is compliant with the state transition function (STF) of the protocol (enforced via validity or fault proofs checked by the smart contract).

(1) Remember - this is me playing the devil's advocate and presenting the other side's argument. These rows are incorrect.

If you subscribe to the logic above though, rollups would then have to choose between:

- Sovereignty (i.e., rollup nodes determine the canonical chain for themselves), or
- Trust-minimized composability with the L1

They are supposedly mutually exclusive. If a rollup has trust-minimized composability with the underlying layer, then such a rollup is "defined through the validating bridge."

Settlement Isn't Real

Now that we've level set on the basics, I'll explain why everything I just told you is a lie

. The crux of it is that nebulous buzz word - "settlement." Settlement is a myth.

when people say "settlement layer" they mean one of three things, all of which are Wrong and Bad

— James Prestwich (@_prestwich) September 20, 2022

Ethereum is often referred to as the "settlement layer" for rollups because the L1 smart contract supposedly decides the canonical rollup chain. This is wrong.

The bridge provides a view into the rollup. The bridge does not define the rollup.

What people really mean by "settlement" here is usually justfinalizing an enshrined bridge. That's also why there's a messy debate over when a rollup finalizes. People often mistakenly think that an ORU finalizes after a week, but that is just when the bridge

finalizes its view. Different "observers" view finality at different times. The bridge is one such "observer" of the rollup (which is slow in this case), but the bridge is not the rollup

.

An observer who runs Ethereum + rollup full nodes would perceive much faster finality. They just need to check:

1. The state transition of the rollup is valid
2. The associated rollup data has been posted and ordered on the DA layer (and the DA layer has finalized this ordering)

Sovereign Rollups vs. Enshrined Bridge Rollups

Here's a more accurate table. I refer to "smart contract" rollups as "enshrined bridge" rollups to drive the point home here:

We think of these bridges as "defining" the canonical rollup today mainly due to the fact thatthere tends to be an "official" bridge which holds much of the rollup's TVL. In practice, that means the bridge holds tremendous power over the rollup.

But let's play out a little thought experiment:

- An Ethereum rollup has an "official" enshrined bridge which checks the rollup's validity using fault proofs. The website for the "official" bridge has a terrible UX so nobody uses it. It has $0 in it.
- Everyone uses other "unofficial" bridges because they have pretty websites and finalize quickly. They custody most of the rollup's funds - billions of dollars. There's also a bunch of native assets on the rollup.

Now, does the "official" bridge really define the rollup? It has $0 in it, so it's really just a "validating contract" rather than a "validating bridge" at this point. If the rollup decided to fork away from that contract completely, it could do so with little issue. More on this later.

All Rollups Are Sovereign Rollups

But wait, wasn't "settlement" the big difference between sovereign and classic rollups? Exactly:

To "not enshrine a settlement layer" is primarily a social distinction rather than a technical one, which means that there is a social contract between the rollup's community that the rollup's transaction validity rules are defined by the community rather than an immutable L1 contract.

I would argue that every rollup should be thought of as sovereign. Some specific validating bridge may tie the rollup to a given state transition function but at the end of the day the "real" state transition function is whichever one people actually use.

— smartcontracts.eth (✨ _ ✨) (@kelvinfichter) March 12, 2023

All rollups are sovereign rollups.

Even Preston Evans of Sovereign Labs notes that sovereign rollups "may be a little bit of a misnomer" and "the whole difference between sovereign and non-sovereign rollups is maybe a bit overblown". And he's building sovereign rollups! I'll attest to the fact that he's a genius who knows what he's talking about far better than I do.

"Sovereign rollups" are a pretty amorphous concept at this point, but that's a bit fuzzy for my liking. I'll specify it here. The debate over whether or not a rollup is sovereign boils down to two concrete points:

1) "Technical Sovereignty" - Source of Truth (SoT) for Rollup Users

Technical sovereignty addresses how users receive the "truth" of the rollup. This is unrelated to the ability for social consensus to fork the chain (which is how we often see "sovereignty" described).

Smart Contract Rollups: Enshrined Bridge = Source of Truth (SoT)

Users of enshrined bridge rollups typically learn the rollup's state in a trust-minimized way by looking at the onchain Ethereum smart contract. The expectation is for users to run an Ethereum full node (but not a rollup full node) which in turn runs a light client of the rollup (the bridge is the light client). The contract is the users' SoT for the canonical chain

So the smart contract is really doing two things here:

- Checks the validity of the rollup's STF

- The contract does onchain verification of a validity proof (in the ZKR context).

- Enforces the rollup's fork-choice rule

- The contract checks that a new proof builds on the previous proof (and not some other fork), that it's processed all of the relevant forced transactions which were sent on L1, etc.

Sovereign Rollups: p2p Proofs = Source of Truth (SoT)

By the definition of technical sovereignty, sovereign rollup users don't look to an enshrined bridge contract which verifies proofs to determine the "canonical" chain. Their SoT comes from somewhere else. Proofs can be distributed in other ways.

Sovereign Labs will implement the following construct

- Immediately distribute proofs p2p

- Distribute proofs via the p2p layer immediately as they are generated. This is the SoT for rollup users, which verify them directly by running rollup light nodes.

- Immediately post proofs onchain (but don't verify onchain)

- Post proofs as data blobs to the DA layer(but don't verify them onchain) immediately as they are generated. This is used for prover incentivization - the first proof which proves a batch wins the race and is paid for their work.

- Periodically post proofs onchain (and verify onchain)

- The bridge requires onchain proof verification, but users don't. Onchain proof verification is expensive, so proofs will only be batched and verified onchain periodically as needed. This allows the bridge to update its view of the rollup and bridge funds.

The result - you get a trust-minimized bridge with the DA layer, but the bridge is no longer the SoT for light clients. These rollups have "technical sovereignty".

Yes, I know that goes against what most of you thought was a "sovereign rollup."

This bridge could even be implemented without a smart contract. It could use a simple zkSNARK verifier as proposedhere. This would allow minimal DA layers such as Celestia (which will not support arbitrary smart contract logic) to have a trust-minimized two-way bridge with its rollups.

You'll notice one issue - enshrined bridge contracts also enforce the rollup's fork choice rule normally. If we distribute the proofs p2p, how will users know that these blocks being proved also follow the rollup's fork-choice rule?

It's actually quite simple - rather than having a smart contract enforce the rollup's fork-choice rule, the fork-choice rule is built into the validity proof itself

. We no longer need a smart contract to enforce it. Users can simply receive the proof directly via p2p and verify that:

- The rollup's state transition is valid
- The new block follows the fork-choice rule - it is the "canonical" tip of the chain which builds on the previous proof

To implement this new rule, you just add another statement into the proof being generated which says "I have scanned the DA layer for proofs (starting at block X and ending at block Y), and this proof builds on the most recent valid proof." This proves the fork-choice rule directly.

This rollup is still posting its validity proofs to the DA layer as data blobs, so the proofs show that it scanned all of them to determine the latest one. However, it's possible for anyone to post a proof, so you also need to bake in your rule around which is the "canonical" proof. In Sovereign Labs' case, the plan is simply for the fork-choice rule to select the first proof that proves a valid new state transition for the rollup (i.e., it's ordered first on the DA layer).

The TLDR results of this are then:

- Fast p2p proofs for users

- p2p proofs can be distributed immediately to users, giving them relatively fast finality at the speed of the DA layer block times (once the rollup transaction data has been posted and finalized onchain). This is their SoT rather than the L1 smart contract.

- Batched onchain proofs

- Onchain proof verification is expensive, so current ZKRs tend to only post proofs every few hours to save on costs. The construct described here would also recursively batch proofs and verify them onchain periodically to save on costs. However, users no longer need to wait for this onchain verification to "finalize" in their eyes (because they get the faster proofs via p2p).

Scroll has a similar plan - they will post intermediate ZK proofs onchain to Ethereum (but not verify them), so light clients can sync fairly quickly. The only difference is that they're only posting them onchain vs. Sovereign Labs is also planning to distribute them p2p prior to onchain verification.

Both constructs allow users to finalize their view of the rollup at the speed of the base layer (rather than waiting to verify the proofs via a smart contract in order to save on gas costs). Rollups can be viewed as achieving finality when:

- Their transaction data has been posted to their DA layer
- The DA layer has finalized
- Any honest node can compute the deterministic resulting state transition

The real difference in "finality" timing is just when different parties become aware of it. For example:

- A full node would know instantly that the new rollup state is final (assuming full transaction data is posted onchain, not just state diffs)
- A validating bridge wouldn't know that the new rollup state is final until it has verified a validity proof or the challenge window has passed

The construct described allows light clients to perceive that fast finality at the speed of the DA layer without running a full node. Note that in any case, true finality is bounded to the DA layer's finality time. Implementations such as a rollup consensus can give you strong pre-confirmations, but it's not truly final until the data is solidified onchain.

2) "Social Sovereignty" - Ability to Upgrade & Fork

The other definition of sovereignty we more often hear about is "social sovereignty" - the ability for a chain's community to socially

hard fork and upgrade as it pleases. Note that this requires allowing for offchain

governance:

- Social sovereignty

≠ 51% of token holders can vote for an upgrade (e.g., to upgrade the enshrined bridge contract), and if you lose the vote you're forced to go along

- Social sovereignty

= If you disagree with the changes, anyone can socially fork the chain (e.g., how Ethereum conducts any hard fork)

Upgrading a rollup's enshrined bridge requires upgrading its smart contract. This upgrade path is often undesirable:

- Today

- Contract upgrade keys are often held by a multisig of core team members. They may have the ability to make instant and arbitrary upgrades to the contract.

- Future

- Hopefully, upgrade keys will move over to rollup governance (or become immutable). In any case, this must take the form of onchain governance

(e.g., token voting, or using some other form of onchain identity to vote). Ideally, the upgrade window would be set comfortably longer than the withdrawal period. If you don't like the upgrade, you can withdraw your funds.

[This contrasts to definition of sovereign rollups which is often provided]

Sovereign rollups upgrade through forks like a layer 1 blockchain. New software versions are published, and nodes can choose to update their software to the latest version. If nodes disagree with the upgrade, they can stay on the old software. Providing a choice lets the community, those that run nodes, decide whether they agree with the new changes. They can't be forced into accepting upgrades, even if most nodes upgrade. This feature, compared to smart contract rollups, is what makes sovereign rollups 'sovereign'.

However, I contend that the above statement equally applies to enshrined bridge rollups

. Nodes for an enshrined bridge rollup:

- Can socially fork the rollup regardless of whether they upgrade the L1 contract
- Cannot socially fork the enshrined bridge and carry over its collateral (i.e., the new representation of those assets will be unbacked if they fork the rollup socially)
- Bridge ≠ rollup

Therefore, even enshrined bridge rollups can implement social hard forks

. They simply have an implicit cost imposed on forking to a varying degree based on how much collateral is in the enshrined bridge contract. Just because the bridge contract requires onchain governance (or is immutable) does not mean the rollup itself requires onchain governance (or is immutable).

As I alluded to earlier - you can imagine an example where there are no funds in the bridge. In this scenario, a social fork of the rollup would effectively impose 0 cost on the rollup. They wouldn't end up with any unbacked collateral. So these are the options to hard fork an enshrined bridge rollup:

- Upgrade the existing contract (requiring onchain governance) and socially go along with this as defining the new rollup
- Socially hard fork a new instance of the rollup (offchain governance). This could point to a new smart contract (e.g., also deployed on Ethereum L1), or they may decide to look somewhere else for proofs (e.g., distribute them p2p)

Whether it's "practical" to fork socially is a spectrum based on how important the bridge is, but it is not a fundamental limitation. If all the users decide that the forked rollup is the new "canonical" rollup and they all use that one, then that's what matters.

And users can still withdraw their collateral from the bridge to the original fork

.

