Probabilistic Polynomial Time *is* a healer: Consensus with Mobile-Omission Faults

Julian Loss[1], Kartik Nayak[2], and Kecheng Shi[3,4]

[1] Ruhr University Bochum
[2] Duke University
[3] CISPA Helmholtz Center for Information Security
[4] Saarland University

Abstract. We study synchronous Byzantine agreement (BA) with t adaptive Byzantine corruptions and mobile, arbitrarily overlapping send and receive omission sets of per-round sizes s and r. The adversary can selectively drop outgoing (respectively, incoming) messages for send omissions (respectively, receive omissions). Mobile means the adversary can choose a set of fresh send/receive omissions for each round. A well-known lower bound of Santoro and Widmayer (STACS 1989) establishes that time, by itself, is not a healer: any c-round agreement protocol with even a single mobile fault must incur an error of $1/[2(2n)^c]$. Yet, while this rules out deterministic protocols, to date, no *randomized* protocol (even with higher probability of error) is known. This is all the more true when considering a model in which mobile omission faults are mixed with t (permanent) Byzantine faults. Assuming $2t + s + r < n$, we provide the first protocols for and clean formal models for the mixed model with mobile omission faults. This matches the resilience of existing works in this area which consider non-mobile omission faults. We then build a player-replaceable BA protocol with subquadratic communication complexity, under the stronger assumption that $2t + s + r \leq (1 - \varepsilon)n$.

1 Introduction

Byzantine agreement (BA, a.k.a. Consensus) is a basic building block for distributed computing and secure multiparty computation (MPC). In a nutshell, BA is the problem of n parties agreeing on a common and non-trivial output by running a distributed protocol Π. The crucial difficulty is for Π to be able to achieve this goal for all parties who follow the code of Π while being resilient against up to t faults who may attempt to obstruct Π. While the strongest type of fault is the eponymous *Byzantine fault* which can deviate from Π arbitrarily, the number of such faults that can be tolerated is sharply limited to $\lfloor \frac{n-1}{3} \rfloor$ and $\lfloor \frac{n-1}{2} \rfloor$ in the unauthenticated and authenticated setting, respectively. This has motivated a long line of study into weaker types of faults, which, to some degree, follow the instructions of Π, but still exhibit some kind of deviant behaviour. One important such type of fault is *omission faults*, which follow Π honestly, but whose incoming or outgoing messages may be selectively dropped by the

network. Compared to Byzantine faults, it is possible for a protocol to tolerate any number $r < n$ of receive faults (parties whose incoming messages can be lost) and any number $s < n$ of send faults (parties whose outgoing messages can be lost), given that $r + s < n$. In addition, one can make meaningful guarantees on the outputs of omission faulty parties, requiring that an omission faulty party either output the correct value or indicate that no output could be reached.

Omission faults are motivated by common real world occurrences such as transient congestion, buffer pressure, routing changes, and denial of service, all of which can cause message drops to affect different senders and receivers over time. While omission faults have been studied since the 80s, their real world importance has led to a renewed surge in attention to the *mixed fault model* where both Byzantine and omission faults may occur [31,30,20,28].[5]

An intriguing aspect of omission faults is that they are most faithfully modelled as *temporary* rather than *permanent*. Clearly, it is overly pessimistic to treat a party who drops a single message due to an unstable network connection as permanently omission-faulty. A much more realistic approach to modelling omission would therefore be to treat omission faults as *mobile*, i.e., so as to have the ability to jump from one party to the next in every round of the protocol. Unfortunately, no existing protocol for omission faults (mixed or pure) is able to withstand this type of mobile omission fault. This is not a coincidence: the early work of Santoro and Widmayer [40] (later refined by Abraham and Stern [4]) already observes that any consensus protocol running for at most c rounds in a model with even weaker, but round-by-round mobile crash faults must have an error probability of at least $1/[2(2n)^c]$, even when cryptography is permitted. While this rules out deterministic solutions, it leaves open the exciting possibility of a randomized protocol with security against t Byzantine faults and r and s *mobile* receive and send omissions such that $2t + s + r < n$.

In this work, we settle this decades-old question once and for all by giving the first protocols in the mixed fault model with fully mobile omission faults. Concretely, we give the following contribution and constructions:

- We begin by giving a clean and rigorous formalization of the mobile omission fault model. We also give the first non-trivial formalization of Byzantine agreement in this regime, which takes into account the ability of a party to self-detect as receive faulty and choose to forgo outputting. Our non-triviality property is a refinement of Loss' and Stern's [31] *No-Living-Undead* (NLU) property which we refer to as *strong NLU*. Informally, it requires that over the course of the protocol, at most r parties ever self-detect as receive faulty and forgo outputting. We stress that this is a critical requirement in the context of mobile (receive) omission faults. Note that otherwise, it would be easy for the mobile adversary to corrupt every party as receive omission once and always cause all parties to output the default value indicating that no output could be reached.

[5] In this model, it is possible to tolerate simultaneously t Byzantine faults and s and r send and receive faults, respectively, provided that $3t + s + r < n$ (unauthenticated) and $2t + s + r < n$.

– Under $2t + s + r < n$, we construct a BA protocol with mobile-omission faults that tolerates arbitrary overlap between the per-round send and receive omission sets.
– Under $2t+s+r \leq (1-\varepsilon)n$, we design a player replaceable BA protocol with mobile-omissions achieving subquadratic communication complexity. As a special case, this also gives the first subquadratic BA under adaptive (but non-mobile) omission faults with the same resilience and arbitrary overlap.

Our protocols assume VRFs, forward-secure signatures, and the ability for a party to erase its state. The all-party construction and proofs appear in Section 4; the player-replacement construction and analysis are in Appendix A.

1.1 Related Work

The study of omission faults has a long history: Perry and Toueg [38] first studied synchronous agreement when faulty processes may omit sends or receives. Garay and Perry [24] subsequently gave the tight information-theoretic condition $n > 3t_b + t_c$ for a mixture of t_b Byzantine and t_c crash faults. Parvedy and Raynal [36] motivated the omission fault model through failures of communication buffers. Zikas, Hauser, and Maurer [47] separated full-omissions with send and receive omission corruption, they proved perfectly secure BA and MPC under $3t + s + r < n$. Very recently, the omission fault model has received much more attention, since it characterizes realistic network failures and protocols in the model may tolerate more benign faults than Byzantine faults. In the cryptographic setting, Eldefrawy, Loss, and Terner [18] gave a BA protocol under $2t + 2s + r < n$, and proved the optimal condition $2t + s + r < n$ for the restricted *spotty* send omission corruption model, in which all or none of one sender's messages are lost in a round. Loss and Stern [31] achieved the resilience threshold against a rushing and strongly adaptive adversary. Loss, Shi, and Stern [30] subsequently improved over [31] to tolerate overlapping faults. Feng et al. [20] achieved expected $O(n^2\kappa)$-bit communication and expected $O(1)$ rounds in the overlapping, rushing, strongly adaptive setting under the same resilience threshold. In a concurrent work, Kamp et al. [28] obtained expected $O(n^2)$ communication, expected $O(1)$ rounds under $2t+s+r < n$ and $t \geq \varepsilon n$ for a constant $\varepsilon > 0$, for non-overlapping omission faults against an adaptive adversary without "after-the-fact removal". Building on the multi-valued BA protocol of Feng et al. [20], Zheng et al. [46] construct an atomic broadcast protocol in the mixed-fault model under the same resilience condition $2t + s + r < n$.

In the information-theoretic setting, Brazitikos and Zikas [7] gave tight BA and MPC feasibility characterizations in the general adversary structures with active and full-omission corruption. Independently of these resilience bounds, Civit et al. [13] proved an $\Omega(t^2)$ worst-case *message* lower bound for deterministic nontrivial BA problems, where its lower-bound execution used only omission/silence behavior by faulty processes.

Mobile Byzantine Agreement and Proactive Security. Early work by Reischuk [39] and Buhrman, Garay, and Hoepman [8] studied agreement with moving faults

under their respective stability assumptions, with the latter obtaining the tight resilience condition $n > 3t$. For faults that may move between consecutive rounds, Garay [22] considered processes that recognize their first round after being released from adversarial control and can refrain from propagating corrupted state, obtaining an algorithm for $n > 6t$. Banu et al. [5] improved the sufficient condition in this model to $n > 4t$. Garay's algorithm assumes that some process remains uncorrupted throughout the execution, whereas Banu et al.'s algorithm requires one process to remain uncorrupted for at least $3n$ rounds. Sasaki et al. [43] considered a model in which a recovered process does not detect its prior infection and may send messages prepared while it was still under adversarial control. Their resilience threshold is $n > 6t$. They proved impossibility when $n \leq 6t$ or the vertex connectivity is at most $4t$, and gave matching algorithms for certain network families under permanently non-faulty-process assumptions. Bonnet et al. [6] studied a recovery model in which process identities and code are protected, while a process released from adversarial control resumes that code with potentially corrupted state.

A related line of work studies secure computation and cryptographic primitives under mobile corruptions. Ostrovsky and Yung [35] introduced a mobile-virus model and techniques for secure computation with recurring recovery and refresh, a perspective revisited in Yung's survey [45]. Canetti and Herzberg [10] constructed distributed pseudorandomness that remains unpredictable after earlier infections, using randomness supplied at initialization. Herzberg et al. [27] developed proactive secret sharing, in which parties renew their shares while preserving the underlying secret, and detect and recover damaged shares. Subsequent work developed these techniques for cryptographic primitives and different communication models. Frankel et al. [21] constructed proactive public-key crypto systems. Cachin et al. [9] gave asynchronous verifiable secret sharing and a proactive refresh model organized around local phases.

Dynamic Link Failures and Message Adversaries. Foundational work by Santoro and Widmayer [40,41,42] studied dynamic, transient transmission failures. They established impossibility bounds for deterministic synchronous agreement. On a complete network, their argument also rules out deterministic consensus when an adversary may selectively omit outgoing messages of at most one party per round, whose identity may change between rounds, using at most $n - 1$ message omissions per round. Schmid et al. [44] studied complementary local failure bounds, distinguishing between send and receive omissions on a per-party basis (note that they still bound the message loss through *link failures* rather than party failures). To abstract such transient network behavior, Charron-Bost and Schiper [11] introduced the Heard-Of (HO) model, which reasons about round-by-round communication topologies without permanently classifying parties as faulty. Approaching the problem via arbitrary mobile link-omission schemes, Godard and Peters [25] proved that deterministic consensus is equivalent to reliable broadcast from some fixed source for convex families of mobile omission patterns. Coulouma, Godard, and Peters [15] subsequently gave a complete characterization of deterministic consensus solvability for *oblivious message ad-*

versaries. By analyzing the local indistinguishability of these communication events, they provided the first complete necessary and sufficient conditions for consensus feasibility under arbitrary, memory-less dynamicity.

Consensus with mobile crashes. In concurrent work, Abraham et al. [1] study state machine replication under partial synchrony with permanent adaptive Byzantine corruptions and mobile crashes. A crashed replica keeps its state. Upon recovery, it resumes interrupted sends and processes buffered messages, allowing it to catch up on views it missed. Our mobile omission model permits a strictly wider range of message losses: an affected party can continue executing while selected incoming or outgoing messages are permanently lost.

