Fast Byzantine Finality Gadget

December 4, 2019

Version 0.1

1 Introduction

Blockchain finality guarantees that all correct blocks will not be reverted after they were committed to the blockchain. Many consensus protocols demonstrate only probabilistic finality meaning that there's some (small) probability that a block and all its transactions will be reverted. For example, in the Nakamoto consensus [Nak+08] there is only the probability that the block will be in a longer chain and the transactions will not be reverted. This probability increases with each next block generated in the same chain.

Provable finality implies that blocks will not be reverted after some agreement is reached. At the same time, third parties can check block finality without tracking the entire network.

In our article, we want to introduce the finality gadget called RANDPA, which provides provable finality. For blockchains with a relatively small amount of block producers, RANDPA achieves fast finality. During testing, we achieved finality in two seconds with 100 block producers. Fast finality is a key property for applications that require high performance and low latency, such as payment systems and gambling applications. RANDPA is based on the finality gadget GRANDPA [Gra], but we have made some improvements and optimizations for achieving faster finality for a specific group of blockchain protocols. However, our finality gadget requires a more strict synchronous model.

This article includes a core description of the protocol, and ideas for improving different aspects of the protocol. Some parts of this article may be refined or removed, and we will use the current description as a starting point for updating and confirming RANDPA security. Version 1.0 of this document will be final.

1.1 Related Work

Work in Progress...

The most closely related work is the GRANDPA protocol [Gra]. Casper the Friendly Finality Gadget (FFG Casper) was introduced to provide finality in an existing blockchain consensus protocol [BG17]. Later, it was presented in the Ethereum 2.0 specification [Eth]. A "Correct-by-Construction" Casper (CBC Casper) was proposed by Zamfir and instantiated as Casper The Friendly GHOST [Zam]. "Minimal CBC Casper" family of consensus protocols was presented in [Zam+18]. Nakamura *et al.* [NJH19] provided systematization of knowledge and formal verification of CBC Casper. Afgjort [Mag+19] is a finality layer presented in partially synchronous model. Wendy the View-Changing Finality Gadget is an concrete instantiation of Hot-Stuff described by Abraham *et al.* [AGM18].

BFT-based blockchain consensus protocols provide provable finality and do not require a finality gadget.

1.2 Our Contribution

Our protocol is based on the GRANDPA finality gadget [Gra]. The GRANDPA protocol serves as a general solution for blockchain finalization. RANDPA is optimized for blockchains that follow the longest-chain rule and include a relatively small amount of participants involved in block producing. Examples of such systems are defined in [Xia+19] as committee-based and DPoS-based consensuses. In DPoS consensus mechanism (e.g. EOS [Eos]), a consensus group is chosen via public stake delegation. In committee-based consensus (e.g. Ouroboros [Kia+17]), participants are randomly chosen via MPC procedure. The advantages of these approaches are fewer validators and, consequently, less time and greater stability of finalization.

For the sake of simplicity, we will refer to both consensus types as minority-based consensus and call the consensus group Block Producers. Block Producers generate new blocks according to the system rules.

We made the following modifications of the original GRANDPA protocol:

- We use the longest-chain rule instead of the GHOST one.
- The start and the end of finalization rounds are determined and depend on block height.
- Only block producers act as finality gadget protocol participants.
- There is no primary election procedure. A block producer of the last published block becomes a primary.

We implemented our protocol in the EOS based blockchain to improve EOS finalization time. RANDPA was implemented as an additional plugin that interacts with the EOS core via events model. Our plugin subscribes to incoming blocks events and it's enough to get all information needed for the protocol. So this approach allows us to use a dedicated thread for RANDPA with minimal synchronization points. We made minimal changes in EOS core. Namely, we expanded the network interface and finalized the mechanism for applying the final blocks. We achieved the finality of two seconds for a network with 100 block producers. Our solution is used by the DAOBet chain [1].

