Brief Announcement: Asymmetric Distributed Trust

Christian Cachin
University of Bern, Switzerland
https://crypto.unibe.ch/cc/
cachin@inf.unibe.ch

Björn Tackmann
DFINITY, Zurich, Switzerland
bjoern.tackmann@alumni.ethz.ch

Abstract

Quorum systems are a key abstraction in distributed fault-tolerant computing for capturing trust assumptions. They can be found at the core of many algorithms for implementing reliable broadcasts, shared memory, consensus and other problems. This paper introduces *asymmetric Byzantine quorum systems* that model subjective trust. Every process is free to choose which combinations of other processes it trusts and which ones it considers faulty. Asymmetric quorum systems strictly generalize standard Byzantine quorum systems, which have only one global trust assumption for all processes. This work also presents protocols that implement abstractions of shared memory and broadcast primitives with processes prone to Byzantine faults and asymmetric trust. The model and protocols pave the way for realizing more elaborate algorithms with asymmetric trust.

2012 ACM Subject Classification Theory of computation → Cryptographic protocols; Software and its engineering → Distributed systems organizing principles

Keywords and phrases Quorums, consensus, distributed trust, blockchains, cryptocurrencies

Digital Object Identifier 10.4230/LIPIcs.DISC.2019.39

Related Version A full version of the paper is available athttp://arxiv.org/abs/1906.09314.

Acknowledgements This work has been supported in part by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 780477 PRIViLEDGE.

1 Extended Abstract

Byzantine quorum systems [4] are a fundamental primitive for building resilient distributed systems from untrusted components. Given a set of nodes, a quorum system captures a trust assumption on the nodes in terms of potentially malicious protocol participants and colluding groups of nodes. Quorum systems are at the core of many distributed programming abstractions.

Traditionally, trust in a Byzantine quorum system for a set of processes $\mathcal{P}$ has been *symmetric*. In other words, a global assumption specifies which processes may fail, such as the simple and prominent *threshold quorum* assumption, in which any subset of $\mathcal{P}$ of a given maximum size may collude and act against the protocol. The most basic threshold Byzantine quorum system, for example, allows all subsets of up to $f < n/3$ processes to fail. Some classic works also model arbitrary, non-threshold symmetric quorum systems [4, 3].

However, trust is inherently subjective. *De gustibus non est disputandum.* Estimating which processes will function correctly and which ones will misbehave may depend on personal taste. A myriad of local choices influences one process' trust in others, especially because there are so many forms of "malicious" behavior. Some processes might not even be aware of all others, yet a process should not depend on unknown third parties in a distributed collaboration. How can one model asymmetric trust in distributed protocols? Can traditional Byzantine quorum systems be extended to subjective failure assumptions? How do the standard protocols generalize to this model?

In this paper, we answer these questions and introduce models and protocols for asymmetric distributed trust. We formalize *asymmetric quorum systems* for asynchronous protocols, in which every process can make its own assumptions about Byzantine faults of others. We introduce several protocols with asymmetric trust that strictly generalize the existing algorithms, which require common trust.

Interest in consensus protocols based on Byzantine quorum systems has surged recently because of their application to permissioned blockchain networks [2]. A middle ground between permissionless blockchains based on Proof-of-Work protocols and permissioned ones has been introduced by the blockchain networks of Ripple (`https://ripple.com`) and Stellar (`https://stellar.org`). Their stated model for achieving network-level consensus uses subjective trust in the sense that each process declares a local list of processes that it "trusts" in the protocol.

Consensus in the *Ripple* blockchain is executed by its validator nodes. Each validator declares a *Unique Node List (UNL)*, which is a "list of transaction validators a given participant believes will not conspire to defraud them;" but on the other hand, "Ripple provides a default and recommended list which we expand based on watching the history of validators operated by Ripple and third parties." Many questions have therefore been raised about the kind of decentralization offered by the Ripple protocol. This debate has not yet been resolved. *Stellar* was created as an evolution of Ripple that shares much of the same design philosophy. The Stellar consensus protocol [5] introduces *federated Byzantine quorum systems (FBQS)*; these bear superficial resemblance with our asymmetric quorum systems but differ technically. Stellar's consensus protocol uses *quorum slices*, which are "the subset of a quorum that can convince one particular node of agreement." In an FBQS, "each node chooses its own quorum slices" and "the system-wide quorums result from these decisions by individual nodes". However, standard Byzantine quorum systems and FBQS are *not* comparable because (1) an FBQS when instantiated with the same trust assumption for all processes does not reduce to a symmetric quorum system and (2) existing protocols do not generalize to FBQS.

Understanding how such ideas of subjective trust, as manifested in the Ripple and Stellar blockchains, relate to traditional quorum systems is the main motivation for this work. Our protocols for asymmetric trust generalize the well-known, classic algorithms in the literature and therefore look superficially similar. They are much more powerful, however.

The contributions as detailed in the full paper [1] are as follows:

- We introduce asymmetric Byzantine quorum systems formally as an extension of standard Byzantine quorum systems and discuss some of their properties.
- We show two implementations of a shared register, with single-writer, multi-reader regular semantics, using asymmetric Byzantine quorum systems.
- We examine broadcast primitives in the Byzantine model with asymmetric trust. In particular, we define and implement Byzantine consistent and reliable broadcast protocols. The latter primitive is related to a "federated voting" protocol used by Stellar consensus [5].

