Abstract

We propose a higher-level and better-formalized guiding principle for Stage 1. We first discuss some contentious examples of projects that might fit or not fit the Stage 1 assessment depending on different interpretations of the requirements. We then propose the principle, its rationale and discuss how the examples presented fit within its definition. We discuss the limitations of the Stages Framework when applied to appchains, using Kinto as a use-case example. Finally, we propose a visual segmentation of Stages when applied to universal chains and appchains.

Background

The Stage 1 milestone in the Stages Framework is intended to represent a decentralization level that fits between projects fully controlled by centralized entities (Stage 0) and projects fully managed by smart contracts (Stage 2). Finding an intermediate between the two is not an easy task, and in this post we aim to formalize certain notions around security properties to make them simpler to reason about, but also more strict and less open to interpretations.

In Stage 1 there are two critical aspects that need to be assessed:

1. Upgrade powers

. In this article from Dec 7, 2023 we formalize the Security Council definition and prove the proper thresholds (i.e. ≥75%) that give at least "half" of the power to the proof system, as intended for Stage 1.

1. Exits

. One of the requirements for Stage 1 can be summarized as follows: "Users are able to exit without the help of the permissioned operators"

. As of today, what defines a permissioned operator is not clearly stated. For example, is the Security Council considered one of them? What about a DAO? If these two entities are supposed to protect from more centralized operators (e.g. a censoring sequencer), what threshold should be used? Or should users be able to eventually exit even without Security Council intervention? These are some of the questions that we aim to answer to in this post.

Three contentious examples

1. OP Mainnet: the fallback mechanism

OP Mainnet, like all standard OP stack chains with active proofs, implements a mechanism to promptly protect from exploits in the fraud proof system that can be visualized as follows:

[

image

1920×1083 127 KB

](https://europe1.discourse-cdn.com/flex017/uploads/l2beat/original/1X/e9d5b7ba9caa24efe5b75ceb8f5ba89b6e1815d6.jpeg)

The project defines two "game" types: the FaultDisputeGame

, which represents permissionless fraud proofs, and the PermissionedDisputeGame

, which inherits from FaultDisputeGame

but adds access control to both proposing and challenging, meaning that only the OP Foundation can propose and challenge. In case of bugs in the fraud proof system, as it already happened on Aug 16th, 2024, the "OP Foundation Operations Safe" can effectively deactivate the proof system and fallback to trusted state root proposals. At the time of deployment on June 10th, 2024, we (L2BEAT) assessed that this mechanism falls within the Stage 1 definition as any malicious state root could be reverted by the 10/13 Security Council within the challenge period. Such mechanism, though, allows the following attack to be performed by the OP Foundation:

1. Activate the PermissionedDisputeGame

;

1. Bribe a quorum-blocking minority (4 members) of the Security Council not to act;

2. Propose a malicious state root;
3. Wait 7 days;
4. Steal all funds in the bridge.

In this setting, the system is safe only if the majority of the Security Council is assumed to be honest, i.e. cannot be bribed. If the Security Council threshold increases, e.g. to a 13/13, this assumption becomes stronger (i.e. worse) as it is sufficient to only bribe one member to steal all funds in the bridge.

2. ZKSync Era: the TransactionFilterer mechanism

ZKsync Era, and more broadly the Elastic chain, has introduced a governance system that includes a proper Security Council for upgrades. Each ZK stack chain connected to the contracts under shared governance also maintains an additional independent administration role, known as the ChainAdmin

. The ChainAdmin

, among other things, can add a “TransactionFilterer

” that can censor certain transactions based on sender, target, value, or calldata. While the process is lengthy and not well defined, the Security Council or DAO can in principle revoke such filterer via a contract upgrade. As before, a failure in the Security Council or DAO minority can prevent the reversion.

[

image

1330×428 94.9 KB

](https://europe1.discourse-cdn.com/flex017/uploads/l2beat/original/1X/e7986c71d58a7177f234b099322734f06916376e.jpeg)

Allowing this mechanism in Stage 1 implies allowing censorship in case of minority failures.

3. Starknet: the forced txs mechanism

Starknet’s current implementation of the L1→L2 messaging mechanism cannot be considered a forced transaction mechanism as the operator is free to cherry-pick any transaction and censor others at its own will, meaning that the system doesn’t satisfy the exit requirement for Stage 1. Starknet plans to use the Security Council as a mechanism to provide censorship resistance by allowing it to force state updates and include the censored transactions. If the threshold used for such action is 75%, then a quorum-blocking minority can be bribed to enable censorship, following the same scenario as ZKsync Era’s.