This paper is organized as follows. Section 2 contains a network model description and basic notions used in the RANDPA protocol. Section 3 provides a detailed description of the RANDPA protocol. Section 4 covers RANDPA security properties. Section 5 describes RANDPA implementation. Section 6 outlines future RANDPA optimizations.

2 Preliminaries

Network Model. RANDPA is based on the GRANDPA protocol, thus, we will use the notions presented in [Gra]:
Voters are protocol participants actively agreeing on block finalization by sending votes. Vote is a chain of block hashes and metadata that are signed with a voter's private key.

[1] `https://daobet.org`

Unlike GRANDPA's definition of voters, only block producers act as voters in RANDPA. So, voters and block producers are the same sets of participants.

We use a fully synchronous network model with a set of n voters, and for each finalization step at most $f < (n-1)/3$ voters are Byzantine. Byzantine voters can diverge from the specified protocol in any way.

Practical Byzantine Fault Tolerance. pBFT is a practical algorithm for Byzantine state machine replication which works in asynchronous systems [CL+99]. Voters are actively interacting with each other during the algorithm and, as a result, agree on a certain state of the system. The protocol is secure if at most $f = (n-1)/3$ nodes are Byzantine. Original pBFT implies that there is a client who sends a request to execute some operation on the state machine. One of the voters is selected as primary. The algorithm in a simplified form can be divided into 3 phases after receiving the client's request:

1. **Pre-Prepare**. Primary sends the PRE-PREPARE message to all other voters.
2. **Prepare**. If any honest voter, except primary, accepts the message from primary, then they record the PRE-PREPARE message. After that, the voter sends the PREPARE message to all voters and stores the messages of this type received from other voters.
3. **Commit**. After receiving $2f$ valid PREPARE messages, any honest voter sends the COMMIT message to all other voters. After the voter accepts $2f + 1$ COMMIT messages, possibly including its own one, it sends a reply to the client.

Then the client waits for $f+1$ replies with the same state from different voters and approves the result. The GRANDPA protocol, and, consequently, RANDPA are based on pBFT with some modifications.

Byzantine Finality Gadget Problem. We will briefly recall the definition of Byzantine finality gadget problem described in GRANDPA [Gra]. Let $\mathcal{F}$ be a protocol with a set of voters V, and less than third of them are Byzantine. We assume that there is a block production protocol P that runs at the same time as the finality gadget protocol $\mathcal{F}$. Voting rule A is a function that depends on a voter and its state, takes as input some block B and returns block B' at the head of a chain including B. If $\mathcal{F}$ finalizes some block B, then, eventually, either $\mathcal{F}$ will finalize some descendant of B or all the chains with head $A(B)$ will contain some descendant of B for all voters at all future states. Then, we can say that the system $(\mathcal{F}, P, A)$ achieves a conditional eventual consensus.

Protocol $\mathcal{F}$ solves Byzantine Finality Gadget Problem if for every block production protocol P, there exist a voting rule A, and the following properties are met:

1. **Safety.** All honest voters finalize the same block at each block number.
2. **Liveness.** If the system $(\mathcal{F}, P, A)$ achieves a conditional eventual consensus, then all honest voters keep finalizing blocks.
3. **Validity.** If an honest voter finalizes some block B, then that block was seen in the best chain observed by some honest voter containing some previously finalised ancestor of B.

In the Analysis section, we show that RANDPA holds these properties.

Additionally, let's recall the definition of supermajority. If a set of voters S for block B has a size of at least $2f + 1$, then S has supermajority over block B.

3 RANDPA Protocol

The RANDPA protocol works in rounds. In each round voters achieve consensus about finalizing a chain of blocks. They determine the Lowest Common Ancestor (LCA) of the blocks that have supermajority during the round. The ancestor is the last block of the finalized chain. The block, followed by the latest finalized block, is the opening block of the finalized chain. This approach allows to finalize some common prefix chain, even if there are several blockchain forks.

