Answers to reviewer 606A questions:

1. Interpretation for *perfect completeness for CKS*: If all inputs to CKS.Verify except an alleged aggregated signature have been generated honestly and if the alleged aggregated signature itself is accepted by AS.Verify, then CKS.Verify on this aggregated signature and the honest inputs always outputs 1. This is, to the highest extent possible, the counter-notion to perfect completeness for aggregation for AS (Definition 3.1.), and it is used in the security proof …

Interpretation for *soundness for CKS*: … This is used in the security proof of …

Interpretation for *unforgeability for CKS*:

Answers to reviewer 606B comments:

1. We do achieve accountability which we formally define (section H2, additional material). An accountable light client system (Definition H5) fulfills *completeness*, *soundness*(i.e., any adversarial prover interacting with honest validators should not be able to generate an acceptable light client proof in some epoch if less than a threshold of honest validators signed a message in that epoch and in every epoch there is at least (another threshold) of validators that are honest), *accountability* *completeness* (an honest node running consensus protocol can detect and output a set of misbehaving validators if it sees a light client proof for some message m contradicting m’ which in turn was signed by some threshold of validators) and *accountability soundness* (an adversary interacting with a single honest validator cannot prove that the honest validator misbehaved). In Section H3 we instantiate and, under concrete set of assumptions, formally prove security of an accountable light client system (Section H.3.2.) using our committee key scheme (based in turn on our custom SNARKS and an aggregatable signature).TO DO: Maybe add a sentence about the generality of this approach.
2. Our approach to building light clients can be easily generalized to the case when validators’ stakes are not equal. In that case, one substitutes the current bitmask of 0s and 1s with a stake mask where the 1s are replaced with suitably defined integer proportions of stake. In turn, our custom SNARKs take as input commitments to multisets of public keys (a key is repeated as many times as dictated by the corresponding stake proportion) and stake masks. The rest of the construction is a natural adaptation of our light client system using this modification.

Answers to reviewer 606C comments: