Brett Palatiello & Gautham Anant

Hey everyone! Please see Brandon's introduction for more context on how these posts should be interpreted and we actively encourage the community to engage with them.

Intro

In our post introducing the new security model we abstracted how Operators allocate Unique Stake. We explained that Operators directly assign percentages of their stake. In reality, the process is more nuanced. This post explains the mechanics of allocating and slashing Unique Stake for Operators.

Initialization

Magnitudes

represent the proportion of an Operator's Total Stake allocated as Unique Stake across various Operator Sets and as non-slashable stake. For each Strategy (stake token), an Operator starts with a Total Magnitude of 1 \times 10^{18}

. However, for legibility, we will assume a Total Magnitude of 10,000.

Let's consider an Operator who has been delegated 100 EIGEN but has not yet allocated any magnitude or registered with an Operator Set. Thus, all their EIGEN is non-slashable.

Magnitude

Proportion

EIGEN

Non-slashable

10,000

100%

100

Total

10,000

100%

100

Allocation

Operators set an allocation delay, which defines the time it takes to move non-slashable magnitude to an Operator Set. This feature allows Operators to accommodate stakers who may want to withdraw from their Operator before new allocations take effect. Of course, for this to be possible, the allocation delay must be longer than the withdrawal delay. If an Operator wants to change their allocation delay it takes 17.5 days to take effect, but this duration is still being debated.

Suppose the Operator sets an allocation delay of 5 days and allocates 2,000 magnitude to \texttt{EigenDA_EIGEN}

(an Operator Set where the AVS name is to the left of the underscore and the Strategy to the right). The allocation is considered pending during this 5-day period, meaning it is not yet slashable by \texttt{EigenDA_EIGEN}

and cannot be allocated elsewhere.

Magnitude

Proportion

EIGEN

\texttt{EigenDA_EIGEN}

(pending allocation)

2,000

20%

20

Non-slashable

8,000

80%

80

Total

10,000

100%

100

Allocations automatically take effect after the allocation delay. Allocations can only be queued from non-slashable magnitude, not from pending allocations or magnitudes already allocated to an Operator Set.

Magnitude

Proportion

EIGEN

\texttt{EigenDA_EIGEN}

2,000

20%

20

Non-slashable

8,000

80%

80

Total

10,000

100%

100

After the allocation delay, the operator can register for \texttt{EigenDA_EIGEN}

, and only upon registration does their EIGEN become slashable by the Operator Set. Therefore, the Operator has 100 EIGEN in Total Stake, with 20 EIGEN in Unique Stake allocated to \texttt{EigenDA_EIGEN}

. Now, \texttt{EigenDA_EIGEN}

can exclusively slash the Operator for up to 20 EIGEN.

Let’s assume that the Operator also allocates 2500 and and 3000 magnitude to \texttt{Eoracle_EIGEN}

and \texttt{Lagrange_EIGEN}

, respectively.

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

3,000

30%

30

\texttt{Eoracle_EIGEN}

2,500

25%

25

\texttt{EigenDA_EIGEN}

2,000

20%

20

Non-slashable

2,500

25%

25

Total

10,000

100%

100

Deallocation

To reduce an allocation, Operators queue a deallocation

which moves magnitude from an existing OperatorSet allocation to its non-slashable magnitude. While the duration is still up for debate, the deallocation process takes 17.5 days (420 hours), during which the deallocation is marked as pending. This period allows AVSs to update their view of Unique and Total Stake to reflect the Operator's reduced allocation and to leave a reasonable amount of time for all tasks created before the deallocation to be slashed for, if necessary. We plan to allow Operator Sets to customize this duration based on their specific needs in a future release.

Let's say the Operator deallocates 1,000 magnitude from \texttt{Lagrange_EIGEN}

. While the deallocation is pending it remains slashable and cannot be cancelled. For example, \texttt{Lagrange_EIGEN}

can still slash the Operator for up to 30 EIGEN.

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

2,000

20%

20

\texttt{Lagrange_EIGEN}

(pending deallocation)

1,000

10%

10

\texttt{Eoracle_EIGEN}

2,500

25%

25

\texttt{EigenDA_EIGEN}

2,000

20%

20

Non-slashable

2,500

25%

25

Total

10,000

100%

100

After 17.5 days, the pending deallocation takes effect automatically and the magnitude becomes non-slashable, allowing it to be reallocated if desired.