Dynamic Participation and Transient Faults. The Sleepy Model, introduced by Pass and Shi [37], provides a framework where parties can arbitrarily transition between active and offline states under adversarial control. Under its communication and setup assumptions, security and liveness are provably maintained when alert (honest and awake) parties outnumber corrupt parties at any given time step, where all corrupt parties are counted as awake. Subsequent works have further investigated this model [17,34,32]. Guo, Pass, and Shi [26] studied a weakly synchronous model for partition-tolerant agreement. While the sleepy model addresses parties that temporarily halt computation, the Mobile Sluggish Fault Model [29] captures unbounded message delays. In this model, sluggish parties continue local execution, and the protocols assume that at every time, a strict majority of parties are honest and prompt. In the original sleepy model, a waking party receives the pending messages it would have received while awake [37]. Momose and Ren [34] also allow messages sent during sleep to be lost and give rejoining parties an explicit recovery period. In the mobile sluggish model, a message sent by a prompt party is delivered once the receiver is prompt and the delivery bound has elapsed; in our model, even a receiver that is reliable in the very next round may never receive the omitted message.

Adaptive and player-replaceable BA. Micali and Vaikuntanathan [33] constructed player-replaceable BA, while Algorand [12] uses VRF-based election to replace speakers at each step. Abraham et al. [2] showed that, for $f = \Theta(n)$ Byzantine corruptions, adaptively secure BA with negligible error requires $\Omega(n^2)$ expected honest-party messages if already-sent messages can be removed after corruption. They also give subquadratic protocols without such "after-the-fact removal". Later work studies player-replaceable BA over gossip and together with early stopping [14,19].

2 Technical Overview

Basic Byzantine agreement Framework. The core protocol of Abraham et al. [3] runs in iterations. In each iteration, a leader is elected, followed by Status, Propose, Commit, and Notify. The leader collects accepted certificates in Status and

uses their ranks to choose its proposal. In Commit, sufficiently many matching commit requests form a certificate, and committing also requires that no leader equivocation is detected. Parties that commit disseminate their certificates in Notify. A party terminates after collecting and forwarding the required number of matching notify headers from distinct parties, where these headers need not come from the same iteration.

2.1 Weak Graded Multicast with Mobile Omissions Π_{WGMMO}

To extend this approach to omission faults, we must take parties that follow the protocol but whose messages are not delivered into consideration, especially for send omissions (as observed in [18]). This is because they may deliver messages inconsistently under control of the adversary. In this section, we introduce the idea behind Π_{WGMMO}, which is the core building block of our BA protocol.

Send-omission detection. Prior mixed-fault protocols may allow the adversary to choose omission-corrupted parties adaptively, but each affected party counts against the corresponding execution-wide budget. In this setting, the Zombies and Ghosts paradigm [31,30,20] uses self-detection to handle omission faults. In our mobile-omission model, the adversary may instead affect fresh, arbitrarily overlapping sets S_u and R_u in every round u, subject to $|S_u| \leq s$ and $|R_u| \leq r$. Since every party may experience omissions during the execution, permanent self-exclusion can eventually silence every party. The No Living Undead (NLU) property, which restricts flags to parties that have experienced receive omissions, likewise does not by itself bound the number of flags by r under mobility. We therefore require *Strong NLU*, which additionally bounds the number of flagged non-Byzantine parties by r at the end of the protocol.

We call a party *send-reliable* (respectively, *receive-reliable*) in a round if it is neither Byzantine-corrupted nor send (respectively, receive) omission corrupted in that round. Note even if a message reaches a send-reliable party in one round, that party may be unable to forward it in a later round due to Byzantine corruption or send omission corruption.

We introduce our novel three-round primitive Π_{WGMMO} (Weak Graded Multicast with Mobile Omissions) to resolve this difficulty. The high-level structure of Π_{WGMMO} is in Fig. 1. Each invocation Π_{WGMMO} has one designated sender P_i with its input m_i. For a sender that multicasts in the Send round, our intuition is to use non-Acknowledged (NACK) parties to detect send omissions publicly. On the one hand, if the sender is send-reliable in the delivery round, at most r non-Byzantine parties fail to receive its message. Even allowing t Byzantine parties to report NACK maliciously, there can be at most $t + r$ valid reports of NACK from distinct parties. Thus, $t + r + 1$ such reports imply send omissions. In fact, prior works [31,30,20] used this argument, where they send those reports back to the sender to allow it to detect its send omission locally. Note a send omission sender may nevertheless leave fewer than $t + r + 1$ parties without the message, so this test need not detect every send omission.

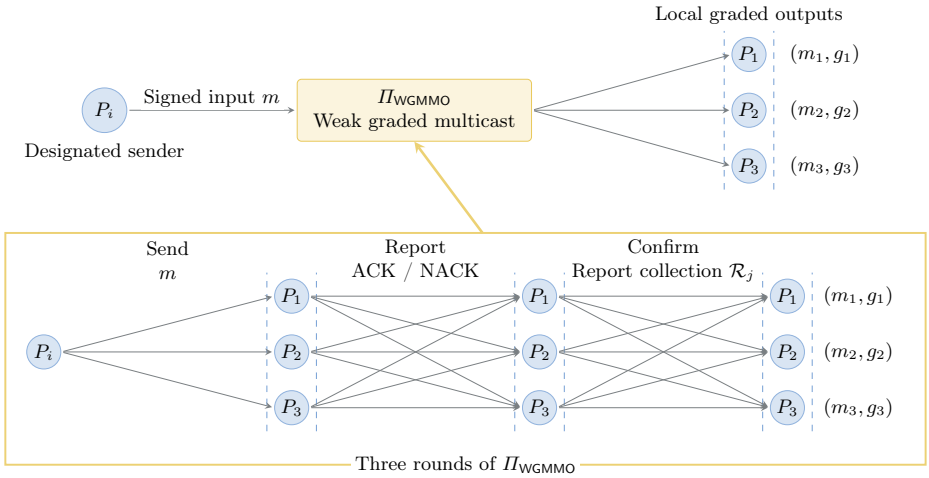


Fig. 1. Structure of Π_{WGMMO}. Arrows show multicast subject to protocol conditions and omissions. Confirm includes received reports, each party determines its output and grade based on its local view.

On the other hand, it suffices to multicast the message when at least $n-(t+r)$ parties have the message. In any later forwarding round, at most t of these holders can be Byzantine and at most s others can suffer send omissions, leaving at least $n-(t+r)-t-s = n-2t-r-s \geq 1$ send-reliable holder under $2t+s+r < n$ in a fixed round. Our protocol establishes such a set of holders through the multicast of signed Acknowledge (ACK) messages containing the original message.

The Π_{WGMMO} primitive proceeds through *Send*, *Report*, and *Confirm*. We call it "weak" since it provides no graded-consistency guarantee when the sender is Byzantine in the Send round.

Graded outputs. Each invocation has one designated sender P_i with input m_i. Each receiver keeps the first valid signed input it obtains, directly or through an ACK. At the end of Confirm, each party P_j outputs $(m, 1)$ if the three conditions below hold for m, $(m, 0)$ if it has received an ACK containing m but these conditions do not all hold, and $(\perp, 0)$ otherwise. For a non-Byzantine sender, the valid signed input is unique.

Π_{WGMMO} guarantees that a non-Byzantine party's grade-one output for m yields an output (m, g) for some $g \in \{0, 1\}$ at every non-Byzantine party that is receive-reliable in the Confirm round, with respect to a non-Byzantine sender.

For validity of the graded weak multicast, if the sender is send-reliable in the Send round, all these parties output its input with grade one. We describe the Π_{WGMMO} protocol in detail in Section 4.1.

Reports and confirmation. After the sender P_i signs and multicasts its message in the Send round, every receiver signs exactly one Report message for P_i: either

an ACK containing the exact signed message it received in the Send round, or a NACK otherwise. In the Confirm round, each party includes all the Report messages it has received in one Confirm message and multicasts it.

At the end of Confirm, each party outputs the valid signed input m it has kept, with grade one exactly when all of the following conditions hold:

1. It receives valid Confirm messages from at least $q_{\text{mob}} = t + r + 1$ distinct parties.
2. It has received an ACK containing the exact message m during the Report round or the Confirm round.
3. None of its processed valid Confirm messages contains NACK Report messages from q_{mob} distinct reporters (q_{mob} distinct NACK reports is thus called a *rejection proof*).

Graded consistency. To show the graded consistency for a non-Byzantine sender, assume a non-Byzantine party P_c outputs $(m, 1)$. If no party that is send-reliable in the Report round sends an ACK for m, every such party must therefore have multicast NACK, and there are at least $n - t - s \geq q_{\text{mob}}$ such parties sending NACK. Every confirmer that is receive-reliable in the Report round therefore forms a *rejection proof*. Among the $q_{\text{mob}} = t + r + 1$ Confirm messages received by P_c, at least one was sent by a non-Byzantine confirmer that is receive-reliable in the Report round. Its Confirm message must contain a *rejection proof*, contradicting condition 3 for its grade-one output. Hence a non-Byzantine party's grade-one output for m implies an ACK for that exact message from some party P_j that is send-reliable in the Report round. This ACK reaches every party in $\mathcal{N} \setminus R_{\text{rep}}$ during the Report round, so at least $n - t - r$ non-Byzantine parties hold it before Confirm. Here, $\mathcal{N}$ excludes every party that is Byzantine-corrupted at any point during the execution. Thus, although the omission sets may move arbitrarily between rounds, at most s of these holders suffer send omissions in the Confirm round, leaving at least $n - t - r - s \geq t + 1$ send-reliable holders. Each such holder multicasts the ACK in its Confirm message, so every party that is receive-reliable in the Confirm round receives the ACK before determining its output. Since the non-Byzantine sender's valid signed input is unique, every such non-Byzantine receiver outputs (m, g) for some $g \in \{0, 1\}$.

2.2 Our Byzantine Agreement

We refer to the signed core of a proposal as its *header*. It binds the iteration, proposer, proposed value, VRF evaluation, and corresponding eligibility proof; the valid proposal additionally carries supporting Status messages. The headers carried in subsequent votes are these proposal headers. We then construct the all-party protocol Π_{BasicBA} using Π_{WGMMO} as follows. The high-level structure is in Fig. 2. After an Initialization round, the protocol runs $L = \lambda n$ iterations, each consisting of Status, Propose, three vote phases, and Certificate dissemination. A party following the protocol labels its vote *emphatic* to support a header when its local view meets the conditions below; otherwise, it labels it *non-emphatic*.

