SBA 2.0 Outline

Toghrul Maharramov
Dusk Foundation

March 29, 2019

Abstract

SBA 2.0 (SBA from now on) stands for Segregated Byzantine Agreement and is an iterative upgrade to the SBA★ algorithm introduced in the Dusk Network whitepaper. SBA is a permission-less Proof-of-Stake consensus protocol (state machine replication algorithm) with statistical finality. This paper outlines the improvements utilized in the algorithm before displaying the abstract definitions of the protocol building blocks followed by a concrete definition of the SBA consensus protocol.

Contents

1 Introduction

Consensus protocols represent a fundamental problem in distributed computing which has been analyzed and researched for decades. Formally defined by [quote], the condition have gone through multiple iterations before being formalized as Byzantine Generals' Problem [quote]. Originally designed for closed-network systems, the "open-access" consensus protocols have been theorized, but never successfully implemented until the arrival of Bitcoin. Satoshi Nakomoto, the pseudonym of the Bitcoin whitepaper author, had defined a Proof-of-Work protocol which became the first truly permission-less consensus protocol to be implemented on a public network. However, Bitcoin's consensus protocol (sometimes dubbed the "Nakomoto Consensus" in scientific literature) has a couple of shortcoming:

1. Probabilistic finality, meaning that the probability of a block being final increases as more blocks are "mined" on top of it.

2. Energy waste required to secure the blockchain.

Since the launch of Bitcoin, various alternative consensus protocols with variable success have been researched and developed. While variations of classic Byzantine Fault-Tolerance have been for corporate "private blockchains", none of those are of particular interest to this paper. Chain-based Proof-of-Stake consensus protocols, on the other hand, represent a "simulation" of Proof-of-Work, without the added need for a wasteful energy consumption. Unfortunately, the chain-based Proof-of-Stake protocols suffer from a drastically-decreased "openness" as well as continuing to retain the probabilistic finality of Proof-of-Work algorithms.

Algorand solves the earlier stated issues of the chain-based Proof-of-Stake protocols by utilizing a cryptographic sortition to extract a unique committee of validator nodes from a pool of candidate nodes during each consensus step to reach near-instant finality without a need for a partially permissioned access to the consensus. However, Algorand requires unrealistically large committees to make sure that a probability of a fork is statistically negligible and is secure if at least 66 percent of the total Algorand supply holders are honest and participate in the consensus.

2 Our Contribution

SBA utilizes multiple in-house improvements to create a robust and efficient consensus protocol which is permission-less and produces statistically final blocks.

Statistical finality is, to our knowledge, a brand new finality definition introduced by the team at Dusk Network. The consensus protocol is considered to have statistical finality if a probability of a fork is negligible in the conditions

defined and required by the consensus protocol.

Blind bid procedure, to our knowledge, has been the first conceptual definition of private Proof-of-Stake, introduced in the Dusk Network whitepaper. Despite having gone through multiple iterations since the aforementioned paper has been publish, the refined procedure fulfills the requirements defined in the paper.

Deterministic sortition is an improvement of cryptographic sortition, which enables the underlying protocol to pseudo-randomly extract committees from a large set of nodes non-interactively, as long as the function used for the extraction is a Random Oracle.

Agreement phase, to our knowledge, is the first definition of a consensus phase which enables the network to probabilistically exit the distributed execution of a loop without producing consensus partitions in the process.

SBA is the first consensus protocol with near-instant finality (i.e. statistical finality) which does not require each of the consensus steps to retain the requirements of an honest majority to remain secure. As long as each consensus round involves an honest majority, the percentage of honest participation in the individual steps does not affect the security of the protocol.

3 Abstract Definition

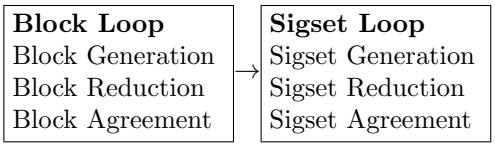
SBA is a hybrid Proof-of-Stake protocol with statistical finality guarantees. The protocol is based on an Honest Majority of Money (an Adversary can corrupt nodes controlling up to f percent of the total stake value [$\geq 3f + 1$]) and weak network synchrony assumptions.

The roles in the protocol are split between two different node types: Block Generators and Provisioners. Block Generators retain their privacy, with the proofs of stake computed in zero-knowledge to preserve the anonymity of the Block Generator. On the other hand, Provisioners are required to deanonymize their stakes and remain transparent about their activities in the consensus while their stake remains valid.

Both bids and stakes have a registration period of t, which begins when a Bid or Stake transaction is included in a final block and is required to elapse before the full-node is eligible to participate in the consensus.