Asymmetric quorum systems

Consider a system of n *processes* $\mathcal{P} = \{p_1, \ldots, p_n\}$ that communicate with each other. *Byzantine quorum systems* have been introduced by Malkhi and Reiter [4] with respect to a *fail-prone system* $\mathcal{F} \subseteq 2^{\mathcal{P}}$, a collection of subsets of $\mathcal{P}$, none of which is contained in another, such that some $F \in \mathcal{F}$ with $F \subseteq \mathcal{P}$ is called a *fail-prone set* and contains all processes that may at most fail together in some execution [4]. A fail-prone system captures an assumption on the possible failure patterns that may occur. We let $\mathcal{A}^* = \{A' | A' \subseteq A, A \in \mathcal{A}\}$.

▶ **Definition 1.** *A* Byzantine quorum system *for $\mathcal{F}$ is a collection of sets of processes $\mathcal{Q} \subseteq 2^{\mathcal{P}}$, where each $Q \in \mathcal{Q}$ is called a* quorum, *such that the following properties hold:*

Consistency: *The intersection of any two quorums contains at least one process that is not faulty.*

Availability: *For any set of processes that may fail together, there exists a disjoint quorum in $\mathcal{Q}$.*

This is also known as a *Byzantine dissemination quorum system* [4]. The Q^3-condition [4, 3] generalizes the assumption that $n > 3f$ are needed to tolerate f faulty ones in Byzantine protocols. A fail-prone system $\mathcal{F}$ satisfies the Q^3*-condition*, abbreviated as $Q^3(\mathcal{F})$, whenever it holds $\forall F_1, F_2, F_3 \in \mathcal{F} : \mathcal{P} \not\subseteq F_1 \cup F_2 \cup F_3$. It is well-known [4] that a Byzantine quorum system for $\mathcal{F}$ exists if and only if $Q^3(\mathcal{F})$.

With asymmetric trust, every process is free to make its own trust assumption and to express this with a fail-prone system. Hence, an *asymmetric fail-prone system* $\mathbb{F} = [\mathcal{F}_1, \ldots, \mathcal{F}_n]$ consists of an array of fail-prone systems, where $\mathcal{F}_i$ denotes the trust assumption of p_i.

▶ **Definition 2.** *An* asymmetric Byzantine quorum system *for $\mathbb{F}$ is an array of collections of sets $\mathbb{Q} = [\mathcal{Q}_1, \ldots, \mathcal{Q}_n]$, where $\mathcal{Q}_i \subseteq 2^{\mathcal{P}}$ for $i \in [1, n]$. The set $\mathcal{Q}_i \subseteq 2^{\mathcal{P}}$ is called the* quorum system of p_i *and any set $Q_i \in \mathcal{Q}_i$ is called a* quorum (set) for p_i. *It satisfies:*

Consistency: *The intersection of two quorums for any two processes contains at least one process for which both processes assume that it is not faulty, i.e., $\forall i, j \in [1, n], \forall Q_i \in \mathcal{Q}_i, \forall Q_j \in \mathcal{Q}_j, \forall F_{ij} \in {\mathcal{F}_i}^* \cap {\mathcal{F}_j}^* : Q_i \cap Q_j \not\subseteq F_{ij}$.*

Availability: *For any process p_i and any set of processes that may fail together according to p_i, there exists a disjoint quorum for p_i in $\mathcal{Q}_i$, i.e., $\forall i \in [1, n], \forall F_i \in \mathcal{F}_i : \exists Q_i \in \mathcal{Q}_i : F_i \cap Q_i = \emptyset$.*

The existence of asymmetric quorum systems can be characterized with a property that generalizes the Q^3-condition for the underlying asymmetric fail-prone systems as follows. Namely, an asymmetric fail-prone system $\mathbb{F}$ satisfies the B^3*-condition*, abbreviated as $B^3(\mathbb{F})$, whenever it holds that $\forall i, j \in [1, n], \forall F_i \in \mathcal{F}_i, \forall F_j \in \mathcal{F}_j, \forall F_{ij} \in {\mathcal{F}_i}^* \cap {\mathcal{F}_j}^* : \mathcal{P} \not\subseteq F_i \cup F_j \cup F_{ij}$.

▶ **Theorem 3.** *An asymmetric fail-prone system $\mathbb{F}$ satisfies $B^3(\mathbb{F})$ if and only if there exists an asymmetric quorum system for $\mathbb{F}$.*

References

1 Christian Cachin and Björn Tackmann. Asymmetric Distributed Trust. e-print, arXiv:1906.09314 [cs.DC], 2019. `arXiv:1906.09314`.

2 Christian Cachin and Marko Vukolic. Blockchain Consensus Protocols in the Wild. In Andréa W. Richa, editor, *Proc. 31st Intl. Symposium on Distributed Computing (DISC 2017)*, volume 91 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:16, 2017. `doi:10.4230/LIPIcs.DISC.2017.1`.

3 Martin Hirt and Ueli Maurer. Player Simulation and General Adversary Structures in Perfect Multi-Party Computation. *Journal of Cryptology*, 13(1):31–60, 2000.

4 Dahlia Malkhi and Michael K. Reiter. Byzantine quorum systems. *Distributed Computing*, 11(4):203–213, 1998.

5 David Mazières. The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus. Stellar, available online, `https://www.stellar.org/papers/stellar-consensus-protocol.pdf`, 2016.