At the current state, even after Deneb and the introduction of blobs, Attestations are responsible for 90% of the traffic on the Consensus Layer's gossip network. Right now, the Attestation

propagation format is the same as the one used during internal operation within the Consensus Client. which is as follows:

class Attestation(Container): aggregation_bits: Bitlist[MAX_VALIDATORS_PER_COMMITTEE] data: AttestationData signature: BLSSignature

where AttestationData is just:

class AttestationData(Container): slot: Slot index: CommitteeIndex # LMD GHOST vote beacon_block_root: Root # FFG vote source: Checkpoint target: Checkpoint ```.

This format totals a maximum of 256 bytes for each attestation (assuming that len(aggregation_bits)=32).

However, it could be possible to reduce this size by at least a 15ish% by defining a different format for AttestationData, which could be as follows:

```python class NetworkAttestationData(Container): slot: Slot index: CommitteeIndex # LMD GHOST vote beacon_block_root: Root # FFG vote source_digest: BytesVector8 target_digest: BytesVector8

where source_digest = source. HashTreeRoot () [0:8] and target_digest = target. HashTreeRoot () [0:8]. This format requires clients to keep track of an internal map of possible CheckpointDigest

(BytesVector8

) as they sync, and maps them to their equivalent Checkpoint

. This reduction would save 64 bytes per attestation which is equivalent to >25ish% of a single attestation. If this is also applied to aggregates and proofs

, it should also decrease the bandwidth for the average node as well. Potentially, bandwidth-wise, something like this can allow for a doubling of the current Blob per block. A consideration to make is that perhaps BytesVector8

as a digest size might open up for possible collision attacks. However, an attacker needs to do so in the 12 seconds of slot time to create a collision with 2 recent CheckpointDigests

. Probably 16

is a better size for something like this (but I am not a cryptographer). this would also require that at each block, each consensus client would need to generate some digests for some potential future checkpoints as well, e.g., generates 4 checkpoints for block root 0xabcd

spanning 4 epochs into the future.
```