---

title: Attestations description: A description of attestations on proof-of-stake Ethereum. lang: en

---

A validator is expected to create, sign and broadcast an attestation during every epoch. This page outlines what these attestations look like and how they are processed and communicated between consensus clients.

What is an attestation? {#what-is-an-attestation}

Every epoch (6.4 minutes) a validator proposes an attestation to the network. The attestation is for a specific slot in the epoch. The purpose of the attestation is to vote in favor of the validator's view of the chain, in particular the most recent justified block and the first block in the current epoch (known as `source` and `target` checkpoints). This information is combined for all participating validators, enabling the network to reach consensus about the state of the blockchain.

The attestation contains the following components:

- `aggregation_bits`: a bitlist of validators where the position maps to the validator index in their committee; the value (0/1) indicates whether the validator signed the `data` (i.e. whether they are active and agree with the block proposer)
- `data`: details relating to the attestation, as defined below
- `signature`: a BLS signature that aggregates the signatures of individual validators

The first task for an attesting validator is to build the`data`. The `data` contains the following information:

- `slot`: The slot number that the attestation refers to
- `index`: A number that identifies which committee the validator belongs to in a given slot
- `beacon_block_root`: Root hash of the block the validator sees at the head of the chain (the result of applying the fork-choice algorithm)
- `source`: Part of the finality vote indicating what the validators see as the most recent justified block
- `target`: Part of the finality vote indicating what the validators see as the first block in the current epoch

Once the `data` is built, the validator can flip the bit in`aggregation_bits` corresponding to their own validator index from 0 to 1 to show that they participated.

Finally, the validator signs the attestation and broadcasts it to the network.

Aggregated attestation {#aggregated-attestation}

There is a substantial overhead associated with passing this data around the network for every validator. Therefore, the attestations from individual validators are aggregated within subnets before being broadcast more widely. This includes aggregating signatures together so that an attestation that gets broadcast includes the consensus `data` and a single signature formed by combining the signatures of all the validators that agree with that `data`. This can be checked using `aggregation_bits` because this provides the index of each validator in their committee (whose ID is provided in`data`) which can be used to query individual signatures.

In each epoch a validator in each subnet is selected to be the`aggregator`. The aggregator collects all the attestations it hears about over the gossip network that have equivalent `data` to its own. The sender of each matching attestation is recorded in the `aggregation_bits`. The aggregator then broadcasts the attestation aggregate to the wider network.

When a validator is selected to be a block proposer they package aggregate attestations from the subnets up to the latest slot in the new block.

Attestation inclusion lifecycle {#attestation-inclusion-lifecycle}

1. Generation
2. Propagation
3. Aggregation
4. Propagation
5. Inclusion

The attestation lifecycle is outlined in the schematic below:

Rewards {#rewards}

Validators are rewarded for submitting attestations. The attestation reward is dependent on two variables, the `base reward` and the `inclusion delay`. The best case for the inclusion delay is to be equal to 1.

```
attestation reward = 7/8 x base reward x (1/inclusion delay)
```

Base reward {#base-reward}

The base reward is calculated according to the number of attesting validators and their effective staked ether balances:

```
base reward = validator effective balance x 2^6 / SQRT(Effective balance of all active validators)
```

Inclusion delay {#inclusion-delay}

At the time when the validators voted on the head of the chain (`block n`), `block n+1` was not proposed yet. Therefore attestations naturally get included **one block later** so all attestations who voted on `block n` being the chain head got included in `block n+1` and, the **inclusion delay** is 1. If the inclusion delay doubles to two slots, the attestation reward halves, because to calculate the attestation reward the base reward is multiplied by the reciprocal of the inclusion delay.

Attestation scenarios {#attestation-scenarios}

Missing Voting Validator {#missing-voting-validator}

Validators have a maximum of 1 epoch to submit their attestation. If the attestation was missed in epoch 0, they can submit it with an inclusion delay in epoch 1.

Missing Aggregator {#missing-aggregator}

There are 16 Aggregators per epoch in total. In addition, random validators subscribe to **two subnets for 256 epochs** and serve as a backup in case aggregators are missing.

Missing block proposer {#missing-block-proposer}

Note that in some cases a lucky aggregator may also become the block proposer. If the attestation was not included because the block proposer has gone missing, the next block proposer would pick the aggregated attestation up and include it into the next block. However, the **inclusion delay** will increase by one.

Further reading {#further-reading}

- Attestations in Vitalik's annotated consensus spec
- Attestations in eth2book.info

Know of a community resource that helped you? Edit this page and add it!