An Unauthenticated BFT Consensus Algorithm in 6 Phases

Giuliano Losa

September 2021

1 Introduction

This document describes a new BFT consensus algorithm in the unauthenticated Byzantine model.

This algorithm is live under eventual synchrony, meaning that, assuming quorum intersection and quorum availability, the malicious nodes cannot prevent well-behaved nodes from reaching a decision. In contrast, in SCP, malicious nodes can delay a decision forever, and liveness can only be guaranteed if we assume that malicious nodes are eventually evicted from the slices of well-behaved nodes.

Like SCP, nodes following the algorithm use an amount of space that is bounded by a constant that is independent of the length of the execution.

The algorithm has 6 phases leading to a decision: suggest/proof; propose; vote-1; vote-2; vote-3; vote-4.

The key to the liveness guarantee is that the leader determines whether a value v is safe at a round when the vote-2 messages of a blocking set show that v is safe, while follower nodes determine that v is safe when the vote-1 messages of a blocking set show that v is safe. Thus, any value determined safe by a leader is also determined safe by all other well-behaved nodes.

Information-Theoretic HotStuff (IT-HS), which seems to be the state-of-the-art unauthenticated BFT consensus algorithm, uses 7 phases: suggest/proof; propose; echo; key-1; key-2; key-3; commit; decision.

The key to obtaining a 6 phases is to not rely on locks, contrarily to IT-HS, and instead require nodes to check that the leader's value is safe before doing anything. This way we know that as soon as f+1 endorse a value, then the value is safe. In IT-HS, a node that has no lock may endorse an unsafe value from a Byzantine leader, and thus an additional phase (the echo phase) is needed to first establish that the leader's value is safe.

2 The Algorithm

We assume that there are at most f malicious nodes out of n nodes with $3f < n$. Any set of $2f+1$ nodes or more is called a quorum, and any set of $f+1$ nodes or more is called a blocking set.

We describe the algorithm in a high-level round-by-round model. Each node executes a sequence of rounds, starting at round 0. Rounds are communication-closed and each round has a unique, predetermined leader. We assume that, before round GSR, the network is unreliable, i.e. messages can be lost; during and after GSR, we assume that every message sent by a well-behaved node is received.

2.1 Messages

A node can send the following messages.

- *proposal*, *vote-1*, *vote-2*, *vote-3*, *vote-4*, each containing a round and a value.
- A *suggest* message, containing
  1. the highest *vote-2* message that the node sent and
  2. the second highest round for which the node sent a *vote-2* message, noted *prev-vote-2*, and
  3. the highest *vote-3* message that the node sent.
- A *proof* message, containing
  1. the highest *vote-1* message that the node sent and
  2. the second highest round for which the node sent a *vote-1* message, noted *prev-vote-1*, and
  3. the highest *vote-4* message that the node sent.

2.2 State

Across rounds, a node only has to remember the highest vote-1 and vote-2, vote-3 and vote-4 message it sent, and the round of the second highest vote-1 and vote-2 messages it sent.

2.3 Evolution of a round

A round `r` proceeds as follows:

1. Upon starting the round, a node `n` does the following:
   (a) it broadcasts a *proof* message for the current round and
   (b) it sends a *suggest* message to the leader of the round.
2. When the leader has determined that value `v` is safe to propose in current round (as described in Rule 1), it broadcasts a *proposal* message for the current round and for `v`.
3. When a node determines that the leader's proposal is safe in the current round (as described in Rule 3), it broadcasts a vote-1 message for the current round and the leader's proposal.
4. A node that receives a quorum of *vote-1* messages for the current round and for the same value `v` sends a *vote-2* message for the current round and for `v`.
5. A node that receives a quorum of *vote-2* messages for the current round and for the same value `v` sends a *vote-3* message for the current round and for `v`.
6. A node that receives a quorum of *vote-3* messages for the current round and for the same value `v` sends a *vote-4* message for the current round and for `v`.
7. A node that receives a quorum of *vote-4* messages for the current round and for the same value `v` decides `v`

NOTE: in an FBQS, information must be propagated throughout the system using the cascading mechanism (e.g. if a blocking set sent `vote-2` then send it too). Information in `proof` and `suggest` messages may also need to be propagated.