They may or may not choose to bridge it back into the new fork. The new rollup would simply lose the ties to the collateral behind the bridge. You let the bridge die and move on.

In this sense, all rollups retain total "social sovereignty". They do not have social sovereignty over assets that are locked in the underlying bridge on another chain.

Offchain governance can always fork the rollup, but onchain governance is needed to fork the bridge. Bridge ≠ rollup.

The bridge determines the canonical chain from *its own perspective* in a similar way as Circle defines the canonical Ethereum fork as far as USDC is concerned.

Rollup users could pick a different fork if they were willing to sacrifice bridged assets.

— Josh (@Jskybowen) March 17, 2023

By this "social sovereignty" definition, I argue that:

Sovereign rollups are the superset term which encompasses all rollups. "Classic" rollups are basically just a type of sovereign rollup which additionally tack on a validating bridge.

Bringing things full circle, we come back to Mustafa's point I quoted above:

To "not enshrine a settlement layer" is primarily a social distinction rather than a technical one, which means that there is a social contract between the rollup's community that the rollup's transaction validity rules are defined by the community rather than an immutable L1 contract.

The difference between sovereign vs. classic rollups as he described in that great article (read it) is primarily a description of the community's current social contract

. Ethereum rollups today have a rough social contract that users look to the bridge for the canonical rollup.

The point I am highlighting here is simply that this current social contract is not fundamental

. Indeed, it is even highly likely

that social consensus would shift away from the bridge at times (e.g., if there's a major hack within a rollup's own state, and the bridge can't be upgraded to fix it, which I'll detail in an example below). The community always retains social sovereignty.

so in that regard a sovereign rollup is not really that different than a typical Ethereum rollup, except with the social distinction that it's an off-chain community that decides on upgrades via forks, rather than the bridge. https://t.co/AkLNmk3R5d

— Mustafa Al-Bassam (@musalbas) April 6, 2023

Basically this is most of us right now, except it's for Mustafa:

Everyone (def me at least lol) finally deeply understanding whatever V said like a year ago is par for the course

— Jon Charbonneau (@jon_charb) March 19, 2023

All Rollups Are L1s (& L2s)

Sunny had a unique mental model in a recent speech:

"Every chain is an "L1" for its internal state and an "L2" for its foreign state"

You could extend this logic to pretty much any external representation of an asset being an L2. So Coinbase is an L2 to Ethereum, Lightning is an L2 to Bitcoin, every chain is an L2 for Circle (they always control USDC), etc. They're just L2s with varying trust assumptions

However, I propose a new mental model. A single chain (or other entity) can simultaneously fulfill three different roles:

The L2 property is achieved by rollups with validating bridges which also ensure DA.

The sidechain property is achieved via a trusted bridge (e.g., honest majority) between the native L1 and the sidechain.

In essence, I use "L1" to refer to the portion of assets which derive their social consensus (and thus their value) from that chain. It is the ledger of record for these assets.

Enshrined Bridge Rollups Are Layer 2s

A classic rollup acts as an L2 to Ethereum when I bridge ETH → rollup because:

I do not add any security or social consensus assumptions beyond ETH's native chain (Ethereum). I place no trust in the security or

social consensus of the rollup

.

That does not

mean you aren't taking any additional risk. You're putting your money into a bridge (e.g., smart contract). Of course a faulty contract could rug you, the proof implementation could be wrong, etc. But that's no different than putting your money into a DeFi scam contract on Ethereum. You can see the code, and you make a decision. It's not Ethereum's fault or the rollup's fault if the bridge gets hacked. It's your fault for putting your money into a buggy contract. (You may also have a 1 of N assumption for optimistic bridges, but that isn't the case either for ZK bridges.)

Additionally, the rollup's governance or operators should not have the power to upgrade the contract and steal my ETH. Then I would be placing an honesty assumption on them. The contract should be immutable, or there should be an upgrade delay which is significantly longer than the forced withdrawal mechanism.

If these conditions are fulfilled - then even a malicious rollup social consensus, operator, and/or governance can't take my money. I can always get my ETH back on Ethereum.

This is not

the case for sidechains (honest-majority bridges). If I bridge my ETH via an honest majority bridge to another chain, Ethereum can't save me if that chain is malicious. They can freeze my funds and/or lie to the bridge to withdraw its collateral. I have no recourse.

For example, IBC is currently implemented by each connected chain running an onchain light client of the other chain. These light clients trust the honest majority of the other chain. If the Osmosis validator set is corrupted, they can lie and tell the Cosmos Hub whatever they want. (IBC today doesn't include state transition proofs as part of the Tendermint light client, but note that IBC can be modified to additionally implement state transition proofs as well.)

Enshrined Bridge Rollups Are Layer 1s

Enshrined bridge rollups are also "L1s" by my definition for their native

state. This state derives its meaning and value from the rollup. Take Optimism as an example:

- OP's value is derived from the social consensus around the Optimism chain.
- OP bridged to Ethereum would derive its value from the underlying OP collateral on Optimism.
- They're 1:1 redeemable, so they should trade ~1:1 in price (otherwise there's a risk-free arbitrage) other than some discount to account for the withdrawal period (time = money) and some level of smart contract bridge risk.
- If Optimism disappeared, that wrapped token would be unbacked.

So let's consider a hypothetical example:

- Optimism has fully matured - it has a validating bridge, forced transaction inclusion, bridge contract has governance upgrade delays, etc.
- Optimism has an enshrined bridge smart contract on Ethereum which holds 10% of the rollup's funds (e.g., bridged ETH)
- Ethereum and Optimism have communities with distinct social consensus and interests
- Some crazy shit happens on Optimism (e.g., someone conducts a[highly profitable trading strategy](), and now the attacker controls the majority of all rollup funds
- Optimism nodes all want to fork the rollup to undo it, but governance doesn't have the power to upgrade their existing smart contract. The attacker now holds most of the OP, and they reject any vote to fork.
- [Ethereum social consensus is unwilling to fork the base layer to modify the contract]()
- Rollup social consensus then chooses to deploy a new smart contract that defines the "canonical chain" in their eyes. It looks back on the same historical data that was posted to Ethereum for Optimism (user balances are all copied over, etc.), except it erases the highly profitable trading strategy.

