Let us suppose that, for pure proof of stake, the main chain uses a simple RANDAO-based RNG. That is, the main chain has an internal state, R

, and every validator, upon depositing, commits to the innermost value in a hash onion, H(H(H(.....S.....)))

; the state stores the “current commitment value” C_{V_i}

for each validator V_i

. To create a block, a validator V_i

must include the preimage of the C_{V_i}

value (call it H^{-1}(C_{V_i})

), and the following state changes happen: (i) C_{V_i} := H^{-1}(C_{V_i})

, (ii) R := R\ xor\ H^{-1}(C_{V_i})

; that is, the current commitment value for that validator is set to the preimage, so next time the validator will have to unwrap another layer of the onion and reveal the preimage of that

, and the preimage is incorporated into R

.

This has the property that when some validator gets an opportunity to propose a block, they only have two choices: (i) make a block, where they can only provide one specific hash value, the preimage of the previous value they submitted, or (ii) not make a block. The validator knows what the new R

value would be if they make a block, but not the R

value if they don’t (as it’s based on the next validator’s preimage, which is private information); the validator can either choose the value that they know or a value that they don’t know, the latter choice coming at a cost of their block reward B

.

In this post, we will introduce a model for thinking about attacks on this beacon, including proving some thresholds for security, and describe how to reduce them.

Suppose an attacker has \alpha

stake, and they want to make an attack whose purpose is to revert

the beacon chain (note that this is distinct from earlier separate analysis on profitably manipulating the output

). The attacker’s job is to outrun the honest chain. The attacker has an advantage, however: the attacker does not need to always take the first chance they get to make a block, rather they can wait for a later chance if the randomness that block gives is more favorable to them. The attacker can precalculate long “chains” of alternate skips and blocks, and publish the one that is most favorable to them.

We can model this as follows. An attacker chooses a target h

, the height they want to build, and s

, the maximum number of skips that they allow themselves (s

is constrained by time, and the need to overtake the growth of the honest chain; since honest nodes have 1-\alpha

stake, this implies s = h * \frac{\alpha}{1-\alpha}

). Let C(s, h)

be the expected number of possible chains the attacker could make with s

total skips and h

height. We know that C(s, 1) = \alpha

, because for any number of skips, there is a probability of \alpha

that a block can be made with that number of skips. We also know that C(0, h) = \alpha^h

, as making an h-length chain with no skips would require getting lucky h

times. We can also calculate C(s, h)

recursively:

C(s, h) = \sum_{i=0}^{s} \alpha * C(i, h-1)

Basically, if you've allotted a total of s

skips, then you can use any i \le s

of them immediately, and then the remaining skips for the remaining heights. Notice that this also implies:

C(s+1, h+1) = \alpha * C(s+1, h) + \sum_{i=0}^{s} \alpha * C(i, h) = \alpha * C(s+1, h) + C(s, h+1)

This proves that C is a Pascal's triangle, with all cofficients multiplied by \alpha^h

. We can then use known formulas to calculate the limiting behavior of diagonals

in C, so we can find the expected number of chains that, for some given \alpha

, in the long run, overtake honest nodes. Focusing on the "expected number of possible chains" is a mathematical convenience; the expected number of possible chains is guaranteed to be at least as large as the probability that there is at least one chain, so for any \alpha

where this value approaches zero, the attacker cannot win. First of all, let's look at the diagonal s=h

(corresponding to 50% honest nodes). This value can be written as \frac{(2n)!}{(n)!(n)!}

; using a simplified Stirling's approximation of n! \approx (\frac{n}{e})^n

, this gives:

\frac{(\frac{2n}{e})^{2n}}{(\frac{n}{e})^{2n}} = 2^{2n} = 4^n

This is for a Pascal's triangle in general. In our specific case, the n

in question is h

, and we need to multiply by \alpha^h

, so we get (4\alpha)^h

. An attacker can in the long run outrun an honest chain with 50% of nodes if the attacker has more than 25%. Now, let's generalize our formula; suppose we want the diagonal s = k*h

for arbitrary k

. Then, we get:

\frac{\alpha^h((k+1)h)!}{(kh)!h!} \approx \frac{\alpha^h(\frac{(k+1)h}{e})^{(k+1)h}}{ (\frac{kh}{e})^{kh} (\frac{h}{e})^{h}} = \frac{\alpha^h(k+1)h^{(k+1)h}}{kh^{kh}h^h} = \frac{\alpha^h(k+1)^{(k+1)h}}{k^{kh}}

We want to look at k = \frac{\alpha}{1-\alpha}

(k+1 = \frac{1}{1-\alpha}

), so we get:

\frac{\alpha^h * (\frac{1}{1-\alpha})^\frac{h}{1-\alpha}}{(\frac{\alpha}{1-\alpha})^{(\frac{h\alpha}{1-\alpha})}} = (\frac{\alpha * (\frac{1}{1-\alpha})^\frac{1}{1-\alpha}}{(\frac{\alpha}{1-\alpha})^{(\frac{\alpha}{1-\alpha})}})^h

The inside expression crosses 1 at ~0.36 (ie. below 0.36, as h

increases it converges toward zero, and above 0.36 as h

increases it converges toward infinity). This answer coincides almost perfectly with the results from the simulations I've written that try to use pruned A* pathfinding to determine if there exists a path that can overtake the main chain.

Now can we increase the scheme's security to make it better than 0.36? Certainly! Let us reuse the trick from proof of activity of requiring a small set of notaries to approve each block (we'll say 2 at first). Thus, any block will require approval from 3 of 3 parties, so we replace \alpha

with \alpha^3

and k

with ((\frac{1}{1-\alpha})^3 - 1)

. Our expression becomes:

(\frac{\alpha^3(k+1)^{k+1}}{k^k})^h = (\frac{\alpha^3((\frac{1}{1-\alpha})^3)^{(\frac{1}{1-\alpha})^3}}{(\frac{1}{1-\alpha})^3 - 1)^{(\frac{1}{1-\alpha})^3 - 1)}})^h

The inside expression now crosses 1 at \alpha \approx 0.425

(also roughly confirmed by the pathfinding simulations). If we add two more notaries, it goes up to 0.455. We can get arbitrarily close to 0.5 simply by adding more notaries; at the ~100 notary level that becomes possible with aggregate signatures, the security reduction is negligible

.

Conclusions:

- RANDAO beacons are promising because of their high degree of simplicity compared to threshold schemes, lack of dependence on fancy cryptography or distributed key generation procedures, as well as their lack of an in-protocol liveness threshold
- The “naive” version of the RANDAO beacon does have a vulnerability to attacker path selection, and only has 51% attack resilience up to \alpha \approx 0.36
- However, this can be largely remedied by requiring a committee of notaries to verify each block (which is already a prudent idea because it reduces forking); adding an extra 4-of-4 notary committee can by itself increase resilience to \alpha \approx 0.455

. However, going too far in this direction compromises the goal of lacking an in-protocol liveness threshold.

- Adding a cryptoeconomic aggregate signature, or BLS/STARK-based aggregate signature, would make reversion attempts even more expensive

Possible future work:

- Calculate the exact resilience constants for M-of-N oracles
- Think of ways to dynamically adjust for different liveness levels