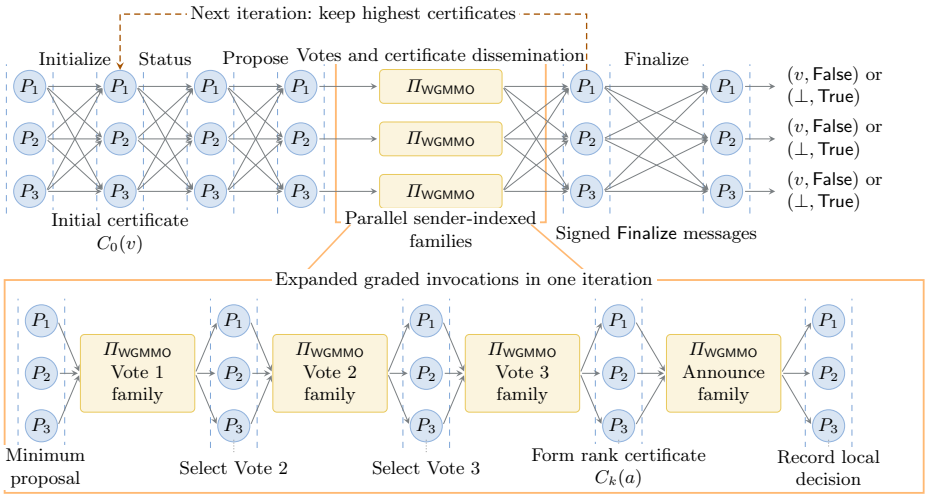


Fig. 2. Structure of Π_{BasicBA}. Crossed arrows represent multicast subject to omissions. Initialization is performed once. The shaded part uses a sender-indexed Π_{WGMMO} call for each Vote and for the certificate dissemination. The dashed return arrow shows that parties keep both certificates (for Status messages) and local decisions across iterations. After all iterations, parties with local decisions multicast Finalize messages, and all parties then determine their BA output.

This sender-assigned label is distinct from each receiver's output grade. The three vote phases let parties form rank certificates from signed emphatic Vote_3 inputs. Parties then announce these certificates through Π_{WGMMO} to record local decisions, and report their stored certificates in later Status messages to restrict subsequent proposals. After L iterations, parties execute one final round and output.

We adapt the three-vote structure and minimum-VRF selection of OmniBA [28] to mobile omissions. The core idea is to force parties to consider lower-evaluation headers, thereby ruling out emphatic votes for higher-evaluation headers. Each vote phase and the certificate-dissemination phase run n parallel three-round Π_{WGMMO} invocations, one per designated sender.

Rank certificates and agreement. Vote 3 is the last of the three vote phases described below. If a party's final outputs from this phase in iteration k contain emphatic Vote_3 inputs for header a with grade one from $t+1$ distinct designated senders, it combines these signed inputs into a rank-k certificate $C_k(a)$.

Parties then announce their rank-k certificates through another parallel Π_{WGMMO} phase, using NIL if they have none. Each party stores the rank certificate in every non-NIL announcement output, regardless of grade. Grade-one announcements for the same header from $t+1$ distinct input senders let an undecided party store a *local decision*. It continues participating until the final round.

If a non-Byzantine party stores such a *local decision*, weak graded consistency of Π_{WGMMO} ensures that a certificate for that header is received by at least $n-t-r$ non-Byzantine parties.

Status and proposals. In each iteration, parties report their highest stored certificate of rank below the current iteration in signed Status messages, using $\perp$ if they have none. A proposer collects q_{mob} valid Status messages from distinct parties and includes them in its signed proposal. The proposed value follows the maximum-rank value rule of Section 4.2; in particular, when the Status messages contain certificates, it is taken from a certificate of maximum rank. After a non-Byzantine party records a local decision in iteration k, every later valid proposal therefore includes a Status message from a non-Byzantine holder of a rank-k certificate, reporting rank at least k. Every later rank certificate is supported by a valid proposal. Induction on ranks then preserves the value across iterations.

Vote 1. Vote 1 records each party's choice among valid received proposals. Write $\tau(a)$ for the VRF evaluation in proposal header a. A party labels its Vote_1 emphatic and inputs it with header a and its proposal exactly when a is the unique header attaining the minimum evaluation among its valid received proposals. With no valid proposal, it inputs NIL. If distinct headers tie at the minimum, it instead inputs SPLIT carrying the two least tied headers under a fixed public order. This records the tie rather than supporting either header; the sender labels NIL and split votes non-emphatic.

Vote 2. Each party applies the minimum-header rule to all valid Vote_1 messages in its final outputs from the Vote 1 phase, received via Π_{WGMMO}, regardless of grade or emphatic support. It inputs the resulting NIL, split, or single-header vote into its Π_{WGMMO} instance for the Vote 2 phase. For a single header a, the sender labels its vote emphatic exactly when its Vote 1 outputs contain emphatic Vote_1 inputs for a with grade one from at least $t+1$ distinct designated senders; otherwise, the sender labels its vote for a non-emphatic. A split either propagates an existing split at the minimum or records a tie between headers in different Vote_1 messages. The sender labels a split non-emphatic, preserving the tie for Vote 3. Let D_1 denote the non-Byzantine parties that are receive-reliable in the Confirm round of the parallel Vote 1 invocations of Π_{WGMMO}. If a non-Byzantine party sends an emphatic Vote_2 for a, weak graded consistency ensures that a already occurs in every party's Vote 1 outputs within D_1. Thus, when these parties choose their own Vote_2, a rules out any different header b with $\tau(b) \geq \tau(a)$, even if the output carrying a has grade zero. In particular, emphatic Vote_2 messages sent by parties in D_1 must agree. Their non-emphatic votes may still differ, and receive omissions may allow parties outside D_1 to send conflicting emphatic votes. Thus we require one more Vote phase.

Vote 3. Each party applies the same minimum-header rule to all valid Vote_2 messages in its final outputs from the Π_{WGMMO} instances in the Vote 2 phase, regardless of grade or emphatic label, handling NIL and SPLIT as in Vote 2.

For a uniquely selected header a, the sender labels its Vote_3 emphatic exactly when its Vote 2 outputs contain both emphatic Vote_2 messages for a with grade one from at least $t+1$ distinct designated senders and valid Vote_2 messages from at least $q_{\mathrm{mob}} = t + r + 1$ distinct designated senders (for arbitrary valid message contents). Otherwise, the sender labels its Vote_3 for a non-emphatic. The latter requirement is the *coverage condition*; it counts valid votes of any grade, including NIL and splits. Since at most $t + r$ parties lie outside D_1, satisfying coverage implies that at least one of the party's Vote 2 outputs is for a valid Vote_2 message signed by a party in D_1, even if it experiences receive omissions in both phases. Hiding all such messages instead leaves insufficient coverage for an emphatic vote. If a non-Byzantine party sends an emphatic Vote_2 for a, this message from D_1 rules out an emphatic Vote_3 for any different header b with $\tau(b) \geq \tau(a)$, even when the message is non-emphatic or has grade zero. Together with the emphatic-support requirement, this prevents any conflicting non-Byzantine emphatic Vote_3 messages and hence conflicting rank certificates in the same iteration (see Theorem 2 below).

Termination and strong NLU. A *good iteration* happens when the minimum eligible VRF evaluation is unique and its proposer is send-reliable in the Propose round and receive-reliable in the Status round of iteration k.

In a *good iteration*, the validity of Π_{WGMMO} gives enough grade-one outputs for emphatic inputs through three Vote phases, so the announcement phase ensures that at least $n - t - r$ non-Byzantine parties have *local decisions*. At least $n - t - r - s \geq t + 1$ of these parties are send-reliable in the final round.

In the final round, every party with a local decision signs and multicasts its value, and receivers accept a value supported by $t+1$ distinct final-round signers. Every party that is receive-reliable in the final round receives these signatures and outputs the same value, while the remaining non-Byzantine parties may output the flagged (empty) output, achieving strong NLU except with failure probability at most $((t + s + r)/n)^{\lambda n}$ over λn iterations.

2.3 Player Replacement

We then construct BA with subquadratic communication in the mobile omission fault model under $2t + s + r \leq (1 - \varepsilon)n$, for any constant $\varepsilon \in (0, 1)$. We replace all-party sending in each round with fresh hidden committees selected using VRFs. Unlike in the all-party Π_{WGMMO} construction, reporters do not know which parties were eligible input senders but failed to reach them. Explicitly reporting ACK/NACK for all n parties would require $\Omega(n)$ bits per Report, already resulting in at least quadratic communication when multicast.

One execution of Π_{PRWGMMO} handles all eligible inputs in parallel, using one fresh shared committee in each of its three rounds (Fig. 3).

Graded outputs. Each eligible input sender P_i has an input m_i. The Π_{PRWGMMO} protocol asks each party P_j to output a set $\mathcal{O}(P_j)$ of pairs $(m_{i,j}, g_{i,j})$, with $g_{i,j} \in \{0, 1\}$. For each input signer P_i, party P_j chooses one valid input $m_{i,j}$

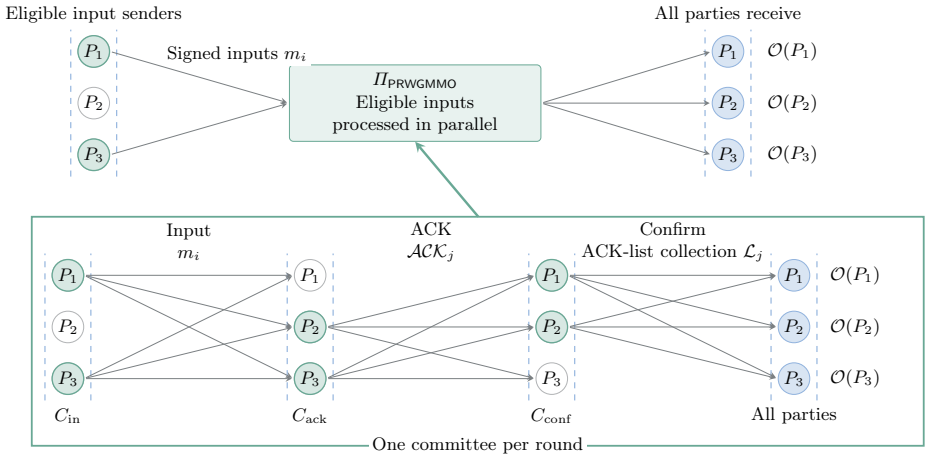


Fig. 3. Structure of Π_{PRWGMMO}. The lower panel expands one parallel invocation: green nodes denote eligible senders, while all parties receive. Repeated columns represent the fresh, possibly overlapping role committees. Confirm forwards received ACK-list collections. Arrows are multicast attempts subject to omissions, and each receiver locally forms $\mathcal{O}(P_j)$ and determines its grades from its local view.