SBA protocol can be conceptually defined with two inner loops (Block Loop and Sigset Loop), with execution of the Sigset Loop dependent on the successful termination of the Block Loop. Block Loop is responsible for forming a consensus on a uniform block and Sigset Loop is responsible for forming a

uniform signature set of the Provisioners which attested the winning candidate block during the Block Reduction phase and is required to form a certificate notarazing the block. The team is currently working on an upgrade of the SBA protocol which only utilizes a Block Loop.



The security model of SBA is based on Snow White, a provably secure Proof-of-Stake protocol. SBA is secure under a Δ-delayed mildly adaptive adversary (an adversary who is required to choose the nodes controlling a maximum of f percent of the total stake he/she is willing to corrupt Δ rounds before the actual corruption) and in a weakly synchronous network with a propagation delay of up to δ seconds.

3.1 Block Generator

Block Generator is the first of the two full-node types eligible to participate in the consensus. To become a Block Generator, a full-node has to submit a Bid transaction.

The Block Generator is eligible to participate in one phase - Block Generation phase. In the aforementioned phase, Block Generators participate in a non-interactive lottery to be able to forge a candidate block.

3.2 Provisioner

Provisioner is the second of the two full-node types eligible to participate in the consensus. To become a Block Generator, a full-node has to submit a Stake transaction. Unlike a Block Generator, a Provisioner node is required to deanonymize the value of the stake to be able to participate in the consensus. While it is technically possible to obfuscate the stake value, the team has decided against the latter as the addition of stake value obfuscation would have slowed down the consensus and simultaneously increased the block size.

The Provisioner is eligible to participate in five phases - Block Reduction, Block Agreement, Sigset Generation, Sigset Reduction and Sigset Agreement phases.

3.3 Binary Reducation

The Binary Reduction algorithm is an adaptation of an algorithm defined by Turpin and Coan, with two variations of the algorithm forming the core of SBA

unlike Algorand, which utilizes the original algorithm as an extenstion of BBA★.

Binary Reduction is a two-step algorithm, with the input of the second step depending on the output of the first one. If no consensus have been reached on a uniform value, the algorithm return value and waits for the next instantiation.

Binary Reduction acts as a uniform value extraction function which is then fed through the Set Agreement algorithm (outlined below) before exiting the loop in case of a successful termination of the Set Agreement algorith,

3.4 Set Agreement

During the conduction of a technical analysis of Algorand, the team has discovered a vulnerability, which increases the probability of a the consensus forking, dubbed a "timeout fork". As a result, the team has concluded that SBA requires an additional step in both of the inner loops to guarantee statistical finality under the basic assumptions of the protocol.

Set Agreement is an asynchronous algorithm running in parallel with the inner loop. The protocol includes two variation of the Set Agreement algorithm - Block Agreement and Sigset Agreement. Successful termination of one of the variations of the algorithm indicates that the relevant inner loop has been successfully executed and the protocol can proceed to the next loop. The algorithm provides a statistical guarantee that at least one honest node has received a set of votes exceeding the minimum threshold required to successfully terminate the respective phase of the protocol.

3.5 Blind Bid

The Blind Bid scheme enables the Block Generator to provably preserve his/her anonymity while participating in the consensus.

The proof of the bid alongside the proof of the score correctness is included in every Coinbase transaction of a candidate block. The proof does not leak information about the bidder or the size of the bid. The Blind Bid proves the validity of the bid (through a Merkle opening proof) and the correctness of the score (through the bid size proof).

4 Key Notations

r - consensus round.
$\{0,1\}s$ - consensus step. $0s$ if in Block Production loop and $1s$ if in Sigset Production loop.
BG^r - set of valid bids N_b for round r
$C_{r,s}$ - consensus committee for round r and step s.

B^r - block for round r.
$B_c^{r,s}$ - candidate block for round r and step s.
$CERT^r$ - certificate for round r.
$\pi^{r,s}$ - zero-knowledge proof for round r and step s

5 Concrete Definition

5.1 Block Loop

5.1.1 Block Generation

If $s = 1$ and a node N_i receives a B^{r-1} with a committed `sigset` of at least t valid identities or if $s > 3$, the node N_i can proceed to the Block Generation phase **iff**:

1. $s = 1 \bmod 3$
2. $N_{i,b} \in BG^r$
3. $score_{r,s} > T_{BG}$

The Block Generation phase is executed in the following manner:

1. Construct $CERT^{r-1}$
2. Generate $\pi_{r,s}$ for $score_{r,s}$
3. Construct $B_c^{r,s}$
4. Propagate $score_{r,s}$, $\pi_{r,s}$, $CERT^{r-1}$ and $B_c^{r,s}$

5.1.2 Block Reduction

STEP 1

Once a node N_i receives a $B_c^{r,s}$ corresponding to $score_{r,s}$ received before the time out of Δ_{SCORE} or times out of Δ_{BLOCK}, the node N_i can proceed to the Block Reduction phase **iff**:

1. $s = 2 \bmod 3$
2. $N_{i,s} \in C_{r,s}$

The STEP 1 of Block Generation phase is executed in the following manner:

1. if $B_c^{r,s} \neq \varnothing$ then $vote = SIGN(v = \mathcal{H}(B_c^{r,s}))$; else $vote = SIGN(v =$BLOCKREGEN$)$
2. Propagate the values v and $vote$ from the previous step.

