VIP-193: COMMITTEE-BASED POA

Ziheng (Peter) Zhou
VeChain Foundation
peter.zhou@vechain.com

Zhijie Ren
VeChain Foundation
zhijie.ren@vechain.com

November 6, 2019

- VIP: 193
- Title: Committee-Based POA
- Author: Ziheng Zhou (peter.zhou@vechain.com) and Zhijie Ren (zhijie.ren@vechain.com)
- Category: Core
- Status: Draft
- CreateAt: 2019-11-05

1 Overview

The Proof-of-Authority consensus algorithm [1], or POA in short, is efficient of using network bandwidth. It divides time into rounds with a fixed length and assumes that the majority of its (authorized) nodes perform consensus in the same round.

In each round, nodes select a leader (the node responsible for generating a new block) based on the round number, block height and their local views of active nodes. Therefore, the procedure can be considered instant, which allows more time for transmitting transaction (TX) data in each consensus round.

POA relies on the staked reputation (by nodes) and economic incentives to keep the system secure. However, it cannot stop malicious leaders causing temporary inconsistency. To improve the security of POA, we propose to introduce a committee to endorse the new block generated in each consensus round. The verifiable random function (VRF) [2] is used for nodes to locally decide their committee memberships. With the committee mechanism, a malicious leader would have to collude with the committee to cause inconsistency. However, the property of VRF guarantees that the committee is selected randomly. Therefore, it makes it much more difficult for adversaries to launch such attacks.

2 Specifications

2.1 Notations

Symbol	Description
N	Total number of nodes
$u = 1, 2, \ldots, N$	node index
$H(\cdot)$	Cryptographic hash function
$\texttt{SIG}_u(\cdot)$	Signature of u
$\texttt{PK}_u$	Public key of u

$\mathtt{SK}_u$	Private key of u
$\rho(\cdot)$	Merkle root of the input set
r	Consensus round number
$l_r = 1, 2, ..., N$	Leader in round r
s_r	Block summary produced by l_r
τ_r	$s_r \parallel \mathtt{Sig}_{l_r}(s_r)$
ϵ	Predefined threshold to determine committee memberships
d	Number of endorsements required for forming a valid block
Σ_r	Pending transaction set prepared by l_r
$B^*_{u,r}$	Last block of the canonical chain recognized by u in round r
$H_{vrf}, \Pi_{vrf}, V_{vrf}$	Functions that compute hash, generate proof and verify proof defined in the VRF scheme, respectively
$\mathtt{M}_r$	Common VRF message generated for committee selection
ω, ω^*	Branch and trunk
m, L	Epoch number and the number of consensus rounds in one epoch
b_m	Random beacon computed using randomness from the last block of the m'th epoch

2.2 Committee Membership

The verifiable random function can be considered as a public-key version of keyed cryptographic hash. The hash of a given message can only be computed by the owner of the private key and can be verified by anyone given the public key $\mathtt{SK}$ and message $\mathtt{M}$. The scheme defines the following functions:

- $\pi \leftarrow \Pi_{vrf}(\mathtt{SK}, \mathtt{M})$ - function that generates VRF proof;
- $\{\mathtt{TRUE}, \mathtt{FALSE}\} \leftarrow V_{vrf}(\mathtt{SK}, \mathtt{M}, \pi)$ - function that verify VRF proof;
- $h \leftarrow H_{vrf}(\mathtt{SK}, \mathtt{M}) = H(\pi)$ - function that computes the hash.

Node u determines his committee membership by Algorithm 1. Here $\mathtt{M}_r$ is the common VRF message computed in round r and ϵ a predefined threshold. See Section 2.7 for details of the computation of $\mathtt{M}_r$. Given the VRF proof and the corresponding public key, anyone can verify the membership claimed by u by Algorithm 6 as shown below.

Algorithm 1 Procedure for u to determine his committee membership.

```
Compute M_r
Compute π_{u,r} = Π_vrf(SK_u, M_r) and λ = H(π_{u,r})
if λ ≤ ε then
    Return (TRUE, π_{u,r})
else
    Return FALSE
end if
```

Algorithm 2 Procedure for anyone to validate u's claimed committee membership.