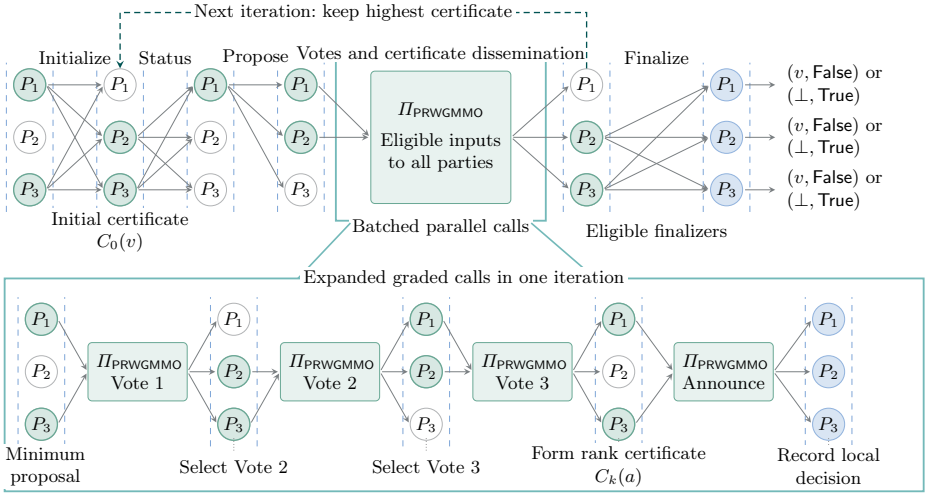


Fig. 4. Structure of Π_{PRBA}. Repeated columns depict the same population; green nodes indicate a fresh selection of eligible parties for each sending role. Initialization is performed once. Each Π_{PRWGMMO} call batches eligible inputs and returns graded outputs to all parties; arrows represent multicast subject to omissions. The dashed return arrow shows that parties keep their highest certificates and local decisions across iterations. After all iterations, eligible finalizers with local decisions multicast Finalize messages to produce the BA output.

from its received Input-round messages or ever received ACK lists. At the end of Confirm, it includes $(m_{i,j}, 1)$ if the two conditions below hold for $m_{i,j}$, and includes $(m_{i,j}, 0)$ otherwise. If no valid input from P_i is available, it includes no pair for P_i.

Reports and confirmation. In the Input round, each eligible input sender signs and multicasts its input. In the ACK round, each eligible reporter signs and multicasts one Report message containing these inputs. This forms its $\mathcal{ACK}$ list. For an exact signed input m, a received $\mathcal{ACK}$ list that does not contain m serves as an implicit NACK for m. A missing list contributes no NACK.

In the Confirm round, each eligible confirmer includes the first valid ACK list it received from each ACK signer in one Confirm message and multicasts it. Each receiver keeps the first valid $\mathcal{ACK}$ list per ACK signer obtained directly or in a valid Confirm message. It also keeps the first valid Confirm message per confirmer.

At the end of Confirm round, a chosen input m has grade one exactly when both of the following conditions hold, using the committee threshold q defined in Appendix A.1:

1. The party has kept valid Confirm messages from at least q distinct eligible confirmers.
2. None of the valid Confirm messages it has kept contains $\mathcal{ACK}$ lists from q distinct eligible ACK signers that all omit m. (We say such $\mathcal{ACK}$ lists *reject* m.)

Graded consistency and validity. Under the committee concentration conditions, a non-Byzantine party's output $(m, 1)$ implies an earlier ACK multicast containing m by a reporter that is send-reliable in the ACK round. These conditions also ensure that there is a confirmer that is receive-reliable in ACK and send-reliable in Confirm. It forwards the $\mathcal{ACK}$ list. For a non-Byzantine input sender, the signed input is unique, so every receive-reliable party in Confirm includes (m, g) in its output set for some $g \in \{0, 1\}$.

For an eligible input sender that is send-reliable in the Input round and sends m, every receive-reliable party in the Confirm round receives m and valid Confirm messages from at least q distinct eligible confirmers, with no Confirm rejecting m. It therefore includes $(m, 1)$ in its output set.

We use the all-party protocol's Status, Propose, three-vote, and announcement structure, using Π_{PRWGMMO} and committee thresholds in place of Π_{WGMMO} and all-party thresholds (Fig. 4). The full construction and analysis are given in Appendix A.

3 Preliminaries

3.1 Network model

We consider n parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ connected by a complete network of authenticated point-to-point channels. Communication is synchronous and proceeds in rounds. Every message sent in a round is delivered by the end of that

round unless the adversary drops it under the rules specified below. The channels do not provide privacy. To *multicast* is to send the same message to all n parties, including the sender.

3.2 Adversary model

Corruption types and budgets. We consider three types of corruption: Byzantine, send omission, and receive omission corruption. Byzantine-corrupted parties may deviate arbitrarily from the protocol. Send and receive omission corruptions allow the adversary to selectively drop outgoing and incoming messages to/from a receive/send omissions party, respectively, subject to the rules below. Parties subject only to omission corruptions continue to follow the protocol. Let $B \subseteq \mathcal{P}$ denote the set of parties that are Byzantine-corrupted at any point during the execution, and let $\mathcal{N} = \mathcal{P} \setminus B$ denote the *non-Byzantine* parties. Parties in $\mathcal{N}$ may still experience send or receive omissions in arbitrarily many rounds. For each round u, the sets B_u, S_u, R_u record the parties subject to Byzantine, send omission, and receive omission corruption, respectively. The bounds $|B_u| \leq t$, $|S_u| \leq s$, and $|R_u| \leq r$ must hold at all times.

Adaptivity and mobility. At the start of round u, B_u is initialized to the final value of B_{u-1}, with $B_0 = \emptyset$, while S_u and R_u are initialized to $\emptyset$. During the round, the adversary may adaptively add parties to these sets based on its current view, subject to the stated budgets and the message-dropping rules below. The sets can only grow within a round. Maintaining an omission corruption across rounds therefore requires selecting the party again in each round. The send and receive omission sets may overlap arbitrarily, and their unions over the execution may exceed the per-round budgets. For a completed round u, B_u, S_u, R_u denote the final values of these sets. In particular, $B_u \subseteq B_v \subseteq B$ whenever $v > u$. A party is *send-reliable* in round u if it is outside $B_u \cup S_u$, where we count a party in S_u only if its send omission takes effect before that action. A party is *receive-reliable* in round u if it is outside $B_u \cup R_u$. A party corrupted in a later round may still be reliable in round u.

Rushing and atomic sends. The adversary is rushing: it may choose Byzantine parties' round-u messages after observing that round's transcripts from non-Byzantine parties, and may arrange for Byzantine messages to arrive before messages from non-Byzantine parties. Before each party begins its round-u atomic send action, the adversary has priority to corrupt parties, subject to the stated budgets. If P_i is not under Byzantine control when its atomic send begins, it signs its outgoing messages, updates its forward-secure signing key, erases the round-u key, and submits the messages to the network. These operations are atomic, with no adversarial interleaving [23]. The party updates its key even if it sends no messages; see Section 3.4. A party already under Byzantine control may deviate arbitrarily. Send omission corruption affecting P_i's round-u messages must take effect before its atomic send begins: a request to drop a message is valid if the send corruption has been made prior to the atomic send, and

in this case, adversary is able to selectively drop this send omission's messages for any later rounds. Similarly, later Byzantine corruption of the sender cannot delete messages already in the network ("no after-the-fact removal" [2]). Receive-omission corruption may take effect at any time during round u and allows the adversary to selectively drop messages still awaiting delivery to the affected receiver. Consequently, if P_i is send-reliable in round u and sends a message to $P_j \notin R_u$ in round u, that message must be delivered to P_j by the end of the round.

3.3 Protocol properties

Definition 1 (Byzantine agreement). *Let Π be a protocol executed by $\mathcal{P} = \{P_1, \ldots, P_n\}$. Each party P_i starts with an input $v_i \in \mathcal{V} \cup \{\perp\}$, where $\mathcal{V}$ is a public value domain and $\perp \notin \mathcal{V}$ denotes a missing input. The initial input assignment is fixed before the protocol starts. Write $U = \{P_i : v_i = \perp\}$; parties in U still participate in the protocol. Each non-Byzantine party $P_i \in \mathcal{N}$ outputs a pair*

$$(x_i, Z_i) \in (\mathcal{V} \times \{\mathsf{False}\}) \cup \{(\perp, \mathsf{True})\}.$$

An output (x_i, False) is unflagged*, and $(\perp, \mathsf{True})$ is* flagged*. The required properties are:*

- ***Validity.*** *If $|U \cap \mathcal{N}| \leq r$ and every party in $\mathcal{N} \setminus U$ starts with the same input $v \in \mathcal{V}$, every unflagged output of a party in $\mathcal{N}$ has value v.*
- ***Agreement.*** *For any $P_i, P_j \in \mathcal{N}$,*

$$Z_i = Z_j = \mathsf{False} \quad \Longrightarrow \quad x_i = x_j.$$

- ***Termination.*** *Every party in $\mathcal{N}$ terminates at the end of the protocol with exactly one output pair in the specified format.*

We say that Π achieves Byzantine agreement (BA) with flagged outputs if it satisfies these properties in the adversary model above.

This validity condition facilitates composition with a larger protocol. A preceding stage may produce a common value v for all but at most r non-Byzantine parties, leaving the remaining parties without an output because of receive omissions. These local outputs can serve directly as BA inputs, with missing outputs represented by $\perp$. Validity then ensures that every unflagged output of a non-Byzantine party has value v, while parties without initial inputs continue to participate.

These properties alone permit every party to flag. The following no-living-undead condition restricts which parties may flag [31,30,28].

Definition 2 (Ordinary NLU). *A protocol Π with outputs as in Definition 1 satisfies* ordinary no-living-undead *if, for every $P_i \in \mathcal{N}$,*

$$Z_i = \mathsf{True} \quad \Longrightarrow \quad P_i \in \bigcup_u R_u.$$

In the mobile-omission model, every party may experience receive omissions during the execution. Ordinary NLU therefore does not limit the number of flags to r. We strengthen it as follows.

Definition 3 (Strong NLU). *A protocol Π satisfies* strong no-living-undead *if it satisfies Ordinary NLU and additionally $|\{P_i \notin B \mid Z_i = \mathsf{True}\}| \leq r$, where r is the per-round receive-omission budget.*

A BA protocol satisfying Strong NLU is called a *strong-NLU BA protocol*.

3.4 Cryptographic Primitives

We analyze the protocols with ideal cryptographic primitives. Throughout, κ is the cryptographic security parameter, and λ is a public parameter controlling the number of iterations. The player-replacement construction also uses committees of expected size κ. A cryptographic word has κ bits. Each signature, VRF output, and VRF proof costs $O(1)$ cryptographic words. We assume $\log(n\lambda) = O(\kappa)$. We count only communication sent by non-Byzantine parties.

Ideal VRF-based eligibility. Following [33,2], we use VRFs for proposer eligibility and ordering in both constructions, and for eligibility in player-replacement roles. All these uses are analyzed with idealized VRFs secure under adaptive party corruptions, including selective exposure of secret keys. Abraham et al. [2] establish such adaptive security under standard cryptographic assumptions; simpler constructions are available in the random-oracle model. For each fresh role label ℓ, let $D_\ell \in (0, 1]$ be a public difficulty parameter fixed before that label is activated. For party P_i, the ideal VRF functionality samples $Y_{i,\ell} \sim \mathrm{Unif}([0, 1))$ and returns the evaluation together with a publicly verifiable proof $\pi_{i,\ell}$ for that exact value. Define $E_{i,\ell} := \mathbf{1}[Y_{i,\ell} < D_\ell]$. A verifier accepts P_i as eligible for role ℓ exactly when $\pi_{i,\ell}$ verifies and the certified evaluation satisfies $Y_{i,\ell} < D_\ell$. Outputs are independent across distinct party–label pairs. For a proposer label, only an eligible party may propose, and eligible proposals are ordered solely by their certified VRF evaluations $Y_{i,\ell}$, with a smaller evaluation ordered first. Distinct headers with the same evaluation are tied and are handled by the minimum-header rule. For a committee label, the selected committee is $C_\ell := \{P_i : E_{i,\ell} = 1\}$, and every member of C_ℓ is eligible to perform the corresponding committee instruction. The all-party construction uses $D_\ell = 1$ for proposer labels, whereas the player-replacement construction uses the difficulties specified in Appendix A.1. For a concrete finite-range VRF, we choose the output length so that the collision probability over the execution is negligible and include it in the computational error.