On this "new Optimism" fork:

- Bridged assets are now completely unbacked

- Bridged assets which represent native Ethereum assets in the validating bridge contract (e.g., ETH) are unbacked

- Forked "OP" is the valuable one

-

All of Optimism's social consensus has decided that this is the "canonical" version

On the "Optimism Classic" fork:

- Bridged assets are fully backed

- Wrapped assets which represent native Ethereum assets in the validating bridge contract (e.g., ETH) are fully backed

- "OPC" is valueless

-

All of Optimism's social consensus has decided that this valueless (the trader owns most of it anyway), and nobody uses this chain anymore

So, even though the rollup decided to fork:

- Optimism served as an L2 for bridged assets such as ETH

- All users of the original chain just withdraw their ETH back to the L1 and recover their full collateral

- Optimism served as an L1 for native assets that derive their value from Optimism

- The value of OP is fully derived from the social consensus of Optimism. OPC is valueless as social consensus has abandoned it. Forked OP has the full social consensus, so it remains valuable.

In summary, when I use ETH or OP on Ethereum and Optimism (via an enshrined validating bridge) as a light client user, I consent to the following security assumptions and social contracts:

Assets bridged over a trust-minimized bridge retain the full security and social consensus assumptions of their underlying base layer.

Of course it would be bad if someone hacked Optimism and stole all the bridged ETH. That could be a death knell to a rollup if most of their TVL is from that contract. However, they didn't break the rollup or its safety guarantees because:

The bridge is not the rollup. A fault in the bridge (e.g., a hack) does not break the rollup.

That's why your native rollup assets can't be stolen even if the bridge is hacked to withdraw its collateral

.

Maybe the attacker even minted a bunch of wrapped OP on Ethereum, but that's not your problem. OP derives its value from the social consensus and value from Optimism. The bridged OP is simply a claim on that asset. The OP in your wallet didn't go anywhere.

However, you would

be shit out of luck if you held the wrapped bridged OP on Ethereum L1. That's now under-collateralized. You're taking smart contract risk when you hold any bridged asset.

Rollup bridge hacks then pose no risk to rollup-native assets held on the rollup They pose a risk to bridged assets that derive their value from the "real" asset locked in the bridge as collateral.

All rollup assets inherit the base layer's full technical security, but the native assets also rely on the social consensus of the associated rollup

:

I wouldn't say "as safe as Ethereum," maybe as safe as the social consensus of the L2 nodes (when/if they break away from state transition function enforced by the bridge) (or something)

— Daniel "sorry if this is a bit niche" Goldman (@DZack23)May 4, 2023

However, you should notice why it's a perfectly ok assumption. You're obviously happy with the social consensus of Optimism if you hold OP because that's precisely why you hold it

. It derives its value from that social consensus

.

Just because OP relies on the social consensus of Optimism doesn't mean that it doesn't inherit security from EthereumAll rollups derive security from their DA layer (particularly reorg resistance and DA) even when there are no bridged assets. Yes, social consensus for native assets could choose to fork, but that's always the case:

Agreed. Any asset is really only as safe as social consensus (even if you create a erc20 or a 721 the social consensus drives value) but the technical consensus is fundamentally hard to break on L1 as the ordering of transactions is clearly established with L1 security and if you…

— Sreeram Kannan (@sreeramkannan) May 4, 2023

If you believe that social consensus makes all technical security guarantees meaningless, then Bitcoin and Ethereum are completely insecure. PoS provides no security to Ethereum and PoW provides no security to Bitcoin by that logic, since social consensus is the ultimate authority. In reality, you're always just subjecting yourself to the social consensus around any asset you hold. What you don't

want to do is to give a separate

social consensus set power over your assets (e.g., Optimism shouldn't be able to steal my ETH).

Burning Bridges

As an interesting thought experiment, let's consider a rollup where the bridge burns your collateral

. What if we made a simple OP Stack fork with an enshrined bridge to Ethereum, but burned the ETH you put into it:

- Today

- Lock your ETH on Ethereum, get a wrapped representation on Optimism

- Ultra-based rollup

- Burn your ETH on Ethereum, and get ubETH on this OP Stack rollup. Let's also make it a based rollup for the memes, because memes.

What you've effectively done here with this bridged ubETH is:

- Retain Ethereum security across this bridge for this rollup
- Transferred the value of this asset from Ethereum social consensus → Optimism social consensus

ubETH no longer derives its value from being able to redeem it for ETH on Ethereum. The collateral is gone. But ubETH is the native token of this based rollup now - you need it to pay for gas, etc. It's also used for governance. Its value is purely from the social consensus around it there.

You could also bridge the ubETH back to Ethereum, but now you get a wrapped version down there. You're not getting back your ETH collateral. Rather, your bridged token is a representation of the ubETH collateral on the rollup (which is the "real" asset now).

Now I consent to the following security assumptions and social contracts:

Note it's also possible to make this feature optional. You could allow for users to lock collateral in the bridge and get a bridged representation (say opETH), but that asset isn't fungible with ubETH. Only ubETH is the native token for fees on the rollup and has other governance rights. It'd be a fascinating social experiment. I'd probably burn some ETH for the culture.

Sidechains - Honest Majority Bridges

When I use ETH or SOL on Ethereum and Solana (via an honest majority bridge) with a light client, I consent to the following security assumptions and social contracts:

When you use an honest-majority bridge to move your ETH to a wrapped version on Solana, you are granting Solana full control over your assets as well

. Solana isn't serving as an L2 extension of the asset you had on Ethereum. It's a sidechain with full control over your ETH. It has the ability to confiscate your funds if it chooses to. It could also lie to the Ethereum bridge and withdraw the initial ETH collateral.

It goes the other way as well. A 51% attack on Ethereum could revert blocks that sent ETH to Solana, making the wrapped representation unbacked. Even a user running an Ethereum full node would lose funds here. This is the classic example Vitalik gave in his infamous [argument for why the future will be multi-chain

, but it will not be cross-chain.

](https://old.reddit.com/r/ethereum/comments/rwojtk/ama_we_are_the_efs_research_team_pt_7_07_january/hrngyk8/)

Note that if I run a full node

, I make no safety assumptions when operating on a native chain (e.g., using ETH on Ethereum). Even a dishonest majority can't fool me. However, if I'm only running a light client, then I am trusting the honest majority. They could maliciously convince me to send me my funds out for example.

Ethereum is an L2

Ethereum can even be an "L2". For example, let's say I bridge my OP token from Optimism to Ethereum. In both cases:

- My OP maintains the same social consensus assumptions of Optimism.
- My OP maintains the same security assumptions of Optimism.

Note that Optimism's security = Ethereum's security in this case, so it's indirect. In any case, the point remains - I'm adding no new assumptions external to Ethereum.

This also captures the notion that when you're using USDC on Ethereum, you're really trusting Circle

. Ethereum can't save you if a centralized issuer has the right to confiscate your funds. Conversely if Ethereum tried to take your USDC, Circle could always give your dollars back. Circle is the real ledger of record here.

So yea, Ethereum is an L2 to Circle. USDC holds precisely as much sway over Ethereum as an enshrined bridge holds over a rollup. "Native USDC" can be thought of as "bridged USD." The dollars are in Circle accounts instead of Ethereum bridge contracts.

In fact, you could say that people like USDC for much the same reason they like rollups

:

- I want to use my ETH on some other chain without trusting anyone but Ethereum
- I want to use my USD on some other chain without trusting anyone but Circle

TLDR on L1s, L2s, & Sidechains

I don't know that this L1 & L2 definition will stick, but I think it's a helpful framework to drive the point home. Different assets derive their value from different sources, and they carry different assumptions as a result.

When I use a chain ETH on chains as:

- L1 (e.g., ETH on Ethereum)

- I rely only on Ethereum security and social consensus.

- L2 (e.g., ETH on Optimism)

- I rely only on Ethereum security and social consensus. I also take implementation risk. (I.e., the smart contract could have a bug, same as any other smart contract on Ethereum. This includes the associated proof system, and potentially a 1 of N assumption for fraud proofs.)

- Sidechain (e.g., ETH on Solana)

- I rely on Ethereum and

Solana for both security and social consensus. I also take implementation risk, same as any other smart contract on Ethereum.

Whenever you're using a chain as an L2, that's when you're subject to bridge risk

. If you're using something that's bridged, you fundamentally carry that risk if the bridged representation has value primarily from the underlying collateral elsewhere. Assets are always safest on their native chains.

However, the native tokens on that L2 do not carry this bridge risk. It's acting as its own L1.

You could deploy an asset natively on multiple chains using a burn-and-mint bridge. However, note that this asset would then carry the security risk of the lowest common denominator of bridges deployed across all chains. For example, if someone tricks one bridge and mints a bunch of assets on that chain, then the whole token supply is affected (since all assets are fungible and redeemable 1:1).

The difference between an L2 vs. sidechain is basically whether the other chain can screw you. Obviously your money can always get stolen if you put it into a faulty smart contract (like a rollup bridge). But that risk is known and internal to the chain you're coming from. If you perfectly inspected that code, you could find the bug. All the data is available to you. Neither Ethereum's nor Optimism's security rugged you if the contract gets hacked.

The difference is that in a sidechain (i.e., honest majority bridge), the risk is external to your sandbox of the data you can read trustlessly. The consensus of the other chain could just lie and rug your bridge even if it's perfectly implemented. In a true L2 rollup bridge, the only risk is implementation, which can be known.

Overall, rollups bring two very different concepts:

- Inheriting security

- Rollups always inherit the security of their DA layer.

- Scaling

- If you have a validating bridge to your DA layer (or another chain on it), then you're scaling - that's what L2s do. These chains can interact with bi-directional messaging, and you retain the security assumptions of your home chain. For example, I can use my ETH on Optimism to do something that requires computation I can't do on Ethereum, and I retain Ethereum security while doing it.

Also note that the L1 & L2 mental model I'm describing here is quite asset-specific. It's not very helpful in considering some completely generic message passing between a rollup and its base layer. Arbitrary messages don't really derive and hold value in the same way from social consensus or something locked in a bridge.

Getting Philosophical on Rollups

Coming back to what I described earlier, a rollup is just an output we get by running a function over some input data. It derives security based on the layer where that data is posted.

We need to stop thinking about rollups as some canonical thing defined by math and science and cryptography that Ethereum tells us is the objective truth. It doesn't need to be this specific thing where I make a new chain, bridge over a bunch of assets, launch some DeFi stuff, etc. then Ethereum tells me it's ok.

I can trustlessly run any function over some data on Ethereum and make a rollup. That's it. I don't need anyone's permission. We come to some rough social consensus on what outputs are interesting bits of data that we care about. Rollups are literally just completely made up things.

As a random example, you could just launch a new rollup deployed on top of Optimism Mainnet to do contract secured revenue (CSR) for it:

The OP Stack is about to make Ethereum contract development crazy again. Hard to explain until you've seen it. It's just stupid how powerful this thing is. The idea of getting CSR on Optimism Mainnet was what really got me. (quik thred) https://t.co/OeyUHUgdF3

— smartcontracts.eth (✨ _ ✨) (@kelvinfichter) February 13, 2023

The rollup can just trustlessly read the data that's already being produced and posted by Optimism:

I honestly don't have any idea if this is an interesting specific use case. But it's a great point of what these systems fundamentally are. There are essentially infinite undiscovered rollups hidden in Ethereum's data. You can make a rollup to trustlessly read and compute over that data however you want, then you could provably communicate it back. That's cool.

Maybe people will find your thing interesting, maybe they won't. But that's the point. It's all social consensus.

The Cost of Enshrined Bridges

It should now be clear that all rollups are "socially sovereign" rollups. However, they do not have sovereignty over assets which are locked in their bridge. Indeed that is exactly the point of these bridges

. You want to use an asset (e.g., ETH) on another domain (e.g., Optimism) without trusting the other domain's security or social consensus.

However, this comes at a cost - enshrining a bridge may impose a high cost on social hard forks

. Traditional honest majority bridges rely on the other chain's consensus. While this is optimal from a security perspective (i.e., I can use my asset on another chain without trusting it), it tethers chains together and limits their flexibility to upgrade. The connected chains would have to hard fork together.

Honest majority bridges typically allow each chains' consensus set to hard fork costlessly. Validators are able to enforce social consensus, making hard forks more like real "upgrades." This is often very desirable, as chains will reasonably want to make upgrades over time.

However, a trust-minimized bridge by definition does not just take the consensus set's word:

Yep! Sovereign Rollups with trust-minimized are much more like traditional dApps in this sense.

They either need governance to manage upgrades in a way that’s legible on-chain or they have to deploy a new instance for each upgrade and have users migrate over gradually.

— Preston Evans (@prestonevans__) September 3, 2022

So there is a tradeoff when you get trust composability, but it's not a mutually exclusive and absolute "your chain is no longer sovereign."

Note that IBC-connected chains would have a similar constraint today if they wanted to hard fork in a manner which changes the light client's rules. For example:

- A hard fork which changes the state machine is fine (e.g., print a bunch of tokens)
- A hard fork which affects consensus requires coordination (e.g., includes irregular validator set changes, header format changes, etc.)

If one chain has a light client-breaking hard fork, then all of its implemented IBC-connected chains must hard fork as well. In Cosmos today this is reasonable to coordinate, but obviously Ethereum isn't going to hard fork for every rollup that wants to upgrade.

This complexity of fork coordination could be mitigated by implementing some form of onchain upgrade mechanism (similar to how rollup onchcain governance could upgrade their smart contract). It just has the same constraint I mentioned earlier - you can't use offchain governance to do this. It needs to be legible onchain.

It's also important to note that consensus faults from connected chains are not "contagious."

The classic example that's cited is the following:

- Osmosis had LP pairs such as OSMO/UST
- As Terra collapsed, its economic security was approaching 0. It would become rational for a 51% attacker to take control of the chain, message Osmosis via IBC to mint a huge amount of wrapped UST, then drain all LP pools paired against UST

Obviously this would be bad. However, similar to a rollup bridge hack not "breaking" the rollup, this does not "break" Osmosis. Anyone who exposed themselves to UST price risk would be wiped out, but that's the risk you take when you willingly expose yourself to the price risk of another asset. UST did indeed go to 0 anyway. This attack wouldn't have "broken" other chains connected to Terra.

Where trust-minimized bridging is particularly valuable though is when a user wants to use a given desirable asset (e.g., ETH) on other domains in a more productive manner while retaining their initial security assumptions. They don't want to allow the chain they're using to confiscate their funds.

This is how rollups "scale" the native assets of Ethereum. All Ethereum rollups inherit its security, but they're not "scaling Ethereum" if they don't have a trust-minimized bridge between them to port across assets and other messages.

The Root of the Misunderstanding

Sreeram recently had an amazing thread (as usual) on Bitcoin sovereign rollups and security which I'll summarize as a starting point. These rollups would post their data to Bitcoin, but Bitcoin doesn't have smart contracts to verify proofs (nodes verify offchain).

There are four properties which determine the security (safety and liveness) of a chain:

1. Re-org Resistance

- Finalized blocks will not be reverted.

1. Censorship Resistance

- Anyone willing to pay the necessary fees can get their transactions included.

1. Data Availability

- All transaction data for the chain has been made available. This allows anyone to recreate the rollup state (and arbitrate fault proofs if needed).

1. Validity

- Anyone can determine that the transactions correctly result in the current state.

Writing data to a base layer (such as Bitcoin) gives you the first three. Number four is where the knives come out and people start fighting. As I discussed earlier, the classic rollup fundamentalists will tell you that you aren't getting the validity from the base layer if the base layer doesn't verify for itself (as in an Ethereum smart contract).

But we don't need Ethereum to tell us that - that's the whole point of proofs! Just send me a proof.

The fundamentalists will tell you to run an Ethereum full node to check that proof. However, that's just because the rollup light client is implicitly embedded within the Ethereum full node. But you don't need to run an Ethereum full node and verify all the other stuff that Ethereum does - you just care about verifying the rollup's proof here

. You can simply have a rollup light client that checks the proof in some other manner (e.g., p2p, or post the proofs to may chains) without the overhead of running an Ethereum full node.

Obviously you get other nice things by running an Ethereum full node, but that's besides the point. To know the validity, I can check in any number of ways. So "getting validity from Ethereum" makes no sense here. You're always running some type of node to check the state of the world:

- If you're a full node, you're assured of validity by checking yourself.
- If you're a light node, you're assured of validity from the proof. You just pick one method or another for how you check that proof (whether that's p2p, via a smart contract bridge, etc.).

In any case - the user is ensured of validity, and they now have all four security properties

.

So yes, you can get the full security (safety and liveness) of the base layer even if the base layer does not verify the proofs for your rollup.

> Good we finally find an actual disagreement rather than talking past each other Here is my position - even sovereign without a validity bridge) inherit full ethereum security for native assets. https://t.co/xn6WTntj1w
>
> — Sreeram Kannan (@sreeramkannan) May 10, 2023

What you don't get is a validating bridge that allows you to move assets back and forth between the rollup in a trust-minimized way, but this is completely orthogonal to whether your rollup inherits the base layer security. Again - "scaling the base layer" and "inheriting security" are completely different things. Both are valuable, but they are very different.

And yet the fundamentalists argue that sovereign rollups rely on an economic majority security

The argument posits that the offchain actors in a sovereign rollup determine the "canonical chain" (and they are presumably the economically important actors) vs. classic rollups allow Ethereum to decide the canonical chain. Indeed they go so far as to say "in a validating bridge, a consensus fork is not possible."