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

2,000

20%

20

\texttt{Eoracle_EIGEN}

2,500

25%

25

\texttt{EigenDA_EIGEN}

2,000

20%

20

Non-slashable

3,500

35%

35

Total

10,000

100%

100

Note, for UX purposes, a restriction of the protocol is that an (operator, strategy) pair can only have 1 pending allocation to or deallocation from an Operator Set at a time. They cannot have 1 of both, they can only have 1 or 0 of either.

Deposits and Withdrawals

Deposits and withdrawals do not affect magnitudes but scale the Total and Unique Stake of the Operator. Deposits take effect immediately, making the newly deposited EIGEN slashable by any Operator Set the Operator has allocated to. For example, if a staker deposits 100 more EIGEN into EigenLayer, and thus increasing the Operator's delegated EIGEN to 200, it would result in the following:

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

2,000

20%

40

\texttt{Eoracle_EIGEN}

2,500

25%

50

\texttt{EigenDA_EIGEN}

2,000

20%

40

Non-slashable

3,500

35%

70

Total

10,000

100%

200

Withdrawals, like deallocations, must be queued and can be completed after 17.5 days (duration still being debated). During this time, the withdrawal remains slashable. The withdrawal can be completed in a separate transaction to redeem tokens after the withdrawal delay. Withdrawals are not exposed to slashing that occurs after the withdrawal delay but before the withdrawal was completed.

Slashing

When an Operator Set decides to slash an Operator, it specifies a slashing proportion. The magnitude, and thus the EIGEN, allocated to that Operator Set by that Operator is decreased by that proportion.

Building off our previous example with 200 EIGEN, if \texttt{Lagrange_EIGEN}

slashes the Operator by 50%, it would result in the following:

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

1,000

11%

20

\texttt{Eoracle_EIGEN}

2,500

28%

50

\texttt{EigenDA_EIGEN}

2,000

22%

40

Non-slashable

3,500

39%

70

Total

9,000

100%

180

Note, slashing by one Operator Set does not affect the magnitudes or EIGEN allocated to other Operator Sets. This is the definition of Unique Stake.

Slashing with a pending deallocation

Another common case is when an Operator has a pending deallocation and they are slashed. Let’s assume instead that the Operator has a pending deallocation of 500 magnitude from \texttt{Lagrange_EIGEN}

:

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

1,500

15%

30

\texttt{Lagrange_EIGEN}

(pending deallocation)

500

5%

10

\texttt{Eoracle_EIGEN}

2,500

28%

50

\texttt{EigenDA_EIGEN}

2,000

22%

40

Non-slashable

3,500

39%

70

Total

10,000

100%

200

If Lagrange slashes the Operator by 50%, then both the current allocation and pending deallocation are slashed by that proportion, resulting in:

Magnitude

Proportion

EIGEN

\texttt{Lagrange_EIGEN}

750

7%

15

\texttt{Lagrange_EIGEN}

(pending deallocation)

250

3%

5

\texttt{Eoracle_EIGEN}

2,500

28%

50

\texttt{EigenDA_EIGEN}

2,000

22%

40

Non-slashable

3,500

39%

70

Total

9,000

100%

180

Future Posts

We will shortly put out posts to discuss and request feedback on important ideas and open issues that will impact what we have described in this article, such as the “Allocator” role and the time delay for slashing.

Conclusion

Understanding the detailed mechanics of allocating and slashing Unique Stake is crucial for Operators. By managing magnitudes, allocation delays, and being aware of the slashing mechanisms, Operators can more effectively control their delegated stake and exposure to potential penalties.

Disclaimer

The Eigen Labs Research Team uses the Forum as a space to share research on the protocol, to preview elements and ideas that may or may not become part of upcoming releases, and to explore ways of using, analyzing, and thinking about the EigenLayer protocol and ecosystem. Unlike our blog posts and social media announcements, which focus on formal updates and decisions, the Forum will be more interactive and experimental, allowing us to exchange ideas and gather feedback from a wider group.