Secure erasures. We assume that parties following the protocol can securely erase local secret state, so that subsequent Byzantine corruptions do not reveal previously erased information. In particular, each party erases its obsolete signing state as part of the atomic send action specified above.

Idealized forward-secure signatures. Each party P_i has a fixed public verification key and a signing key that is updated each round. During its atomic send in round u, P_i may sign tuples $(\mathsf{sid}, i, u, \ell, m)$, which bind each message to its protocol instance, signer, round, and role. Before releasing the messages, it advances its signing state to round $u+1$ and irreversibly erases the round-u state, even if it sends no messages. Period-u signatures remain publicly verifiable. Ideally, if $P_i \notin B_u$, every valid period-u signature under P_i's verification key was generated by P_i in round u according to the protocol. This guarantee persists after any later corruption: a later signing key cannot be used to create additional signatures for a closed period. A concrete forward-secure signature scheme provides this guarantee except with negligible probability in κ [16].

4 Basic Byzantine Agreement with Mobile-Omission Faults

Under $2t + s + r < n$, we construct an all-party Byzantine agreement protocol Π_{BasicBA} from Π_{WGMMO}. The protocol runs for $L = \lambda n$ iterations, where the public parameter λ controls the failure probability.

4.1 Weak Graded Multicast with Mobile Omissions Π_{WGMMO}

Set $q_{\mathrm{mob}} = t + r + 1$. Under $2t + s + r < n$, we have

$$q_{\mathrm{mob}} \le n - t - s, \qquad n - t - r - s \ge t + 1, \qquad n - t - r \ge t + s + 1. \tag{1}$$

Notation Each invocation of Π_{WGMMO} has a designated sender P_i with input m_i and consists of three rounds: Send, Report, and Confirm. At the end of Confirm, every non-Byzantine party P_j outputs (m_j, g_j). Let $D_{\mathrm{conf}} = \mathcal{N} \setminus R_{\mathrm{conf}}$ be the set of non-Byzantine parties that are receive-reliable in the Confirm round.

Theorem 1 (Properties of Π_{WGMMO}). *Under $2t+s+r < n$, every invocation of Π_{WGMMO} satisfies the following properties:*

- ***Weak Graded Consistency.*** *If $P_i \notin B_{\mathrm{send}}$ and some party $P \notin B_{\mathrm{conf}}$ outputs $(m, 1)$, then every $P' \in D_{\mathrm{conf}}$ outputs (m', g') with $m' = m$.*
- ***Validity.*** *If P_i is send-reliable in the Send round and sends m, then every party that is receive-reliable in the Confirm round outputs $(m_j, 1)$ with $m_j = m$.*
- ***Termination.*** *Every non-Byzantine party terminates after three rounds.*

The proof is given after the construction.

Construction Receivers ignore invalid messages and, in each round, process only the first valid message signed by each party for that round (Section 3.4). Let $m_{i\to j}$ denote P_i's signed Send round message to P_j.

A valid report is a reporter-signed tuple $(\mathsf{Report}, i, \mathsf{ACK}, m_{i\to j})$ or $(\mathsf{Report}, i, \mathsf{NACK})$. A valid signed $(\mathsf{Confirm}, i, \mathcal{R})$ message contains at most one valid report per reporter and is *rejecting* if it contains NACKs from q_{mob} distinct reporters.

Each party keeps the first valid sender input it obtains in Send or from a subsequent ACK. It also keeps at most one valid report per reporter, whether received directly or within a valid Confirm message.

Protocol 4.1: Π_{WGMMO} (single-sender weak graded multicast with mobile omissions)

1. **Send.** The designated sender P_i signs m and multicasts the signed message.
2. **Report.** Each party P_j signs and multicasts exactly one report: $(\mathsf{Report}, i, \mathsf{ACK}, m_{i\to j})$ if it received the Send message, or $(\mathsf{Report}, i, \mathsf{NACK})$ otherwise.
3. **Confirm.** Each party P_j signs and multicasts $(\mathsf{Confirm}, i, \mathcal{R}_j)$, where $\mathcal{R}_j$ contains the Report messages it received.

Final output. At the end of the Confirm round, each party outputs $(m, 1)$ for the input m it has kept if it has an ACK for m and valid Confirm messages from at least q_{mob} distinct confirmers, and none of its processed valid Confirm messages contains a *rejection proof*. Otherwise, it outputs $(m, 0)$ if it has an ACK for m, and $(\perp, 0)$ if it does not.

Properties of Π_{WGMMO}

Proof (Theorem 1). **Weak Graded Consistency.** Suppose $P_i \notin B_{\text{send}}$ and $P_j \notin B_{\text{conf}}$, and P_j outputs $(m, 1)$. The forward-secure signatures of Section 3.4 ensure that P_i's Send round input remains unique even after later corruption. Thus every reporter in $\mathcal{N}$ sends either an ACK for m or a NACK. Assume that no reporter that is send-reliable in the Report round sends an ACK for m. Then at least $n-t-s \geq q_{\text{mob}}$ send-reliable reporters in $\mathcal{N}$ multicast NACKs. Among the $q_{\text{mob}} = t+r+1$ Confirm signers observed by P_j, at least one belongs to $\mathcal{N} \setminus R_{\text{rep}}$. Its Confirm message contains these NACKs and is rejecting, contradicting P_j's grade-one output.

Thus, some reporter that is send-reliable in the Report round multicasts an ACK for m, which reaches at least $n - t - r$ parties in $\mathcal{N}$. The reporter's signed Report-round message also remains unique under later corruption, so these receivers keep this ACK. At most s of these parties suffer send omissions in Confirm, so at least $n-t-r-s \geq t+1$ forward the ACK. Every party in D_{conf} receives the ACK and outputs (m, g) for some $g \in \{0, 1\}$, since the sender's valid input is unique.

Validity. Suppose P_i is send-reliable in Send and sends m. The forward-secure signatures of Section 3.4 ensure that this Send round input remains unique even if P_i is corrupted later. Every party in $\mathcal{N} \setminus R_{\text{send}}$ receives m, so every valid NACK is signed by a party in $B \cup R_{\text{send}}$. There are at most $t + r < q_{\text{mob}}$ such signers; hence no valid Confirm message rejects m. At least $n - t - r - s \geq t + 1$ parties in $\mathcal{N}$ are receive-reliable in Send and send-reliable in Report, so they multicast an ACK for m. By the same forwarding argument, every party that is receive-reliable in Confirm receives this ACK. Each such party also receives at least $n - t - s \geq q_{\text{mob}}$ valid Confirm messages from send-reliable parties and outputs $(m, 1)$.

Termination. Every non-Byzantine party outputs at the end of Confirm, after three rounds.

4.2 Certificates, proposals, and three votes

After Initialization, Π_{BasicBA} executes L iterations and then a finalization round. Each iteration consists of Status, Propose, Vote 1, Vote 2, Vote 3, and Certificate dissemination. A rank-zero certificate certifies initialization inputs; a rank-k certificate certifies emphatic Vote-3 inputs from iteration $k > 0$. Certificates support values and constrain subsequent proposals. Local decisions are recorded during Certificate dissemination. Whenever a rule specifies the "least" element, ties remaining after comparing ranks or VRF evaluations as specified by the rule are resolved using fixed public total orders. Fix a public default value $v_{\text{def}} \in \mathcal{V}$.

An *initialization message* is an input in $\mathcal{V}$ signed during the initialization round. A rank-zero certificate $C_0(v)$ contains initialization messages for v from $t + 1$ distinct parties. A party's *highest certificate below* k is a valid certificate of maximum rank among those it holds with ranks in $\{0, \ldots, k - 1\}$, with ties broken by the public order; it is $\perp$ if the party holds no such certificate. Let ℓ_k^{prop} be the proposer label for iteration k. Proposer P_j privately learns its VRF evaluation $u_{j,k} := Y_{j,\ell_k^{\text{prop}}}$ and a corresponding eligibility proof. In Π_{BasicBA}, the proposer difficulty is one, so every party is eligible to propose. A signed proposal header a binds the iteration, proposer j, value v, certified evaluation $u_{j,k}$, and corresponding eligibility proof. Write $\tau(a) = u_{j,k}$ for the header's evaluation.

A valid Status message in iteration k is a signed tuple (Status, k, C), where C is either $\perp$ or a valid certificate of rank below k. A valid proposal in iteration k is a tuple $\pi = (\mathsf{Propose}, k, a, \mathcal{S})$ signed by the proposer named in a. The collection $\mathcal{S}$ contains exactly q_{mob} valid iteration-k Status messages signed by distinct parties, and the header a satisfies the following *maximum-rank value rule*:

1. If the highest certificate rank in $\mathcal{S}$ is positive, all certificates of that rank must certify the same value, which the proposal must use.
2. If the highest certificate rank is zero, the proposal must use the least value certified at that rank.
3. If every Status message contains $\perp$, any value in $\mathcal{V}$ is valid. A proposer following the protocol uses its own input if available, or v_{def} otherwise.

The message types for the three vote phases are Vote_1, Vote_2, and Vote_3. In vote phase $d \in \{1, 2, 3\}$, a sender signs one of

$$(\mathsf{Vote}_d, k, \mathsf{NIL}), \qquad (\mathsf{Vote}_d, k, a, c), c \in \{0, 1\}, \quad (\mathsf{Vote}_d, k, \mathsf{SPLIT}, a, a'),$$

where a split names two distinct headers with the same evaluation. Its sender labels it emphatic by setting $c = 1$, or non-emphatic by setting $c = 0$. This signed support bit is distinct from the receiver's output grade g in Π_{WGMMO}. An emphatic Vote_1 message additionally includes a valid proposal and has the form $(\mathsf{Vote}_1, k, a, 1, \pi)$, where $\pi = (\mathsf{Propose}, k, a, \mathcal{S})$ is signed by its proposer.

A valid vote is signed by its sender in the current phase. A NIL message names no header; and a split names two distinct valid headers for the current iteration with the same evaluation. An emphatic Vote_1 message must also include a valid proposal. The sender labels NIL and split votes non-emphatic through their message type; neither carries a support bit. A split prevents selection of a single header when its evaluation is minimum and supports neither of the headers it contains.

The *minimum-header rule* returns NIL if there is no valid non-NIL message. Otherwise, it considers only messages whose headers attain the smallest VRF evaluation. If a SPLIT occurs at this minimum, it returns the least such SPLIT under the fixed public order. Otherwise, it returns the unique header, or a SPLIT containing the two least distinct headers if several attain the minimum.

After each vote phase, parties use only valid messages for that phase and the current iteration. For $k > 0$, a rank-k certificate $C_k(a)$ consists of emphatic Vote_3 messages for header a in iteration k, signed by $t + 1$ distinct senders. An *announcement* is a signed tuple of the form $(\mathsf{Disseminate}, k, \mathsf{NIL})$ or $(\mathsf{Disseminate}, k, C_k(a))$.

