It is an honor to post on this respected forum. Posting here was suggested by lrettig,@cdetrio, and @chfast, but any errors were introduced after their review.

The procedure below can be used to limit last-revealer attacks in randomness generation on the Ethereum 2.0 Beacon Chain. The full text below is copied from paul.oemm.org/commit_reveal_subcommittees.pdf.

Abstract.

The goal is for a committee to output a number with little or no bias. We consider the commit-reveal procedure (during the commit-period, each committee member publicly commits to a secret contribution to the final output, and during the reveal-period, reveals their secret) in which the last-revealer can create bias by choosing whether to reveal. We generalize commit-reveal by allowing each secret to be committed and revealed by a subcommittee of members, significantly reducing the probability of a last-revealer attack. Unfortunately, the proof is for existence – a lot of work remains in finding optimal parameters for this generalized procedure.

1. Introduction

Definition. A committee number generation procedure

is an effective procedure for members of a committee to agree upon and output a number.

Definition. A biased

committee number generation procedure is one in which committee member(s) can manipulate the output to take a specific form.

Remark. For the purposes of this document, we will not define a measure for bias, but just an axiom that more manipulation corresponds to higher bias.

Definition. A honest

committee member never creates bias.

1. Commit-Reveal

Remark. Our focus is on commit-reveal procedures, which are designed to limit bias. Alternative procedures are not considered.

Definition. A commit-reveal procedure

is a committee number generation procedure with steps

1. during a commit period, each party may commit to a secret by publishing its hash,
2. during a reveal period, each party may reveal their secret corresponding to the hash, and
3. revealed secrets are combined into an output value.

All secrets, hashing, revealing, and combining have predefined formats which all members agree are not vulnerable to bias.

Remark. If all parties are honest, i.e. commit independently of other secrets and reveal at the appropriate time, then we consider there to be zero bias.

2.1 Non-Obligatory Commit-Reveal

Definition. A non-obligatory commit-reveal procedure

allows members who fail to commit or reveal to be ignored when combining reveals to form the final output.

Remark. An obligatory procedure may be attacked by members who refuse to reveal unless doing so creates bias. In the worst-case, they may refuse to ever reveal. We avoid these attacks by focusing on non-obligatory procedures.

2.2 Ordered Commit-reveal

Definition. An ordered commit-reveal procedure

puts an order on when each member reveals its secret.

Definition. A last-revealer attack

occurs when the last member to reveal decides whether to reveal based on the effect on the final output, creating bias.

Remark. Both ordered and non-ordered procedures are vulnerable to last-revealer attacks. Non-ordered procedures are

especially vulnerable if a non-honest member can delay their decision to reveal until the end of the reveal-period, once most or all others have revealed. If one party controls the last several revealers, then they can reveal only certain ones, giving more bias. These attacks can even be forced by hindering communication channels. To limit these attacks, our focus is on the ordered procedure.

Remark. In an ordered procedure, influencing the ordering can influence who reveals last, which leads to a last-revealer attack. We assume a random ordering (perhaps the randomness comes inductively from earlier committee-generated numbers). We consider a model in which the probability of a last-revealer attack is equal to the ratio of dishonest members – i.e. each member has equal probability of being the last revealer.

2.3. Subcommittee Commit-Reveal

Definition. A subcommittee commit-reveal procedure

partitions the committee into subcommittees, and each secret is committed and revealed by a subcommittee. For notation, we use ``n

-subcommittee'' to denote a n

-member subcommittee.

Remark. This is a generalization of the special case with all 1-subcommittees.

Claim. Consider a large committee with ratio p

of non-honest committee members, with 0<p<1

. The ordered subcommittee non-obligatory commit-reveal procedure can have lower probability of a last-revealer attack if it is not limited to all 1-subcommittees.

Proof.

For all 1-subcommittees, the probability of a last-revealer attack is p

, since p

is the probability of a dishonest member as the last revealer. To prove the claim, we show a case in which having subcommittee(s) of size greater than one will result in smaller probability of a last-revealer attack. Consider a committee of all 1-subcommittees except the last subcommittee is a 2-subcommittee. Then the probability of a last-revealer attack is (for large committee approximately) p^2

, since p^2

is the probability that both members of the last subcommittee are dishonest, since if at least one of the two is honest then they will reveal. Since p^2<p

for 0<p<1

, we have a lower probability of last-revealer attack, as desired.

Remark. Subcommittee procedures bring forth an omniscience attack

– if an attacker knows all secrets during commit-time, then they may have complete bias on output by choosing appropriate secret(s). Fortunately, this attack can be reduced to arbitrarily small probability by adding enough early-revealing 1-subcommittees, with probability of omniscience attack decreasing exponentially(!) with each additional 1-subcommittee. The last subcommittee(s) can still have many members, with probability of last-revealer attack decreasing exponentially(!) with each additional member. So there is a trade-off between probability of last-revealer attack and omniscience attack, but each can be made arbitrarily small by choosing subcommittees of appropriate size with appropriate ordering.

1. Conclusion

Remark. We have reduced the problem of a public commit-reveal procedure to a composition of procedures by subcommittees. Current work follows.

1. How to generate a number among each subcommittee. Perhaps recursively perform a subsubcommittee procedure. But the subcommittee problem is different because the secret should be kept private by the subcommittee until reveal-time, and there is less vulnerability to last-revealer attack. Depending on parameters and requirements, a reasonable choice may be Diffie-Hellman generalized to many parties. At least one member would have to reveal the subcommittee's true shared secret along with everything required to verify it.

2. Model new attacks where a single party controls many committee members, can buy secrets early, or even competes with other dishonest members.

3. Find the optimal sizes of subcommittees as a function of full committee size and ratio of honest members. We have already derived formulas for some of the counting problems, and are working on numerically plotting probabilities for small committee sizes, and also hope to analytically solve for optimal subcommittee sizes. Heuristically, half of the committee members could be 1-subcommittees which reveal earlier, and the rest on subcommittees of size one greater than the subcommittee which reveals just before them.

If there is existing work in any of these directions, this author would be grateful for a hint.