[

image

1812×230 66.9 KB

](https://europe1.discourse-cdn.com/flex017/uploads/l2beat/original/1X/625024954582ce385208680e5af8d1b50165a644.png)

A better formalization of the Stage 1 milestone

We propose the following high-level property to be satisfied to be considered a Stage 1 rollup:

The only way (other than bugs) for a rollup to indefinitely block an L2→L1 message (e.g. a withdrawal) or push an invalid L2→L1 message (e.g. an invalid withdrawal) is by compromising ≥75% of the Security Council.

Assumption

: if the proposer set is open to anyone with enough resources, we assume at least one live proposer at any time (i.e. 1-of-N assumption with unbounded N). We don’t assume it to be non-censoring.

Rationale

In practice, “enough resources

” means enough funds to bond in a fraud proof system or spin up the necessary hardware to run nodes and provers. These resource requirements might be significantly high and might significantly impact the decentralization of the proposer set.

We want to allow emergency pauses

to be initiated by non-SC multisigs, which, without further conditions, is in contrast with the property, as it is enough to bribe a Security Council minority to prevent the unpause. The solution is to require an expiration

for the pause and a cooldown period

before the non-SC multisig is allowed to pause again.

Since no proposer is assumed to be non-censoring, the property holds only if the proposers are forced to include txs via L1

.

If an external consensus

is implemented for withdrawals, without further conditions, the property does not hold as the majority or supermajority of proposers needs to be live to process withdrawals. An automatic fallback mechanism

that doesn’t require the Security Council intervention is needed if the majority fails to be live.

Consequence

Let’s apply the principle to the three cases presented above:

- OP Mainnet

: the system cannot be considered Stage 1 as it is possible to only bribe a quorum-blocking minority of the Security Council to confirm an invalid state transition. In a similar way, the system doesn’t implement a recovery mechanism from a malicious pause that can’t be disabled by bribing a quorum-blocking minority.

- ZKsync Era

: the system cannot be considered Stage 1 because the Security Council supermajority is needed to disable a malicious TransactionFilterer, meaning that a quorum-blocking minority fault can again prevent a transaction from being included. For the same reason, since the proposer can only be rotated by the SC supermajority, there can be minority faults.

- Starknet

: if the SC supermajority is used to provide censorship resistance, then a fault in the quorum-blocking minority can prevent it. A natural solution is to allow state updates to the <25% minority, so that the only way to fully censor a transaction is by having >75% of the Security Council agree to it.

There is one nice-to-have which is not covered by the one presented above. As discussed for Starknet, the minority (<25%) of the Security Council can be employed to provide censorship resistance via forcing state updates or rotating sequencers, instead of employing a proper forced tx mechanism. The consequence is that the following property is then not true:

Users are able to exit even if the Security Council completely disappears.

The effect of this property is that then the only difference between Stage 1 and Stage 2 is strictly upgradability, as Stage 2 already naturally requires all the security features to be implemented. In other words, in Stage 1 the Security Council would be employed only to resolve technical bugs and not to recover from other malicious actors or provide essential mechanisms like censorship resistance. After broad discussions, we decided to exclude this property in this first version, but we remind all projects that this step is critical to eventually reach Stage 2.

At the implementation level, the principle implies the following requirements:

- A proper proof system needs to be implemented. Projects without a proof system cannot be safe and live under minority failures.
- A forced txs mechanism needs to be implemented, OR the Security Council minority (<25%) needs to be able to provide censorship resistance guarantees.
- Challenger whitelists in fraud proof systems are not allowed. Projects with a challenger whitelist can be compromised by the Security Council quorum-blocking minority and challengers in the whitelist.
- Unlimited pauses by non-SC multisigs are not allowed. Projects with unlimited pauses by non-SC multisigs can prevent L1→L2 messages by compromising the Security Council quorum-blocking minority. Short-lived pauses with a cooldown period are allowed.

As with the L2 projects recategorization announced in Dec 2024, we'll give affected projects a 6 month period before the changes will be applied.

Moreover, after collecting feedback from multiple rollup projects (including but not limited to Optimism and Arbitrum), we will strictly enforce a ≥7 days challenge period for all Optimistic Rollups to be considered Stage 1. For the assessment, we'll use the broader definition of challenge period that includes artificial "grace periods" after proposal confirmations.

Framework limitations when applied to appchains

The complexities of assessing Stages become evident with very customized chains. Intuitively, when discussing standard, universal rollups, one of the most important aspects is being able to permissionlessly withdraw tokens back to L1. Some tokens, though, might have different security than others: for example, they might be censorable by their issuer, as is the case with USDC. Our scope is not to assess whether permissionless withdrawals are possible for any token, but to check which trust assumptions the rollup operators add beyond the ones already present on the application and token level, which we assume the user to accept when interacting with them. In principle, if a project is Stage 2, and a token is fully permissionless (e.g. ETH), then it should also be permissionless to withdraw, but the same might not apply for all tokens.

If a project restricts deposits to censorable tokens, or restricts applications to permissioned ones, then it might be possible that no user would be able to exit in certain circumstances. For simplicity we call these projects "app-specific chains", or in short "appchains" to distinguish them from universal rollups where any arbitrary message can be exchanged and the state transition function (STF) is not particularly constrained. An example of such a project is dYdX v3: meant as a DEX-only chain, it does not allow arbitrary messages to be sent and does not allow arbitrary smart contracts to be deployed, but is considered a Stage 1 rollup because operators cannot block withdrawals, even when fully shutdown. However, USDC is the only supported token and the whole chain depends on it, meaning that in practice Circle, i.e. a centralized intermediary, would be able to prevent withdrawals for any user.

A fourth, even more contentious example: Kinto

Kinto is an Orbit stack rollup that enforces KYC for all users. This is done by preventing (in the STF!) EOAs from interacting with smart contracts unless explicitly whitelisted.

[

image

2048×1528 178 KB

](https://europe1.discourse-cdn.com/flex017/uploads/l2beat/original/1X/9eea1c2a69b3b8be9fe0cadf4bb5df0f371ef7b1.jpeg)

One of the most important contracts on this whitelist is the ERC-4337 EntryPoint used with smart wallets, and the WalletFactory which only allows to create KYC-gated smart wallets.

While the KYC providers can prevent user withdrawals, the rollup operators cannot: forced transactions are enabled, a whitelist of external actors is used for fraud proof challenges (similarly to pre-BoLD Arbitrum) and the upgrader is a properly set up Security Council.

Moreover, an interesting consideration is Kinto's choice of primarily using externally-bridged tokens (via Socket protocol) with Socket being another one of the whitelisted contracts (so that bridging from external chains is possible). While that might give users a nice "chain-abstracted" view of broad liquidity, it introduces Socket as an important trusted party that could block any withdrawal for any externally bridged token.

We believe Kinto can be become a Stage 1 rollup even though the chain restricts its usage only to KYC-ed users and Socket controls most of the tokens, as this case does not differ in practice with dYdX v3, where Circle has in practice the same power.

Proposed segmentation for universal chains and appchains

We also understand that the Stage 1 designation might confuse users as they might think it implies that they would be able to exit in all circumstances, as with Stage 1 universal chains.

For this reason, we propose to visually distinguish Stage 1 and Stage 2 chains based on whether they're universal, where users can always withdraw using the appropriate tokens and applications, or appchains, where rollup operators cannot easily rug users, but additional application-specific risks may exist.

[

image

1752×988 107 KB

](https://europe1.discourse-cdn.com/flex017/uploads/l2beat/original/1X/6acf2a80f0766ee65a28d7cdb96a29415d79068e.jpeg)

Within this framework, we differentiate rollup operators (e.g., dYdX, Kinto) from application-specific operators (e.g., Circle, KYC providers, Socket), who can restrict user access to coins or applications even on otherwise decentralized chains (e.g., USDC on Ethereum). The distinction blurs when these roles merge, as might happen in certain appchains, and might require broader discussions.

Summary of the changes (TLDR)

- We're updating the requirements to obtain the Stage 1 designation to the higher-level principle proposed above. We'll give projects a 6 months period to adapt to the new requirements.
- We'll start to strictly enforce challenge periods of at least 7 days for Optimistic Rollups to be considered Stage 1.
- We'll visually distinguish Stages when applied to universal chains and when applied to appchains to better surface the app-specific risks that the latter ones might add.