Protocol 4.2: $\Pi_{\mathsf{BasicIter}}(k)$ (one iteration of the all-party protocol)

1. **Status.** Every party P signs and multicasts (Status, k, C), where C is its highest-rank certificate below k, or $\perp$ if it has none.
2. **Propose.** Each party P selects the least q_{mob} valid Status messages it received from distinct signers. If it received fewer than q_{mob} such messages, it sends nothing. Otherwise, it applies the maximum-rank value rule to obtain a header a, then signs and multicasts $\pi = (\mathsf{Propose}, k, a, \mathcal{S})$, where $\mathcal{S}$ is the selected collection of Status messages.
3. **Vote 1.** Every party P applies the minimum-header rule to the valid proposals it received in the Propose round. If the rule returns NIL, P inputs $(\mathsf{Vote}_1, k, \mathsf{NIL})$; if it returns a split, P inputs $(\mathsf{Vote}_1, k, \mathsf{SPLIT}, a, a')$. If the rule returns a single header a, P labels its vote as emphatic and inputs $(\mathsf{Vote}_1, k, a, 1, \pi)$, where π is the valid proposal supporting a. Each party uses its message as the input to its own Π_{WGMMO} invocation, and all n invocations run in parallel.
4. **Vote 2.** Every party P applies the minimum-header rule to the valid Vote_1 messages in its outputs after the Vote-1 Confirm round. It in-

puts $(\mathsf{Vote}_2, k, \mathsf{NIL})$ or $(\mathsf{Vote}_2, k, \mathsf{SPLIT}, a, a')$ if the rule returns NIL or a split, respectively. If the rule returns a single header a, P inputs $(\mathsf{Vote}_2, k, a, c)$, where P sets $c = 1$ exactly when its grade-one outputs contain emphatic Vote_1 inputs for a from at least $t + 1$ distinct designated senders; $c = 0$ otherwise. The n sender-indexed Π_{WGMMO} invocations run in parallel.

5. **Vote 3 and certificate.** Every party P applies the minimum-header rule to the valid Vote_2 messages in its outputs after the Vote-2 phase. It inputs $(\mathsf{Vote}_3, k, \mathsf{NIL})$ or $(\mathsf{Vote}_3, k, \mathsf{SPLIT}, a, a')$ if the rule returns NIL or a split, respectively. If the rule returns a single header a, P inputs $(\mathsf{Vote}_3, k, a, c)$, where P sets $c = 1$ exactly when its grade-one outputs contain emphatic Vote_2 messages for a from the Π_{WGMMO} invocations of $t+1$ distinct designated senders, and its outputs contain valid Vote_2 messages from at least q_{mob} distinct designated senders, regardless of grade; $c = 0$ otherwise. The n sender-indexed Π_{WGMMO} invocations run in parallel. At their end, P forms and stores the least available rank-k certificate from the emphatic Vote_3 messages in its grade-one outputs.
6. **Certificate dissemination.** In $\mathsf{ann}(k)$, every party P inputs a signed $(\mathsf{Disseminate}, k, C_k(a))$ containing its least rank-k certificate into its own Π_{WGMMO} invocation. If it has no such certificate, it inputs a signed $(\mathsf{Disseminate}, k, \mathsf{NIL})$ instead. All n invocations run in parallel. At their end, each party stores every valid rank-k certificate carried by an announcement in its outputs. Each party P maintains a *local decision value* v_P^{dec}, initialized to $\perp$ before the iterations. If $v_P^{\mathrm{dec}} = \perp$ and P's grade-one outputs contain announcements from at least $t + 1$ distinct designated input signers, each carrying a valid rank-k certificate with the same header a, then P sets v_P^{dec} to the value in a.

Protocol 4.3: Π_{BasicBA} (complete all-party Byzantine Agreement with Mobile-Omission Faults)

1. **Initialize.** Every party P with an input in $\mathcal{V}$ multicasts its signed $\mathsf{Initialization}$ message containing that input; parties without an input send no initialization message. Every receiver forms and stores $C_0(v)$ for each $v \in \mathcal{V}$ supported by valid $\mathsf{Initialization}$ messages from $t + 1$ distinct signers. Set $v_P^{\mathrm{dec}} = \perp$.
2. **Run iterations.** Execute $\Pi_{\mathsf{BasicIter}}(k)$ for $k = 1, \ldots, L$. Every party continues to participate in all rounds of the protocol, even after fixing its local decision.
3. **Finalize.** In a single public final round, every party P with $v_P^{\mathrm{dec}} \neq \perp$ signs and multicasts $(\mathsf{Finalize}, v_P^{\mathrm{dec}})$. A party that receives valid $\mathsf{Finalize}$ messages for the same value $v \in \mathcal{V}$ from $t + 1$ distinct signers outputs (v, False). Otherwise, it outputs $(\perp, \mathsf{True})$.

4.3 Safety

For each vote phase d, let $D_d = \mathcal{N} \setminus R_{\text{conf},d}$, where $R_{\text{conf},d}$ is the receive-omission set shared by all parallel Vote-d invocations in their Confirm round.

Lemma 1 (Emphatic votes originate from valid proposals). *If a non-Byzantine sender sends an emphatic phase-d vote for header a, then a valid proposal with header a was sent in the same iteration.*

Proof. For $d = 1$, the non-Byzantine sender includes in its Vote_1 message a valid proposal received in the Propose round. For $d > 1$, an emphatic vote requires grade-one outputs for emphatic votes in the preceding phase from $t + 1$ distinct input senders. At least one of these senders is non-Byzantine, so the induction hypothesis yields an emphatic Vote_1 message and its supporting proposal.

Lemma 2 (Vote-2 comparison). *Suppose a non-Byzantine Vote-2 sender sends an emphatic Vote_2 message for a. Then every party in D_1 sends a non-NIL Vote_2 message whose headers have evaluation at most $\tau(a)$. If equality holds, any such single-header message has header a.*

Proof. The sender of the emphatic vote obtained grade-one outputs for emphatic Vote_1 inputs on a from $t+1$ distinct designated senders. Choose a non-Byzantine designated sender among them. By weak graded consistency (Theorem 1), every $P \in D_1$ outputs that sender's Vote_1 input in the corresponding Π_{WGMMO} invocation. Thus, P's final outputs across the parallel Vote-1 invocations contain this emphatic Vote_1 input for a. By the minimum-header rule, P sends a non-NIL Vote_2 message whose headers have evaluation at most $\tau(a)$. If equality holds and P sends a single-header Vote_2 message, neither a split nor a distinct header can occur at the minimum evaluation: either would cause the rule to return a split. The single-header message must have header a.

Theorem 2 (Consistency within each iteration). *For every $k > 0$, any two valid rank-k certificates have the same header and value.*

Proof. Let $C_k(a)$ and $C_k(a')$ be valid certificates, with $\tau(a) \leq \tau(a')$ without loss of generality. Each contains an emphatic Vote_3 message from a non-Byzantine sender, say P for a and P' for a'. The grade-one outputs supporting P's vote contain emphatic Vote_2 messages from $t+1$ distinct designated senders. At least one of these senders is non-Byzantine. By Lemma 2, every party in D_1 sends a non-NIL Vote_2 message whose headers have evaluation at most $\tau(a)$.

By the coverage condition, P''s outputs contain valid Vote_2 messages from $t+r+1$ distinct senders. At most $t+r$ parties lie outside D_1, so these messages include one from some $X \in D_1$. The minimum-header rule gives $\tau(a') \leq \tau(a)$. Equality follows from our choice of ordering, and X's message attains the selected minimum.

This message cannot be a split, since P' selected a single header. The equality case of Lemma 2 implies that its header is a. Distinct headers at the minimum would also cause the rule to return a split, so P' must select $a' = a$. The two certificates have the same header and value.

Lemma 3 (Rank-certificate dissemination). *If a non-Byzantine party first sets its* local decision *value in iteration k using header a, then every non-Byzantine party that is receive-reliable in the Confirm round of* $\mathsf{ann}(k)$ *receives a valid rank-k certificate with header a by the end of that round. When $k < L$, this occurs before the Status round of iteration $k+1$.*

Proof. Let k be the iteration in which the party first assigns v_P^{dec}, and let a be the header causing that assignment. The party obtained grade-one outputs for announcements from $t+1$ distinct designated announcers, each carrying a valid rank-k certificate with header a. Choose a non-Byzantine announcer P_i among them. Since $P_i \notin B_{\mathrm{send}}$ in the announcement send round, weak graded consistency in Theorem 1 ensures that Π_{WGMMO} outputs that announcement at every non-Byzantine party that is receive-reliable in the Confirm round of $\mathsf{ann}(k)$. Every such party stores the valid rank-k certificate with header a carried by the announcement.

Theorem 3 (Agreement of non-Byzantine local decisions). *All nonempty local decisions of non-Byzantine parties in one execution have the same value.*

Proof (Theorem 3). If $v_P^{\mathrm{dec}} = \perp$ for every $P \in \mathcal{N}$, the claim is trivial. Otherwise, let k be the earliest iteration in which a non-Byzantine party first assigns its local decision value, and choose one such assignment. Let a be the corresponding announcement header and v the assigned value. Write $R_{\mathrm{conf,ann}}(k)$ for the receive-omission set in the Confirm round of $\mathsf{ann}(k)$. By Lemma 3, every non-Byzantine party that is receive-reliable in that round holds a certificate for a at the end of the round. There are at least $n-t-r$ such parties. Every later valid proposal contains $q_{\mathrm{mob}} = t+r+1$ Status messages and must include a message from this set, because its complement has size at most $t+r$. These parties keep $C_k(a)$ and thereafter send Status messages of rank at least k.

We prove by induction that *all* certificates of rank $\ell \geq k$ have value v. The base case $\ell = k$ follows from Theorem 2. For $\ell > k$, choose a non-Byzantine signer among the $t+1$ emphatic Vote-3 signers of a rank-ℓ certificate. Lemma 1 guarantees a valid proposal supporting this vote. That proposal must contain a Status message from a non-Byzantine party that is receive-reliable in the Confirm round of $\mathsf{ann}(k)$, so its maximum status rank is at least k. By the induction hypothesis, every certificate at a rank in $[k, \ell)$ has value v, while Theorem 2 ensures a unique value within each rank. Hence the maximum-rank value rule assigns value v to the proposal. Every non-$\perp$ local decision value of a non-Byzantine party was first assigned from a header supported by a valid certificate in some iteration $\ell \geq k$. By the induction, that header has value v, and the assigned value never changes. Thus all non-$\perp$ local decision values agree.

Theorem 4 (Validity). *If $|U \cap \mathcal{N}| \leq r$ and every party in $\mathcal{N} \setminus U$ starts with the same value $v \in \mathcal{V}$, every positive-rank certificate and every nonempty local decision of a non-Byzantine party has value v.*

Proof. A rank-zero certificate contains signatures from $t+1$ parties, at least one of which is non-Byzantine. Such a signer sends an initialization message only if

it has an input, which must be v. Hence every rank-zero certificate has value v. At least $|\mathcal{N} \setminus (U \cup S_{\text{init}})| \geq n-t-r-s \geq t+1$ parties have input v and are send-reliable in the initialization round, so they send signed Initialization messages for v. Every non-Byzantine receiver that is receive-reliable in that round receives these messages and stores $C_0(v)$. Thus the holder set $\mathcal{N} \setminus R_{\text{init}}$ has size at least $n-t-r$.