Round number is deterministic and depends on the block height. If a voter understands that they are not in the latest round, they are waiting to be synchronized with the current blockchain state.

As we mentioned before, RANDPA is based on the pBFT algorithm. Unlike pBFT, Proposal or Pre-Prepare stages are implicitly expressed as block creation. Each round has two explicit stages - Prevote and Precommit- that correspond with the pBFT Prepare and Commit stages. Every Prevote and Precommit have a corresponding round number. If a voter has two or more votes at any stage of the round, then other voters accept the vote that they received first.

Table 1: Notations

N_r	round number
H_c	height of the current head block
P	primary's id for current round
V	set of correct PREVOTEs for current round
C	set of correct PRECOMMITs for current round
t	round time (estimated in blocks)
H_l	height of the latest finalized block

Remark. If the finality gadget doesn't have time to finalize blocks, we can increase round time in blocks t. The condition $(H_c - 1) = 1 \mod t$ in the second point of the protocol may change depending on the parameter t. You need to choose the number of blocks the Prevote stage takes and substitute 1 for this value.

Protocol 1 Finality Gadget RANDPA

0. Initialization. Let $N_r = 0$. Value H_c depends on the blockchain state. In this article, we consider $t = 2$. H_l is equal to the first block followed by the genesis block.
   Each honest voter v does the following:

1. If $\lfloor (H_c - 1)/t \rfloor > N_r$, then voter v starts a new round. Each voter has their own opinion about current round number. A new round number is $N_r = \lfloor (H_c - 1)/t \rfloor$. Primary for the round N_r becomes the voter P who produced a block at the height $\lfloor (H_c - 1)/t \rfloor \cdot t$.

2. Voter v sends a PREVOTE for chain of blocks $(H_l, H_p]$, where H_p is the latest block produced by primary P. If the chain is empty, then voter v does not participate in the round.

$$\text{PREVOTE} = (N_r, id_{H_l+1}, ..., id_{H_p})$$

3. After voting in Prevote, voter v sends a PRECOMMIT for block $B = LCA(V)$, if $(H_c - 1) = 1 \bmod t$, and B has supermajority for V.

$$\text{PRECOMMIT} = (N_r, id_B)$$

4. If C has supermajority for some block A (not necessary over the block B mentioned in point 3), then the voter broadcasts a chain $(H_l, A]$ with proof. The Proof contains all PREVOTES and all PRECOMMITS for the block A.

   The result is the finalized chain $(H_l, A]$.

   Otherwise, if the voter sees that $\lfloor (H_c - 1)/t \rfloor > N_r$ and they cannot finalize anything, then they just go to the next round.

Differences between GRANDPA and RANDPA

1. In RANDPA, round numbers depend on block heights. It gives us the following:

   1.1 We set a new head block in the blockchain as a message from primary at the Proposal stage, and a creator of this block - as primary. Firstly, now we can remove the phase of choosing primary. Secondly, it allows us to skip broadcasting messages from primary at the Proposal stage. As a result, finalization time is reduced.

   1.2 Consequently, stages Prevote and Precommit depend on block heights, too. Before moving to the new stage, a voter waits for one block. Therefore, the waiting time only depends on the block time.

   1.3 As in GRANDPA, all RANDPA voters have their idea about what the current round number is. However, since round number is connected to the block height, most voters receiving timely updates from the blockchain will be in the same round.

2. GRANDPA uses the GHOST rule for choosing the best fork. We assume that minority-based blockchains do not require this rule because the fewer network participants there

are, the less they compete for block generation. Consequently, forks are less frequent, and RANDPA follows the longest chain rule.

3. In GRANDPA, estimation was implemented for optimizing the PREVOTE's size. In RANDPA, finalization is faster and the amount of participants is lower. So, the chains sent at the Prevote phase are shorter and do not need this optimization.

4. We don't wait until our result chain includes everything that could be finalized in the round. The round is complete even if nothing is finalized.