As described earlier, this is false. Social consensus always determines the canonical chain, regardless of whether you have an

enshrined bridge. It's not possible to fork the bridge socially, but the bridge is not the rollup. The rollup can do as it pleases. A blockchain is always reliant on social consensus. That's fundamentally what it is.

So here's the mistake that the fundamentalists make in my view:

Many incorrectly assume that the social consensus of an enshrined bridge rollup will abide by the base layer's (e.g., Ethereum's) social consensus regarding which is the "canonical chain." However, this is indeed an ASSUMPTION.

All rollups are sovereign rollups, and even users of an enshrined bridge rollup may choose to follow another chain. Whether or not to abide by the bridge's view of the rollup is itself a matter of social consensus.

Thankfully, Kelvin has been slowly converting the "rollup is defined by the bridge" team

WTF Is "Settlement" Then?

As James so eloquently put it:

stop saying "settlement layer" it makes you sound dumb

— James Prestwich (@_prestwich) September 20, 2022

"Settlement" generally gets thrown around for one of three things as he describes:

1) Ledger of Record (for a specific asset)

- This could be Ethereum for ETH, SOL for Solana, etc. It's the definitive ledger of record for that asset. "Settlement layer" is fuzzy (e.g., I can "settle" an ETH trade on Coinbase). This is how I use the term "L1."

2) Finalizing an Enshrined Bridge (for a specific rollup)

- The bridge doesn't "settle" the rollup. It just finalizes its bridge. Bridge ≠ rollup.

3) Petty Boosterism (for my favorite coin)

- Settlement layer sounds cool I guess.

Basically, you should stop saying "settlement layer" because you sound like a total ethboi shill

Instead, say "ledger of record" or "enshrined bridge" or "I have no personality except my favorite block chain"

— James Prestwich (@_prestwich) September 20, 2022

I'll add a fourth one as well:

4) Source of Truth (for rollup nodes)

- The bridge can be users' SoT, implicitly verifying the rollup's state transition within a base layer (e.g., Ethereum) full node. It provides a single point of reference for nodes to look to. These proofs could also be distributed via the p2p layer. However, posting them to a chain (or many chains) could be useful in circumstances where p2p communication is uniquely challenging. For example, it can be difficult to conduct IVG fault proofs via p2p, so it's easier to have a smart contract arbitrate them. That technical distinction is noteworthy here.

Overall, "settlement layer" has been a fuzzy mix of the above, referring to a bridging/liquidity hub which verifies the state transitions of connected chains. For Ethereum rollups today, it's incredibly important - the bridge is so important to their existence that it would in practice likely dictate the "canonical chain" in the event of any social consensus split today.

However, this is a spectrum of importance which will continue to evolve with different "settlement layers." Enshrined bridges may become even more valuable as many chains hopefully begin to interoperate more. Conversely, native burn-and-mint bridging for the likes of USDC with CCTP could be a force in the other direction, reducing the need for bridged assets (and their associated risks).

I think settlement layer is a bad name but we don't yet have a replacement name for "domain outside of your chain where you make (and attempt to back up) claims about your chain"

— smartcontracts.eth (✨ _ ✨) (@kelvinfichter) November 3, 2022

Lmk if you have any better names out there.

Settling to 2 Chainz - Validiums Aren't Real

Here's a fun idea - what if we have a rollup that "settles" to two chains?

Let's say we deploy Optimism as a "validium." Normally this would use one settlement layer and one DA layer:

But now let's deploy it as a validium"settling" to multiple chains:

- Optimism has two validating smart contract bridges on two chains - one on Settle_1

and another on Settle_2

. Both bridges receive validity proofs proving the validium's correct state transitions.

- Optimism uses DA_1

as its DA layer. DA_1

sends DA attestations to smart contracts on Settle_1

and Settle_2

.

Some argue that this would degrade your security to the lowest common denominator of the chains you're settling toHowever, the only funds at risk are the funds bridged from the chain you're settling to. Remember, bridge ≠ rollup.

Let's think about what could break here:

- DA layer

- The validator set of DA_1

could lie to either bridge, sending it an attestation that data is available when it is not. It could potentially also censor or reorg depending on the chain.

- Settlement layers

- The contracts on these chains are just finalizing the bridges

. A fault in Settle_1

could make contracts there think your validium is sending messages that it isn't sending, but that fault won't impact the validity of messages seen on Settle_2

.

A "settlement layer" isn't finalizing your rollup. It's finalizing its bridge which has a view into your rollup. If Settle_1

is malicious, it can't steal your OP in your wallet on Optimism for example. Anyone using this rollup can compute over the data on the DA layer and see the deterministic result. [So the lowest common denominator security doesn't apply to "settlement layers," but it does

apply to your DA layer](https://twitter.com/kelvinfichter/status/1588019209321283585?s=20). The DA layer is where you derive your security from. If you were to rely on separate DA layers equally, then a fault on either side could cause a fault everywhere.

So validiums are just rollups

. They're rollups to whatever DA layer they use.

The rollup isn't inheriting security of their settlement layer. Just think about it. Let's say you have a classic rollup that uses Ethereum for DA and has a validating bridge there. I can go deploy a validating bridge for your rollup on some shit chain by myself. It looks to Ethereum for DA, and it also gets sent a perfectly legit validity proof. I didn't make your rollup an insecure validium. I just gave your Ethereum rollup another bridge.

Rollups inherit security from their DA layer. You get your security by posting your data somewhere, then everyone can go agree on what that data means. That's about it.

Bridging & Liquidity Hub

As it pertains to rollups, arguably the primary benefit from posting proofs to Ethereum is because it serves as a hub for liquidity and bridging.

Rollups already get security from the DA layer, but trust-minimized bridging is what allows a rollup to fully tap into the economic activity of other chains. That's the real reason that rollups enshrine trust-minimized bridges to Ethereum. They want to tap into the full liquidity and network effects that Ethereum facilitates.

Remember that trust-minimized bridging between rollups is only possible for rollups which use the same DA layer. Any rollup can enshrine a trust-minimized bridge either to their DA layer, or directly to other rollups which share their DA layer. So we can broadly imagine three bridging topologies:

Pairwise Bridging

Rollups can enshrine bridges with each other, effectively forming a "cluster" of rollups that are all tied together. If there are many chains, this eventually results in high messaging complexity. Pairwise bridging between N rollups = N^2 bridges.

The fungibility of assets is another problem. Considering two examples:

- Transfer an asset from chain A → B → C
- Transfer an asset from chain A → D → C

The two assets on Chain C are not

fungible. You always need to trace back the routes for wrapped asset bridging.

Hub-and-Spoke Bridging

A single chain can serve as a bridging hub to coordinate between many rollups. Rollups would enshrine a bridge into this "settlement layer." This may be the base layer itself (e.g., Ethereum or Celestia), or it may be a "settlement rollup" on top of the DA layer (e.g., Eclipse or Astria EVM). These hubs could restrict their VMs, or be general purpose.

In any case, rollups which enshrine bridges into this shared "settlement layer" get direct trust-minimized bridging with it. Additionally, they would have indirect trust-minimized bridging to other rollups that also enshrine bridges to this same settlement layer.

You reduce the N^2 bridging complexity back down to N, and you solve the issue of asset fungibility that arises from bridging between many different chains. It can also serve as a focal point of shared liquidity.

Aggregated Bridging

Lastly, we could get fancy with ZK stuff and do aggregated bridging with many rollupsI wrote about this idea from Sovereign Labs almost a year ago, and they released another great post covering it recentlyhere.

This is another option to get all-to-all bridging and reduce the bridging complexity back from O(N^2) to O(N), without

using one chain as the coordination hub. The basic idea is surprisingly simple - take a proof for all N chains, recursively aggregate all of them offchain, then verify the aggregate proof on each rollup. Now each rollup has a single proof for all rollups in this network!

Effectively you're making a very "flat" bridging topology vs. a more vertical hub-and-spoke model where you're bridging up and down through a single point. You can now eliminate the two-hop bridging process as well.

Stickiness & Network Effects in the Rollup Stack

Maybe you're thinking this is all just a bunch of semantics around social consensus jargon, and none of this matters. Well, now I'll explain why that is not the case. Aside from the obvious security implications, there's another huge point:

The arguments I have laid out have fundamental implications for the balance of power in the rollup stack.

Let's dive into that. Where do the network effects for DA layers, settlement layers, and rollups really come from?

Rollups

Following my logic argued above, rollups may have more power and sovereignty over the valuable new state which they create natively. Even when they have an enshrined bridge to Ethereum, I don't inherently consider them "baby chains" that are locked into Ethereum.

If all the valuable new state is being created on rollups with their own native assets that derive valuable from their own environment, that has meaningful implications for the power dynamics in the stack.

DA Layers

As a recap of what I provided above to synthesize things:

- Rollups can inherit full security from their DA layer, regardless of whether they have an enshrined bridge connected to it
- It's only possible to get trust-minimized bridging between rollups that share a DA layer

Overall, there are positive network effects here. It's beneficial to have many rollups sharing the same DA layer - they can pool security and increase the size of the ecosystem that gets trust-minimized bridging with each other. Having each chain be its own DA layer that connected chains can run DAS on top of would create incredibly high overhead and fragmented security.

Conversely, overloading a single layer that may not be able to handle the whole world's DA throughput could increase costs for everyone. Scalable DA will matter.

Settlement Layers

So, are all the rollups going to ditch Ethereum then? They have sovereignty over their valuable state, and maybe they want to do their own thing. There's always been speculation around this idea. Or maybe they just need a DA layer, but they don't need to "settle" to Ethereum anymore?

Well, not so fast - settlement layers still have some serious network effects. We see this incredibly clearly today since we're colored by Ethereum. As I mentioned above, this is arguably the biggest benefit to today's rollups. They draw a huge amount of their TVL and users by tapping into Ethereum's massive liquidity and network effects.

ETH is Ethereum's #1 export. You're not about to dip on Ethereum if ETH actually becomes "money" (whatever this means), and your whole rollup economy is based on it. Trust-minimized bridged ETH is the Trojan Horse into the rollups:

This dynamic will look quite different for different settlement layers. Others won't have the depth of liquidity and native activity that Ethereum will have. The bridging topologies may also evolve meaningfully over time (e.g., as in the aggregated bridging example).

The following are some of the areas which could make a settlement layer attractive to a rollup:

- Asset issuance

- For example, this could be desirable native assets (e.g., ETH) or from a stablecoin issuer (e.g., if USDC or DAI issues only on the settlement layer, then it's bridged up) that people want to use on rollups in a trust-minimized manner

- Bridging hub

- Even if the assets used in a rollup ecosystem aren't native to the settlement layer, a bridging hub is still valuable for routing between them (e.g., as described above with fungibility issues around bridging paths)

- Users

- There's not a lot of these in crypto. This is kinda related to the above points.

- DeFi & Shared Liquidity

- Stuff like DeFi pooling could be useful where you keep all the liquidity on one chain then batch transactions from a bunch of chains to interact with the unified liquidity.

- Onchain verification vs. p2p

- As described previously, p2p fault proofs can be particularly challenging. It's potentially easier to run them onchain.

This is also one of the reasons why I find theproposal I mentioned earlier to enable ZK verification on Celestiaquite interesting. It allows rollups to create a trust-minimized two-way bridge with Celestia, bridging up their native token into the Celestia rollup ecosystem. To the extent this becomes a widely used asset in these rollups, that's sticky in the way that I mentioned for ETH in Ethereum rollups.

I'd also note that just being "branded" as part of an ecosystem could still help with network effects. Social alignment with the values of an ecosystem such as Ethereum or Celestia does have real implications.

Conclusion

It's social consensus all the way down, rollups are real, and nothing is real.