Compute $\texttt{M}_r$
if $V_{vrf}(\texttt{PK}_u, \pi_{u,r}) = \texttt{FALSE}$ **then**
    Return $\texttt{FALSE}$
end if
$\lambda \leftarrow H(\pi_{u,r})$
if $\lambda \leq \epsilon$ **then**
    Return $\texttt{TRUE}$
else
    Return $\texttt{FALSE}$
end if

2.3 Block Summary

At the beginning of round r, leader l_r computes the summary of the new block, s_r, and broadcast it for the committee members to endorse the block. Algorithms 3 and 4 shows the computation and validation process of s_r, respectively. In Algorithm 4, $B^*_{u,r}$ is the last block of the canonical chain locally recognized by u at the beginning of round r. Lines 6-10 dictate that l_r's view on the canonical chain must be consistent with that of u.

Algorithm 3 Procedure for l_r to compute and broadcast s_r.

Prepares Σ_r and computes $\rho\left(\Sigma_r\right)$
Get the last block on the canonical chain, $B^*_{l_r}$ and computer $h_0 = H(\texttt{HEADER}(B^*_{l_r,r}))$
$s_r \leftarrow h_0 \parallel r \parallel \rho\left(\Sigma_r\right)$
Broadcast $\left(s_r, \texttt{Sig}_{l_r}(s_r)\right)$

Algorithm 4 Procedure for u to validate s_r.

1: Compute l_r
2: **if** s_r is not sent from l_r **then**
3:     Return $\texttt{FALSE}$
4: **end if**
5: Verify $\texttt{Sig}_{l_r}(s_r)$
6: Extract $B^*_{u,r}$ and compute $\texttt{HEADER}(B^*_{u,r})$
7: **if** $\texttt{HEADER}(B^*_{u,r}) \neq \texttt{HEADER}(B^*_{l_r,r})$ **then**
8:     Return $\texttt{FALSE}$
9: **end if**
10: Return $\texttt{TRUE}$

2.4 Endorsement

After receiving block summary s_r, if node u is a committee member, he then executes Algorithm 5 to endorse the block proposed by l_r. Here we define $\tau_r = s_r \parallel \texttt{Sig}_{l_r}(s_r)$.

Algorithm 5 Procedure for u to endorse a new block.

Verify s_r by Algorithm 4
Broadcast $\Big(u, s_r, \texttt{SIG}_u(\tau_r), \pi_{u,r}\Big)$

Once a node receives an endorsement message from u, he first verify s_r by Algorithm 4 and then validate u's endorsement by Algorithm 6.

Algorithm 6 Procedure for validating u's endorsement.

Verify $\texttt{SIG}_u(\tau_r)$
Verify u's committee membership by Algorithm 2

2.5 Block

Algorithm 7 describes the procedure for leader l_r to publish a new block in round r.

Algorithm 7 Procedure for l_r to publish a new block.

Compute and broadcast s_r by Algorithm 3
Broadcast TX set Σ_r
Wait for d valid endorsements and arrange the committee members such that $u_1 < u_2 < \cdots < u_d$
$\Gamma_r \leftarrow \texttt{SIG}_{u_1}(\tau_r) \,\|\, \texttt{SIG}_{u_2}(\tau_r) \,\|\, \ldots \,\|\, \texttt{SIG}_{u_d}(\tau_r)$
Broadcast $\left(s_r, \texttt{Sig}_{l_r}(s_r), \texttt{Sig}_{l_r}(\Gamma_r), \{u_i, \texttt{SIG}_{u_i}(\tau_r), \pi_{u_i,r}\}_{i=1}^{d}\right)$
Construct header body Ω and broadcast $\texttt{Sig}_{l_r}(\Omega)$

A block header body Ω contains the following elements:

1. Hash of the parent header $H(\texttt{HEADER}(B^*_{l_r,r}))$
2. Round number r
3. TX Merkle tree root $\rho\left(\Sigma_r\right)$
4. Signature $\texttt{Sig}_{l_r}(s_r)$
5. Committee member list $(u_1 < u_2 < \cdots < u_d)$
6. Merkle tree root of VRF proofs $\{\pi_{u_i,r}\}_{i=1}^{d}$
7. Merkle tree root of signatures $\{\texttt{SIG}_{u_i}(\tau_r)\}_{i=1}^{d}$
8. Signature $\texttt{Sig}_{l_r}(\Gamma_r)$

A block consists of a header $\left\{\Omega, \texttt{Sig}_{l_r}(\Omega)\right\}$ and TX set Σ_r.

Algorithm 8 Procedure for validating a block.

Verify s_r by Algorithm 4
Verify $\rho\left(\Sigma_r\right)$
for $i = 1, 2, \ldots, d$ **do**
    Verify u_i's endorsement by Algorithm 6
end for
Construct Γ_r and verify $\texttt{Sig}_{l_r}(\Gamma_r)$
Construct header body Ω and verify $\texttt{Sig}_{l_r}(\Omega)$
Verify Σ_r

Note that we only list the elements related to the committee mechanism for simplicity. Algorithms 7 and 8 should be adjusted accordingly to accommodate other elements such as the gas limit, state root, receipt root, etc.

2.6 Canonical Chain Rule

In this subsection, we provide the rules for selecting the canonical chain (or the *trunk*) based on the committee mechanism. Let ω be a branch, i.e., a chain of blocks. Given a block B on the branch, i.e., $B \in \omega$, we define the weight of the chain starting from B till the end of ω, $W(\omega, B)$, as the size of the set that contains all the leaders and committee members who have their signatures included in at least one block on the chain. Note that a node is only counted once.

Given two branches ω_1 and ω_2, let B_0 be the last block shared by ω_1 and ω_2, i.e., $B_0 \in \omega_1 \cap \omega_2$ and $\forall B' \in \omega_1 \cap \omega_2$, $B' \preceq B_0$. Let $N(\omega, B)$ be the operator that finds the next block of B on branch ω. We choose the canonical chain ω^* from two branches ω_1 and ω_2 by Algorithm 9 as shown below:

Algorithm 9 Rules for choosing canonical chain ω^* from branches ω_1 and ω_2.

$B_1 \leftarrow N(\omega_1, B_0)$
$B_2 \leftarrow N(\omega_2, B_0)$
while $W(\omega_1, B_1) = W(\omega_2, B_2)$ **and** $N(\omega_1, B_1) \neq \texttt{nil}$ **and** $N(\omega_2, B_2) \neq \texttt{nil}$ **do**
    $B_1 \leftarrow N(\omega_1, B_1)$
    $B_2 \leftarrow N(\omega_2, B_2)$
end while
if $W(\omega_1, B_1) > W(\omega_2, B_2)$ **then**
    $\omega^* \leftarrow \omega_1$
else if $W(\omega_1, B_1) < W(\omega_2, B_2)$ **then**
    $\omega^* \leftarrow \omega_2$
else
    if B_1 is published later than B_2 **then**
        $\omega^* \leftarrow \omega_1$
    else
        $\omega^* \leftarrow \omega_2$
    end if
end if

2.7 Random Beacon

In Algorithms 1 and 2, nodes need to compute common message $\mathtt{M}_r$ to determine and validate committee memberships. It is desirable to add certain randomness in the computation such that adversaries only know their committee memberships in a limited number of rounds in future.

To do that, we divide the consensus process into epochs each of which lasts for L rounds where L is a predetermined fixed number. Let $\hat{B}_{u,r}^m$ be the last block on the canonical chain observed by u in the m'th epoch. Node u can compute a random beacon b_u^m from $\hat{B}_{u,r}^m$ as:

$$b_u^m \leftarrow H(\mathtt{Sig}_{l_r}(\Gamma_r)). \tag{1}$$

Here we define $b_u^0 = H(B_{\text{genesis}})$.

Beacon b_u^m is then used to calculate VRF messages for the next epoch. In particular, we compute message $\mathtt{M}_{r'}$ in round r' as:

$$\mathtt{M}_{r'} \leftarrow H(b_u^m \parallel r') \tag{2}$$

where $mL + 1 \leq r' \leq m(L + 1)$.

Now let us consider the case when the leader who generates $\hat{B}_{u,r}^m$ is malicious. Since signatures from the committee members are deterministic, the best he can do is to go through every combination of d signatures and pick the one that maximizes his interest. The number of possible trials is fairly limited and therefore, we significantly limit the leader's influence on the beacon.

2.8 Optimizing Bandwidth Usage

A more efficient usage of network bandwidth often results in a higher system throughput. By more efficient, we mean that there is more time to transmit TXs in each round of consensus. To do that, we detach the endorsing process from the block validation process. In particular, committee members can endorse a new block without receiving and validating the complete set of TXs included in the block. See Algorithms 3 and 4 for details. In this way, we would be able to significantly shorten the whole endorsing process and potentially allow more bandwidth used for TX transmission.

Note that TXs will be checked by nodes when validating any newly received block and any invalid TX will cause the block to be discarded even though all the endorsements are valid.

2.9 Block Reward

With the committee mechanism in place, the block reward needs to be shared between the leader and committee members. We propose to give half of the total block reward to the leader and the rest equally distributed to the committee members.

2.10 Double-Spending Attack

The main goal of introducing the committee mechanism is to make it more difficult for adversaries to cause inconsistency. Perhaps the most damaging of such attacks is the double-spending attacks (DSAs) [3] where adversaries are allowed to take control of a few consecutive consensus rounds such that they can produce two parallel branches to launch a DSA. Here we are going to infer a lower bound for the probability of launching a k-block DSA.

Let p_ϵ be the probability of a node being selected as a committee member. This probability is directly related to threshold ϵ and is equal to every node thanks to VRF. Let us assume that there are f malicious nodes that can behave arbitrarily. To produce two valid, but contradicting blocks in the same round, adversaries must control both the leader and d committee members. The probability of existence of d committee members being malicious can be computed as:

$$F(p_\epsilon, d, f) = \sum_{i=d}^{f} \binom{f}{i} {p_\epsilon}^i \, (1 - p_\epsilon)^{f-i} \tag{3}$$

Therefore, the probability of adversaries controlling the committee for k consecutive rounds is $F(p_\epsilon, d, f)^k$. Table 2 lists the values of this probability with different inputs. We set f according to the Byzantine Fault Tolerance (BFT) assumption. It is a security assumptioned that is considered reasonable and widely used in practice.

$N = 101, f = \lfloor \frac{N}{3} \rfloor$	$k = 1$	$k = 5$	$k = 10$
$p_\epsilon = 0.1, d = 7$	4.17×10^{-2}	1.26×10^{-7}	1.59×10^{-14}
$p_\epsilon = 0.1, d = 8$	1.41×10^{-2}	5.58×10^{-10}	3.11×10^{-19}
$p_\epsilon = 0.15, d = 11$	6.78×10^{-3}	1.43×10^{-11}	2.04×10^{-22}

Table 2: Probabilities $F(p_\epsilon, d, f)^k$ for different values of p_ϵ, d and k.

Now let us consider the worse-case scenario when

1. A malicious leader is selected to generate the block in the last epoch from which the random beacon for the current epoch is computed;
2. There are k consecutive rounds in the current epoch where the leaders are all malicious.

Let c be the total number of valid endorsements received by the leader mentioned in the first condition. The probability of launching a k-block DSA in the scenario can be computed as:

$$F^*(k, p_\epsilon, d, f) = \binom{c}{d} F(p_\epsilon, d, f)^k. \tag{4}$$

In general, we can consider $F^*(k, p_\epsilon, d, f)$ as the lower bound for the probability of launching a k-block DSA.

References

[1] Vechain development plan and whitepaper. https://cdn.vechain.com/vechainthor_development_plan_and_whitepaper_en_v1.0.pdf, 2018.

[2] D. Papadopoulos, D. Wessels, S. Huque, M. Naor, J. Vcelak, L. Reyzin, and S. Goldberg. Making NSEC5 practical for DNSSEC. *IACR ePrint*, 1999. https://eprint.iacr.org/2017/099.pdf.

[3] N. Satoshi. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008.