This is a simplified and general version of the original Empire Stakes Back proposal, that is compatible withFernet.

Assumptions

1. There is at least an honest majority of stakers participating in block production.
2. The honest majority assumption is the same assumption used for ETH stakers.
3. The sequencer selection algorithim is not game-able and the distribution of blocks over a “upgrade” period will be random and fairly reflect the honest majority assumption.
4. There are at least two independent sequencer / staking clients at launch.
5. The default portal standard is NOT IMMUTABLE

and will follow the registry / incentives contract to facilitate upgrades.

Setup

This proposal defines two additional smart contracts on L1; a token contract and an incentives contract that are required for the proposal to operate.

The token contract is a standard ERC20 contract.

The incentives contract has the following functionality:

1. the sole ability to mint / burn tokens from the token contract handled on every valid block from rollups listed in the contract
2. a function for stakers to add stake or exit their stake from a given rollup.
3. a function to signal stake should be migrated from rollup A to new rollup B

:::warning

Both the incentives contract and the token contract are immutable.

:::

Deployments vs Upgrades

This proposal defines two distinct actions which can be combined.

*Credit to Adam as some of the thinking is borrowed from him.

Deployment

– The process of creating a new version of Aztec eligible for block reward and migrating stake to this version.

Upgrade

– The process for migrating state and defining a canonical version of Aztec that portals will follow.

The deployment flow is not specific to this proposal

, and it is recommended that all proposals, even immutable proposals use a similar incentives contract. This ensures there will only be one staking and token system for all versions of Aztec

, regardless of if a version decides to have an upgrade mechanism or be immutable.

How it works

Deployment

1. Anybody deploys a new rollup contract to Ethereum and a deployment proposal registered against the incentives contract.
2. Stakers can vote for this proposal in one of two ways
3. When committing to a block on L1, the sequencer tags the block header with a deployment signal

4. The deployment signal is the Ethereum address of the rollup contract that should be registered for block rewards.
5. If the block becomes canonical, the deployment proposal is counted as submitted.
6. The deployment signal is the Ethereum address of the rollup contract that should be registered for block rewards.
7. If the block becomes canonical, the deployment proposal is counted as submitted.
8. By signalling via an L1 transaction that they wish their stake to migrate to the new rollup upon activation.
9. Existing stakers are only egible to vote.
10. Existing stakers are only egible to vote.
11. When committing to a block on L1, the sequencer tags the block header with a deployment signal
12. The deployment signal is the Ethereum address of the rollup contract that should be registered for block rewards.
13. If the block becomes canonical, the deployment proposal is counted as submitted.
14. The deployment signal is the Ethereum address of the rollup contract that should be registered for block rewards.
15. If the block becomes canonical, the deployment proposal is counted as submitted.
16. By signalling via an L1 transaction that they wish their stake to migrate to the new rollup upon activation.
17. Existing stakers are only egible to vote.
18. Existing stakers are only egible to vote.
19. Requirments for block reward eligibility
20. For a rollup to be registered on the incentives contract and be eligible for block rewards, it must be receive 750 votes either as a deploy signal or directly on L1.
21. For a rollup to be registered on the incentives contract and be eligible for block rewards, it must be receive 750 votes either as a deploy signal or directly on L1.
22. Time lock
23. New rollups on the incentives contract have a mandatory timelock of 1440 blocks before they can be activated.
24. During this time, the old rollup will still receive blocks and stake can be migrated via the deployment signal or directly on L1.
25. Stake can be migrated directly on L1 or via processing a block on the old chain with the upgrade signal.
26. A rebate could be possible on L1.
27. A rebate could be possible on L1.
28. New rollups on the incentives contract have a mandatory timelock of 1440 blocks before they can be activated.
29. During this time, the old rollup will still receive blocks and stake can be migrated via the deployment signal or directly on L1.
30. Stake can be migrated directly on L1 or via processing a block on the old chain with the upgrade signal.
31. A rebate could be possible on L1.
32. A rebate could be possible on L1.
33. New Chain Activation
34. After the time lock has expired, the new chain can be activated by sending a block. All stakers that signalled for the deployment will be moved across in the first block on the new rollup.
35. After the time lock has expired, the new chain can be activated by sending a block. All stakers that signalled for the deployment will be moved across in the first block on the new rollup.

At this point in time no upgrade has occurred

– the new deployment is just registered to receive block rewards.