3.1 Liveness Challenges

RANDPA's optimizations affect the protocol liveness. Due to the fact that participants try to finalize new blocks as soon as possible, they do not wait until anything is finalized. If a voter sees that they have to start a new round, they do that even if the previous one wasn't complete. Such behavior leads to the following situation: if for some reason (e.g. network synchronization issues) a non-finalized chain is too long, than RANDPA cannot finalize it.

To avoid this, we introduce the Support Finality Mechanism (SFM). SFM allows to finalize arbitrary long blockchains. If voters have not finalized anything during several rounds, they switch to SFM and try to finalize the chain in this mode. We present three different options for SFM below.

1. RANDPA with an increased round time t.

2. Another byzantine finality gadget. The gadget may demonstrate slower finalization time but finalize arbitrary long chains.

3. Internal blockchain finality mechanism. Some blockchains already have a finality mechanism. As the finality mechanism in such chains may not be fast enough, RANDPA becomes a relevant solution. The internal finality mechanism will be operating as SFM. For instance, the EOS blockchain has an internal finality mechanism for finalizing longer chains.

Switching between the RANDPA and SFM-mode depends on the chosen SFM. In the first two cases, it is required to run the pBFT algorithm to achieve consensus on block height for switching to SFM and SFM working time. The third option depends on the particular internal finality mechanism. In EOS, the internal finality mechanism may work in parallel with RANDPA. In this case, RANDPA uses the EOS finality only if voters cannot finalize the chain themselves. In other cases, voters do not depend on the EOS finality mechanism.

4 Security

Work In Progress...

5 Implementation

Work in Progress...

References

[AGM18] Ittai Abraham, Guy Gueta, and Dahlia Malkhi. "Hot-stuff the linear, optimal-resilience, one-message BFT devil". In: *CoRR, abs/1803.05069* (2018).

[BG17] Vitalik Buterin and Virgil Griffith. "Casper the friendly finality gadget". In: *arXiv preprint arXiv:1710.09437* (2017).

[CL+99] Miguel Castro, Barbara Liskov, et al. "Practical Byzantine fault tolerance". In: *OSDI*. Vol. 99. 1999. 1999, pp. 173–186.

[Eos] *EOS.IO Technical White Paper v2*. `https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md`.

[Eth] *Ethereum 2.0 Specifications*. `https://github.com/ethereum/eth2.0-specs`.

[Gra] *GRANDPA*. `https://github.com/paritytech/finality-grandpa`.

[Kia+17] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. "Ouroboros: A provably secure proof-of-stake blockchain protocol". In: *Annual International Cryptology Conference*. Springer. 2017, pp. 357–388.

[Mag+19] Bernardo Magri, Christian Matt, Jesper Buus Nielsen, and Daniel Tschudi. "Afgjort–A Semi-Synchronous Finality Layer for Blockchains". In: (2019).

[Nak+08] Satoshi Nakamoto et al. "Bitcoin: A peer-to-peer electronic cash system". In: (2008).

[NJH19] Ryuya Nakamura, Takayuki Jimba, and Dominik Harz. "Refinement and Verification of CBC Casper". In: *networks* 2 (2019), p. 4.

[Xia+19] Yang Xiao, Ning Zhang, Wenjing Lou, and Y Thomas Hou. "A Survey of Distributed Consensus Protocols for Blockchain Networks". In: *arXiv preprint arXiv:1904.04098* (2019).

[Zam] Vlad Zamfir. "Casper the friendly ghost: A correct by construction blockchain consensus protocol". In: ().

[Zam+18] Vlad Zamfir, Nate Rush, Aditya Asgaonkar, and Georgios Piliouras. *Introducing the "Minimal CBC Casper" Family of Consensus Protocols*. `https://github.com/cbc-casper/cbc-casper-paper/blob/master/cbc-casper-paper-draft.pdf`. 2018.