STEP 2

Once a node N_i receives at least t equal values v corresponding to $B_c^{r,s}$ OR BLOCKREGEN received before the time out of Δ_{STEP} or times out of Δ_{STEP}, the node N_i can proceed to the Block Reduction phase **iff**:

1. $s = 3 \bmod 3$
2. $N_{i,s} \in C_{r,s}$

The STEP 2 of Block Generation phase is executed in the following manner:

1. if $count(v) \geq t$ AND $v \neq$ BLOCKREGEN then verify that $B_c^{r,s}$ is a valid block
2. if $count(v) \geq t$ then $sign = SIGN(v)$; else $sign = SIGN($BLOCKREGEN$)$
3. Propagate the values v and $vote$ from the previous step.

5.1.3 Block Agreement

If a node N_i receives at least t equal values v corresponding to $B_c^{r,s}$ received before the time out of Δ_{STEP}, the node N_i can proceed to the Block Agreement phase **iff**:

1. $N_{i,s} \in BG^r$

otherwise it should skip the Block Agreement phase and proceed to Block Generation phase.

The Block Agreement phase is executed in the following manner:

1. Construct aggregated signature `aggsig` for the previous Block Reduction phase
2. Construct a set of valid identities `pkset`
3. Propagate `aggsig` and `pkset`

Proceed to Block Generation phase.

5.2 Sigset Loop

5.2.1 Sigset Generation

If $s = 1$ and a node N_i receives a at least t valid `aggsig` and `pkset` or if $s > 3$, the node N_i can proceed to the Sigset Generation phase **iff**:

1. $s = 1 \bmod 3$
2. $N_{i,s} \in C_{r,s}$

The Sigset Generation phase is executed in the following manner:

1. Construct `sigset`
2. Propagate `sigset`

5.2.2 Sigset Reduction

STEP 1

Once a node N_i receives a valid `sigset` received before the time out of Δ_{SIGSET} or times out of Δ_{SIGSET}, the node N_i can proceed to the Sigset Reduction phase **iff**:

1. $s = 2 \bmod 3$
2. $N_{i,s} \in C_{r,s}$

The STEP 1 of Block Generation phase is executed in the following manner:

1. if `sigset` $\neq \varnothing$ then $vote = SIGN(v = \mathcal{H}())$; else $vote = SIGN(v =$SIGSETREGEN$)$
2. Propagate the values v and $vote$ from the previous step.

STEP 2

Once a node N_i receives at least t equal values v corresponding to `sigset` OR SIGSETREGEN received before the time out of Δ_{STEP} or times out of Δ_{STEP}, the node N_i can proceed to the Sigset Reduction phase **iff**:

1. $s = 3 \bmod 3$
2. $N_{i,s} \in C_{r,s}$

The STEP 2 of Sigset Generation phase is executed in the following manner:

1. if $count(v) \geq t$ AND $v \neq$ SIGSETREGEN then verify that `sigset` is a valid signature set
2. if $count(v) \geq t$ then $sign = SIGN(v)$; else $sign = SIGN($SIGSETREGEN$)$
3. Propagate the values v and $vote$ from the previous step.

5.2.3 Sigset Agreement

If a node N_i receives at least t equal values v corresponding to `sigset` received before the time out of Δ_{STEP}, the node N_i can proceed to the Sigset Agreement phase **iff**:

1. $N_{i,s} \in BG^r$

otherwise it should skip the Sigset Agreement phase and proceed to Sigset Generation phase.

The Sigset Agreement phase is executed in the following manner:

1. Construct aggregated signature `aggsig` for the previous Sigset Reduction phase
2. Construct a set of valid identities `pkset`
3. Propagate `aggsig` and `pkset`

Proceed to Sigset Generation phase.

6 Future Work

As mentioned previously, the upcoming upgrade to the SBA will not require a Sigset Loop to remain secure. This will drastically reduce block times as well as cutting down the number of steps required to reach the consensus to four.

Apart from that, the future update of this paper will feature concrete network parameters, as well as the extended formal proofs of the selected parameters alongside the proofs of the security of the consensus protocol.