Every later valid proposal contains $q_{\text{mob}} = t+r+1$ Status messages, so one of their senders lies in $\mathcal{N} \setminus R_{\text{init}}$, whose complement has size at most $t+r$. For a positive-rank certificate, choose a non-Byzantine sender among its $t+1$ emphatic Vote-3 input senders. Lemma 1 guarantees a valid supporting proposal. Its status collection contains a message from a non-Byzantine party that is receive-reliable in the initialization round, so the proposal includes at least one certificate. By the induction hypothesis, every certificate at a smaller rank has value v, and Theorem 2 ensures consistency within each rank. The maximum-rank value rule selects v. This proves the inductive step for positive-rank certificates. Every first assignment to v_P^{dec} for $P \in \mathcal{N}$ uses the value of such a certificate.

4.4 Progress, Strong NLU, and probability

Call iteration k *good* if the globally smallest evaluation is unique and its proposer P_i is send-reliable in the Propose round and receive-reliable in the Status round of that iteration. The adversary can infer each non-Byzantine party's eligibility only from its current-round transcripts, so its choice is independent of each party's proposal evaluation. Under the ideal VRF primitive, the conditional probability that an iteration is bad is therefore at most $(t+s+r)/n$.

Lemma 4 (Progress in a good iteration). *In a* good iteration*, at least $n-t-r$ non-Byzantine parties have a non-$\perp$ local decision value by the end of the iteration.*

Proof. Let $a^\star$ be the good proposer's header. Because the proposer is outside $B_{\text{prop},k}$ and receive-reliable in the Status round, it receives Status messages from at least $n-t-s \geq q_{\text{mob}}$ senders that are send-reliable in that round. Theorem 2 ensures that all positive-rank certificates at the maximum status rank agree, and the maximum-rank value rule is also defined in the rank-zero and all-$\perp$ cases. The proposer forms and sends a valid proposal. Its VRF evaluation is the unique global minimum. Forward-secure authentication rules out a second header from this proposer in that round, even after a later corruption.

Every non-Byzantine party that is receive-reliable in the Propose round receives this proposal. At least $n-t-r-s \geq t+1$ of these parties are also send-reliable in the Vote-1 Send round and input emphatic Vote_1 messages for $a^\star$. By the Validity property in Theorem 1, every party in D_1 obtains grade-one outputs for all these inputs. Since $a^\star$ is the unique valid header at the globally smallest evaluation, every $P \in D_1$ selects $a^\star$ from the valid Vote-1 messages in its outputs, which include $t+1$ grade-one emphatic inputs.

At least $n-t-r-s \geq t+1$ parties in D_1 are also send-reliable in the Vote-2 Send round. They input emphatic Vote_2 messages for $a^\star$, and validity

gives every party in D_2 grade-one outputs for these inputs. In addition, at least $n - t - s \geq q_{\text{mob}}$ Vote-2 senders are send-reliable in that phase's Send round. Each inputs one valid Vote_2 message, possibly NIL. By validity of Π_{WGMMO}, every party in D_2 outputs this Vote_2 input with grade one. Thus, for every $P \in D_2$, the valid Vote_2 messages in P's outputs satisfy the coverage requirement, with signed NIL inputs also counting toward coverage.

Every $P \in D_2$ selects $a^\star$, has $t + 1$ grade-one emphatic Vote-2 inputs in its outputs, and satisfies coverage. It inputs an emphatic Vote_3 message. At least $n - t - r - s \geq t + 1$ such senders are send-reliable in the Vote-3 Send round. Validity of Π_{WGMMO} gives every party in D_3 grade-one outputs from these senders. Their signed emphatic Vote_3 messages form a valid rank-k certificate for $a^\star$.

There are at least $n - t - r$ such rank-certificate holders in D_3, of which at least $n-t-r-s \geq t+1$ are send-reliable in the announcement Send round. Validity of Π_{WGMMO} ensures that every non-Byzantine party that is receive-reliable in the announcement Confirm round obtains grade-one outputs for these announcements carrying valid rank-k certificates. By Theorem 2, their rank certificates share the same header $a^\star$. There are at least $n - t - r$ such receivers, each of which meets the announcement threshold for that header. If its v_P^{dec} is $\perp$, it sets that variable to the value in $a^\star$. Otherwise, it leaves its local decision value unchanged. Thus all these parties have non-$\perp$ local decision values.

Theorem 5 (Correctness of Π_{BasicBA}).

Termination, Validity, and Agreement hold unconditionally in the idealized cryptography model. Except with probability at most $\epsilon_{\text{mob}} = \left(\frac{t+s+r}{n}\right)^L$, all parties that are receive-reliable in the final round output the same value without a flag, while each non-Byzantine party in R_{fin} outputs either that value or a flag. Thus, for $L = \lambda n$, the failure probability is negligible in λ, and Π_{BasicBA} is a secure strong-NLU Byzantine agreement protocol.

Proof. Any unflagged output is supported by $t+1$ valid final-round signatures. At least one signer is non-Byzantine and signs only its non-$\perp$ value v_P^{dec}. Theorem 3 gives Agreement, and Theorem 4 gives Validity. By Lemma 4, after a good iteration, at least $n - t - r$ parties in $\mathcal{N}$ have non-$\perp$ local decision values. At most s of these parties are in S_{fin}, so at least $n-t-r-s \geq t+1$ parties multicast valid finalization messages for their common local decision value to every party that is receive-reliable in the final round. Each such receiver outputs that value unflagged. Any other non-Byzantine party either outputs the same value or flags. Hence every flagged non-Byzantine party lies in R_{fin}, giving both Ordinary NLU and the bound of r flags. Since the parameters are integral, $2t+s+r < n$ implies $t + s + r \leq n - t - 1 \leq n - 1$. Conditioned on the execution history preceding iteration k, the ideal VRF-sortition bound applied to $B_{\text{prop},k} \cup R_{\text{status},k} \cup S_{\text{prop},k}$ bounds the probability that the fresh minimum-evaluation proposer lies in this set by $(t + s + r)/n$. Applying this bound in each of the L iterations and using the chain rule gives the stated error bound. For $L = \lambda n$, this error is at most $(1-1/n)^{\lambda n} \leq e^{-\lambda}$. The round schedule is fixed, so Termination is unconditional.

4.5 Rounds and communication

Theorem 6 (Complexity of Π_{BasicBA}). *Under the assumptions of Section 3.4 and $2t + s + r < n$, suppose that every value in $\mathcal{V}$ can be encoded using $O(\kappa)$ bits. For $L = \lambda n$, Π_{BasicBA} takes exactly $14\lambda n + 2$ synchronous rounds, and the non-Byzantine parties send a total of $O(\kappa\lambda n^7)$ bits.*

Proof (Theorem 6). We count only communication by non-Byzantine parties. Initialization and finalization use one round each. Every iteration has two direct rounds and four three-round graded phases, giving $2 + L(2 + 4 \cdot 3) = 14L + 2$ rounds. The n sender-indexed executions in each graded phase are batched, so each party sends at most one message to each receiver per round. The entire protocol uses at most $(14L + 2)n^2$ point-to-point transmissions.

By the value-length assumption, a signed initialization message, proposal header, or finalization message has $O(\kappa)$ bits. A rank certificate contains $t + 1$ signed initialization or emphatic Vote_3 messages and has $O(\kappa(t+1))$ bits. A proposal contains q_{mob} Status messages, each carrying at most one rank certificate. Thus an emphatic Vote_1 input, which includes a proposal, has $O(\kappa q_{\mathrm{mob}}(t + 1))$ bits. All other valid graded inputs are no larger.

A batched Report message contains at most n signed reports, each carrying at most one graded input, and has $O(\kappa n q_{\mathrm{mob}}(t + 1))$ bits. A batched Confirm message contains at most n reports for each of the n input senders, giving $O(\kappa n^2 q_{\mathrm{mob}}(t+1))$ bits. With at most n^2 point-to-point transmissions per round and a constant number of rounds per iteration, the graded phases together use $O(\kappa n^4 q_{\mathrm{mob}}(t+1))$ bits per iteration. The direct Status and Propose rounds have no larger cost.

Initialization and finalization each use $O(\kappa n^2)$ bits. Summing over L iterations and using $q_{\mathrm{mob}}, t + 1 \le n$, the total communication is

$$O\big(\kappa n^2 + \kappa L n^4 q_{\mathrm{mob}}(t + 1)\big) = O(\kappa L n^6).$$

For $L = \lambda n$, this gives $O(\kappa\lambda n^7)$ bits.

References

1. Abraham, I., Alpturer, K., Garimidi, P., Neu, J., Roughgarden, T., Tas, E.N.: Beyond Leaderless: Randomized Consensus Against Mobile Crashes. Cryptology ePrint Archive, Paper 2026/2952 (2026), https://eprint.iacr.org/2026/2952
2. Abraham, I., Chan, T.H.H., Dolev, D., Nayak, K., Pass, R., Ren, L., Shi, E.: Communication complexity of byzantine agreement, revisited. In: Robinson, P., Ellen, F. (eds.) 38th ACM Symposium Annual on Principles of Distributed Computing. pp. 317–326. Association for Computing Machinery, Toronto, ON, Canada (Jul 29 – Aug 2, 2019). https://doi.org/10.1145/3293611.3331629
3. Abraham, I., Devadas, S., Dolev, D., Nayak, K., Ren, L.: Synchronous byzantine agreement with expected $O(1)$ rounds, expected $O(n^2)$ communication, and optimal resilience. In: Goldberg, I., Moore, T. (eds.) FC 2019: 23rd International Conference on Financial Cryptography and Data Security. Lecture Notes in Computer Science, vol. 11598, pp. 320–334. Springer, Cham, Switzerland, Frigate Bay, St. Kitts and Nevis (Feb 18–22, 2019). https://doi.org/10.1007/978-3-030-32101-7_20
4. Abraham, I., Stern, G.: Consensus with one mobile crash in synchrony or one crash in asynchrony has infinite executions. Decentralized Thoughts (2024), https://decentralizedthoughts.github.io/2024-03-07-mobile-is-FLP/, march 7, 2024. Accessed September 12, 2026
5. Banu, N., Souissi, S., Izumi, T., Wada, K.: An improved Byzantine agreement algorithm for synchronous systems with mobile faults. International Journal of Computer Applications **43**(22), 1–7 (2012). https://doi.org/10.5120/6400-8878
6. Bonnet, F., Défago, X., Nguyen, T.D., Potop-Butucaru, M.: Tight bound on mobile Byzantine agreement. Theoretical Computer Science **609**, 361–373 (2016). https://doi.org/10.1016/j.tcs.2015.10.019
7. Brazitikos, K., Zikas, V.: General adversary structures in byzantine agreement and multi-party computation with active and omission corruption. In: Boyle, E., Mahmoody, M. (eds.) TCC 2024: 22nd Theory of Cryptography Conference, Part IV. Lecture Notes in Computer Science, vol. 15367, pp. 200–233. Springer, Cham, Switzerland, Milan, Italy (Dec 2–6, 2024). https://doi.org/10.1007/978-3-031-78023-3_7
8. Buhrman, H., Garay, J.A., Hoepman, J.H.: Optimal resiliency against mobile faults. In: Twenty-Fifth International Symposium on Fault-Tolerant Computing. Digest of Papers. pp. 83–88. IEEE Computer Society (1995)
9. Cachin, C., Kursawe, K., Lysyanskaya, A., Strobl, R.: Asynchronous verifiable secret sharing and proactive cryptosystems. In: Atluri, V. (ed.) ACM CCS 2002: 9th Conference on Computer and Communications Security. pp. 88–97. ACM Press, Washington, DC, USA (Nov 18–22, 2002). https://doi.org/10.1145/586110.586124
10. Canetti, R., Herzberg, A.: Maintaining security in the presence of transient faults. In: Desmedt, Y. (ed.) Advances in Cryptology – CRYPTO'94. Lecture Notes in Computer Science, vol. 839, pp. 425–438. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 21–25, 1994). https://doi.org/10.1007/3-540-48658-5_38
11. Charron-Bost, B., Schiper, A.: The Heard-Of model: Computing in distributed systems with benign faults. Distributed Computing **22**(1), 49–71 (2009). https://doi.org/10.1007/s00446-009-0084-6

