AIM[1]-002: Better Randomness and Proposer Selection

liayoo 2021-01-28

Problem Description

- Within a set of validators, a proposer should be selected every epoch, pseudo-randomly
- Randomness can be generated asynchronously & individually
- Randomness should be unpredictable before the generation time
- Randomness should be deterministic at the generation & afterwards (all nodes should generate the same random number/string for a given epoch)
- Known security attacks
    - Stake grinding
    - DDoS

Proposed Changes

AS-IS

- Pseudo-random number generator (PRNG) seed = genesisHash + current epoch
- Proposer selection logic
    - “Round-robin selection tells us far in advance who's going to be producing blocks and when” (ref)

```
updateProposer() {
        ...
        const validators = this.node.bc.lastBlockNumber() < 1 ?
                        lastNotarizedBlock.validators : this.getWhitelist();
        const seed = '' + this.genesisHash + this.state.epoch;
        this.state.proposer = Consensus.selectProposer(seed, validators);
        ...
}
```

```
static selectProposer(seed, validators) {
        const alphabeticallyOrderedValidators = Object.keys(validators).sort();
        const totalAtStake = Object.values(validators).reduce((a, b) => { return a + b }, 0);
        const randomNumGenerator = seedrandom(seed);
```

[1] AI Network Improvement Memo. Visit https://docs.ainetwork.ai for the full list.

```
    const targetValue = randomNumGenerator() * totalAtStake;

    let cumulative = 0;
    for (let i = 0; i < alphabeticallyOrderedValidators.length; i++) {
        cumulative += validators[alphabeticallyOrderedValidators[i]];
        if (cumulative > targetValue) { return alphabeticallyOrderedValidators[i]; }
    }
    return null;
}
```

TO-BE

- seed = H(last_votes_hash(N-1) + H(last_votes_hash(N-2) + … + H(last_votes_hash(N-99) + last_votes_hash(N-100))) + current epoch
    - Similar to **randao mix** (ref)
    - The last_votes contains both proposer’s and other +⅔ validators' votes (a lot of unknown and randomness introduced), so it’s less susceptible to a malicious proposer’s tweaking of other manipulable block information, such as transactions.

Alternatives / Additional Measures

- Commit-reveal randao
    - Eth2 beacon chain
- Robust Round Robin (paper)
    - 5 sec rounds
    - 0.5-1 min tx latency
    - 1500tps
    - SGX / PoW needed at initialization? (for creating long-term reliable identities)
- Verifiable secret sharing (VSS)
    -
- Verifiable delay functions (VDF)
    - “algorithms that take a long time to execute and can't be sped up by running the algorithm on multiple computers at the same time”
    - If an attacker is able to determine the effect of their reveal before time is up, using VDF has no advantage (hardware dependent?)
    - => A VDF research group was recently coordinated with the goal of producing low-cost hardware that approaches the limits for VDF computation time
- Verifiable random functions (VRF)
    - Probabilistic (can have more than 1 proposers selected)

    - Susceptible to selection bias. “The chosen leader may bias the protocol output, e.g., by skipping his turn” (ref)
- Sentry nodes
    - Doesn’t improve randomness but obfuscates nodes’ IP addresses and can help prevent DDoS attacks
    - Examples
        - Polkadot - why they deprecated sentry nodes (github issue)
- Validator set shuffling
    - Periodically & constantly shuffle the validator set
    - Would be introducing a new concept of “committee”, a selected set of validators
    - Examples
        - https://github.com/ethereum/annotated-spec/blob/master/phase0/beacon-chain.md#compute_shuffled_index

Links

- https://blog.coinfabrik.com/comparison-of-pos-projects-unbiased-leader-election/

Document History

Date	Who	Change	Notes
2021-01-28	liayoo	Initial draft	
2021-02-02	liayoo, minsulee2, platfowner, cshcomcom, shyun-comcom	Internal review	
2021-05-13	platfowner	Published	