Upgrading

If the rollup is an upgrade, two additional things will occur

1. State Cloning
2. The new rollup will copy the state of the last canonical rollup in it’s first block when it is activated.
3. The new rollup will copy the state of the last canonical rollup in it’s first block when it is activated.
4. Canonical Chain
5. The chain will be marked as canonical

if it has > 51% of the total stake at activation – note a chain can only be marked as canonical at activation, regardless of how the stake increases or decreases after this point.

- If the chain is canonical, all portals using the portal standard

will allow their assets to be spent on the canonical chain and they will become un-backed on the other chain.

1. The chain will be marked as canonical

if it has > 51% of the total stake at activation – note a chain can only be marked as canonical at activation, regardless of how the stake increases or decreases after this point.

1. If the chain is canonical, all portals using the portal standard

will allow their assets to be spent on the canonical chain and they will become un-backed on the other chain.

Block Rewards

Block rewards are paid out as follows :

RollupBlockReward = (RollupStake / TotalStake)* TotalBlockReward

Rollups must have a stake above MIN_STAKE

to be eligible for rewards and stop free-riders.

MIN_STAKE

= 10% of canonical chain's stake

Upgrade Flow

[

upload_96b2b58457edd736d31cfa8eef7c877e

1920×1826 138 KB

](https://europe1.discourse-cdn.com/business20/uploads/aztec/original/1X/9d660f307725d00787e1b307ae32849d9a64ced6.jpeg)

How is stake moved between rollups?

Outside of an upgrade, stakers can manually move their stake on L1 by interacting with the incentives contract. Stake can only be excited or entered for activated

rollup implementations.

Moving Stake to new Rollup Implementations

When a sequencer produces a block that contains a deploySignal

, they commit their stake to move across to that rollup upon activation.

The rollup circuits are modified to achieve this as follows:

1. The rollup circuits need to prove the sequencer is a valid staker with the lowest VRF for a given rollup for Fernet. This is achieved by passing in a staking tree root for the rollup, as defined by the incentives contract.

2. If the block header contains an deploy signal, the block must also prove their stake has been added to the stake migrations mapping.

Each deployProposal

stores a stake migration mapping on L1. e.g stakeMigrations = mapping(uint256=> mapping(uint256 => bytes32))

, indexed by rollupId

e.g stakeMigrations[1][2]

will be the root of the tree for all users migrating from rollup 1 to 2.

Stake is stored in an indexed merkle tree on the incentives contract for each rollup. The leaf of the tree is:

PublicKey: STAKER_PUBLIC_KEY

1. The first block of a new rollup, which activates the rollup, must also take all stakeMigrations

and insert them into the staking tree for the rollup being activated and zero the mapping. This tree root is generated as part of the ZKP proving the first block of rollup 2. (Stakers commit to this rollup, so it can be trusted to move stake).

Preventing Rapid Stake changes

Ethereum uses activation delays for new stakers to prevent an actor acquiring > 50% of the stake quickly. The incentives contract would have similar functionality to ensure that:

1. The incentives contract implements an entrance queue and an exit queue, to ensure that stake can’t be dialed up and down quickly to exploit portal assets in a hardfork.
2. Only those already staking can vote on a new rollup.
3. Stake can’t be added to a registered, but not yet active, rollup via the incentives contract, stake can only be migrated.

Malicious Deploys or Upgrades

The following steps must be completed do to pull off a malicious upgrade:

1. Achieve social consensus on the features and block-height for an upgrade to be deployed.
2. Convince client teams to upgrade their software to point to a new rollup implementation.
3. Convince solo stakers / institutional stakers to upgrade to the latest client software – if this occurs it is assumed that rollup blocks will be published with a new upgrade signal
4. Convince client teams to upgrade their software to point to a new rollup implementation.
5. Convince solo stakers / institutional stakers to upgrade to the latest client software – if this occurs it is assumed that rollup blocks will be published with a new upgrade signal
6. Assuming 1 was not possible the malicious entity could bribe 750 / 1000 sequencers to include the deployment to register for block rewards
7. They would then have to convince 51% of the stake to migrate to their version to steal any assets.

Achieving all these is considered impossible for a malicious proposal, if the honest majority of stakers assumption holds.

An itterative path to mainnet launch

Aztec will need multiple iterations to be created over time to get to feature completion / security. This should not come at the cost of fragmenting the eco-system via multiple token contract / incentives contracts which can only realistically be deployed once, at launch.

The deployment flow outlined in the Empire Stakes Back Again, paves the way for future version of Aztec to be deployed and all use the same staking and token system, while preserving the option for subsequent versions of Aztec to decide how or if they can upgrade on a case by case basis.