Rule 1. *All values are safe in round 0. If* $r \neq 0$*, a leader determines that the value* `v` *is safe to propose in round* `r` *when the following holds:*

1. *A quorum* `q` *has sent suggest messages in round* `r`*, and*
2. *According to what is reported in suggest messages, either*
   (a) *no member of* `q` *sent any vote-3 before round* `r`*, or*
   (b) *there is a round* `r'` $<$ `r` *such that*
      i. *no member of* `q` *sent any vote-3 messages for a round strictly higher than* `r'`*, and*
      ii. *any member of* `q` *that sent a vote-3 message in round* `r'` *did so with value* `v`*, and*
      iii. *there is a blocking set* `b` *(e.g.* $f+1$ *nodes) that all claim in their suggest messages that* `v` *is safe at* `r'` *(see Rule 2).*

Rule 2. *We say that a node claims that* `v` *is safe in* `r'` *in a suggest message when either*

1. `r'` *is 0,*
2. *the node's highest vote-2 message, as reported in the suggest message, was sent at round* `r''` $\geq$ `r'` *and for value* `v`*, or*
3. *the second highest round for which the node sent a vote-2 message, as reported in the suggest message, is a round* `r''` $\geq$ `r'`*.*

A node that receives a proposal from the leader of the current round determines that the value `v` is safe in round `r` $\neq 0$ like the leader does, except that it uses *proof* messages instead of *suggest* message, *vote-4* instead of *vote-3*, and *vote-1* instead of *vote-2*:

Rule 3. *A node that receives a proposal from the leader of the current round determines that the value* `v` *is safe to propose in round* `r` *when:*

1. *A quorum* `q` *has sent proof messages in round* `r`*, and*
2. *According to what is reported in proof messages, either*
   (a) *no member of* `q` *sent any vote-4 before round* `r`*, or*
   (b) *there is a round* `r'` $<$ `r` *such that*
      i. *no member of* `q` *sent any vote-4 messages for a round strictly higher than* `r'`*, and*
      ii. *any member of* `q` *that sent a vote-4 message in round* `r'` *did so with value* `v`*, and*
      iii. *there is a blocking set* `b` *(e.g.* $f+1$ *nodes) that all claim in their suggest messages that* `v` *is safe at* `r'` *(as described in Rule 4)*

Rule 4. *We say that a node claims that* `v` *is safe in* `r'` *in a proof message when either*

1. `r'` *is 0,*
2. *the node's highest vote-1 message, as reported in the proof message, was sent at round* `r''` $\geq$ `r'` *and for value* `v`*, or*
3. *the second highest round for which the node sent a vote-1 message, as reported in the proof message, is a round* `r''` $\geq$ `r'`*.*

3 Liveness argument

Consider a round `r` that a) has a well-behaved leader and b) is sufficiently long for all the messages sent by well-behaved nodes to well-behaved nodes to be received.

Claim 1. *If all well-behaved nodes determine that the leader's value is safe, then a decision is made by all well-behaved nodes.*

Claim 2. *If a well-behaved node claims that `v` is safe at `r'` in its suggest message in round `r` $>$ `r'`, then there is a blocking set `b` composed entirely of well-behaved nodes such that, in any proof message in rounds greater or equal to `r`, every member of `b` claims that `v` is safe at `r'`.*

Proof. This rests on the fact that what is claimed safe by a well-behaved node can only increase. □

Claim 3. *A well-behaved node eventually determines that the leader's value `v` is safe.*

Proof. According to the rule that the leader uses to propose a value (rule 1), there are three cases. First, if the round is 0 then all well-behaved nodes trivially determine that the leader's value is safe.

Second (Item 2a), suppose that the round is not 0 and that the leader proposes `v` because a quorum `q` reports not sending any `vote-3` messages. Then, there is an entirely well-behaved blocking set `b` that never sent any `vote-3` messages. Since `vote-4` messages are sent in response to a quorum of `vote-3` messages, and since a quorum and a blocking set must have a well-behaved node in common, we conclude that no well-behaved node ever sent a `vote-4` message. Thus, once a well-behaved node `n` receives *proof* messages from all other well-behaved nodes, `n` concludes that the proposal is safe according to Item 2a of Rule 3.

Third (Item 2b), suppose that the round is not 0 and we have a quorum `q` and a round `r'` $<$ `r` such that:

1. no member of `q` sent any *vote-3* messages for a round strictly higher than `r'`, and
2. any member of `q` that sent a *vote-3* message in round `r'` did so with value `v`, and
3. there is a blocking set `b` (e.g. $\mathtt{f}+1$ nodes) that all claim in their *suggest* messages that `v` is safe at `r'` (see Rule 2).

We make the following observations:

a) By Item 1 above, no well-behaved node sent any *vote-4* message in any round higher than `r'`; otherwise, a quorum would have sent the corresponding `vote-3` messages and, by the quorum-intersection property, this contradicts Item 1.

b) By Item 2 above, for a similar reason, any well-behaved node that sent a `vote-4` message in round `r'` did so for value `v`.

c) By Item 3, there is a well-behaved node that claims that `v` is safe in `r'` in its *suggest* message. By Claim 2, we conclude that there is a blocking set `b` composed entirely of well-behaved nodes that claim in their *proof* messages that `v` is safe at `r'`.

By Items a), b), and c) and Rule 3, we conclude that, once every well-behaved node has received a *proof* message from every other well-behaved node, every well-behaved node determines that the leader's proposal is safe. □

Finally, by Claim 3 and Claim 1, we conclude that a decision is reached by all well-behaved nodes.