12. Chen, J., Micali, S.: Algorand. arXiv:1607.01341v9 (2017). https://doi.org/10.48550/arXiv.1607.01341, https://arxiv.org/abs/1607.01341v9
13. Civit, P., Gilbert, S., Guerraoui, R., Komatovic, J., Paramonov, A., Vidigueira, M.: All byzantine agreement problems are expensive. In: Gelles, R., Olivetti, D., Kuznetsov, P. (eds.) 43rd ACM Symposium Annual on Principles of Distributed Computing. pp. 157–169. Association for Computing Machinery, Nantes, France (Jun 17–21, 2024). https://doi.org/10.1145/3662158.3662780
14. Cohen, R., Loss, J., Moran, T.: Efficient agreement over byzantine gossip. In: Clark, J., Shi, E. (eds.) FC 2024: 28th International Conference on Financial Cryptography and Data Security, Part I. Lecture Notes in Computer Science, vol. 14744, pp. 74–91. Springer, Cham, Switzerland, Willemstad, Curaçao (Mar 4–8, 2024). https://doi.org/10.1007/978-3-031-78676-1_5
15. Coulouma, É., Godard, E., Peters, J.: A characterization of oblivious message adversaries for which consensus is solvable. Theoretical Computer Science **584**, 80–90 (2015). https://doi.org/10.1016/j.tcs.2015.01.024
16. David, B., Gazi, P., Kiayias, A., Russell, A.: Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 66–98. Springer, Cham, Switzerland, Tel Aviv, Israel (Apr 29 – May 3, 2018). https://doi.org/10.1007/978-3-319-78375-8_3
17. Efron, Y., Neu, J., Ren, L., Tas, E.N.: Optimal good-case latency for sleepy consensus. In: Daemen, J., Thomé, E. (eds.) Advances in Cryptology – EUROCRYPT 2026, Part III. Lecture Notes in Computer Science, vol. 16543, pp. 397–426. Springer, Cham, Switzerland, Rome, Italy (May 10–14, 2026). https://doi.org/10.1007/978-3-032-25324-8_14
18. Eldefrawy, K., Loss, J., Terner, B.: How byzantine is a send corruption? In: Ateniese, G., Venturi, D. (eds.) ACNS 2022: 20th International Conference on Applied Cryptography and Network Security. Lecture Notes in Computer Science, vol. 13269, pp. 684–704. Springer, Cham, Switzerland, Rome, Italy (Jun 20–23, 2022). https://doi.org/10.1007/978-3-031-09234-3_34
19. Elsheimy, F., Kamp, S.H., Loss, J.: Early stopping is cheap. Cryptology ePrint Archive, Report 2025/1107 (2025), https://eprint.iacr.org/2025/1107
20. Feng, H., Lu, Z., Tang, Q., Ye, Y.: Optimal byzantine agreement in the presence of message drops. In: Hanaoka, G., Yang, B.Y. (eds.) Advances in Cryptology – ASIACRYPT 2025, Part V. Lecture Notes in Computer Science, vol. 16249, pp. 163–195. Springer, Singapore, Singapore, Melbourne, VIC, Australia (Dec 8–12, 2025). https://doi.org/10.1007/978-981-95-5116-3_6
21. Frankel, Y., Gemmell, P., MacKenzie, P.D., Yung, M.: Optimal resilience proactive public-key cryptosystems. In: 38th Annual Symposium on Foundations of Computer Science. pp. 384–393. IEEE Computer Society Press, Miami Beach, Florida (Oct 19–22, 1997). https://doi.org/10.1109/SFCS.1997.646127
22. Garay, J.A.: Reaching (and maintaining) agreement in the presence of mobile faults. In: Distributed Algorithms. Lecture Notes in Computer Science, vol. 857, pp. 253–264. Springer (1994). https://doi.org/10.1007/BFb0020438
23. Garay, J.A., Katz, J., Kumaresan, R., Zhou, H.S.: Adaptively secure broadcast, revisited. In: Gavoille, C., Fraigniaud, P. (eds.) 30th ACM Symposium Annual on Principles of Distributed Computing. pp. 179–186. Association for Computing Machinery, San Jose, CA, USA (Jun 6–8, 2011). https://doi.org/10.1145/1993806.1993832

24. Garay, J.A., Perry, K.J.: A continuum of failure models for distributed computing. In: Segall, A., Zaks, S. (eds.) Distributed Algorithms, 6th International Workshop, WDAG '92, Haifa, Israel, November 2-4, 1992, Proceedings. Lecture Notes in Computer Science, vol. 647, pp. 153–165. Springer (1992). https://doi.org/10.1007/3-540-56188-9_11
25. Godard, E., Peters, J.: Consensus vs. broadcast in communication networks with arbitrary mobile omission faults. In: Structural Information and Communication Complexity. Lecture Notes in Computer Science, vol. 6796, pp. 29–41. Springer (2011). https://doi.org/10.1007/978-3-642-22212-2_4
26. Guo, Y., Pass, R., Shi, E.: Synchronous, with a chance of partition tolerance. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part I. Lecture Notes in Computer Science, vol. 11692, pp. 499–529. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26948-7_18
27. Herzberg, A., Jarecki, S., Krawczyk, H., Yung, M.: Proactive secret sharing or: How to cope with perpetual leakage. In: Coppersmith, D. (ed.) Advances in Cryptology – CRYPTO'95. Lecture Notes in Computer Science, vol. 963, pp. 339–352. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 27–31, 1995). https://doi.org/10.1007/3-540-44750-4_27
28. Kamp, S.H., Loss, J., Nayak, K., Shi, K.: OmniBA: Round-efficient BA with quadratic communication under mixed faults. Cryptology ePrint Archive, Report 2025/2006 (2025), https://eprint.iacr.org/2025/2006
29. Kim, J., Mehta, V., Nayak, K., Shrestha, N.: Making synchronous BFT protocols secure in the presence of mobile sluggish faults. Cryptology ePrint Archive, Report 2021/603 (2021), https://eprint.iacr.org/2021/603
30. Loss, J., Shi, K., Stern, G.: Consensus in the presence of overlapping faults and total omission. In: Boyle, E., Mahmoody, M. (eds.) TCC 2024: 22nd Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 15364, pp. 353–382. Springer, Cham, Switzerland, Milan, Italy (Dec 2–6, 2024). https://doi.org/10.1007/978-3-031-78011-0_12
31. Loss, J., Stern, G.: Zombies and ghosts: Optimal byzantine agreement in the presence of omission faults. In: Rothblum, G.N., Wee, H. (eds.) TCC 2023: 21st Theory of Cryptography Conference, Part IV. Lecture Notes in Computer Science, vol. 14372, pp. 395–421. Springer, Cham, Switzerland, Taipei, Taiwan (Nov 29 – Dec 2, 2023). https://doi.org/10.1007/978-3-031-48624-1_15
32. Malkhi, D., Momose, A., Ren, L.: Towards practical sleepy BFT. In: Meng, W., Jensen, C.D., Cremers, C., Kirda, E. (eds.) ACM CCS 2023: 30th Conference on Computer and Communications Security. pp. 490–503. ACM Press, Copenhagen, Denmark (Nov 26–30, 2023). https://doi.org/10.1145/3576915.3623073
33. Micali, S., Vaikuntanathan, V.: Optimal and player-replaceable consensus with an honest majority. Tech. Rep. MIT-CSAIL-TR-2017–004, Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory (Mar 2017), https://hdl.handle.net/1721.1/107927
34. Momose, A., Ren, L.: Constant latency in sleepy consensus. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022: 29th Conference on Computer and Communications Security. pp. 2295–2308. ACM Press, Los Angeles, CA, USA (Nov 7–11, 2022). https://doi.org/10.1145/3548606.3559347
35. Ostrovsky, R., Yung, M.: How to withstand mobile virus attacks (extended abstract). In: Logrippo, L. (ed.) 10th ACM Symposium Annual on Principles of Distributed Computing. pp. 51–59. Association for Computing Machinery, Montreal, QC, Canada (Aug 19–21, 1991). https://doi.org/10.1145/112600.112605

36. Parvédy, P.R., Raynal, M.: Uniform agreement despite process omission failures. In: 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), 22-26 April 2003, Nice, France, CD-ROM/Abstracts Proceedings. p. 212. IEEE Computer Society (2003). https://doi.org/10.1109/IPDPS.2003.1213388
37. Pass, R., Shi, E.: The sleepy model of consensus. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology – ASIACRYPT 2017, Part II. Lecture Notes in Computer Science, vol. 10625, pp. 380–409. Springer, Cham, Switzerland, Hong Kong, China (Dec 3–7, 2017). https://doi.org/10.1007/978-3-319-70697-9_14
38. Perry, K.J., Toueg, S.: Distributed agreement in the presence of processor and communication faults. IEEE Trans. Software Eng. **12**(3), 477–482 (1986). https://doi.org/10.1109/TSE.1986.6312888
39. Reischuk, R.: A new solution for the Byzantine generals problem. Information and Control **64**(1–3), 23–42 (1985). https://doi.org/10.1016/S0019-9958(85)80042-5
40. Santoro, N., Widmayer, P.: Time is not a healer: Preliminary version. In: STACS 89. Lecture Notes in Computer Science, vol. 349, pp. 304–313. Springer (1989). https://doi.org/10.1007/BFb0028994
41. Santoro, N., Widmayer, P.: Majority and unanimity in synchronous networks with ubiquitous dynamic faults. In: Structural Information and Communication Complexity – SIROCCO 2005. Lecture Notes in Computer Science, vol. 3499, pp. 262–276. Springer (2005). https://doi.org/10.1007/11429647_21
42. Santoro, N., Widmayer, P.: Agreement in synchronous networks with ubiquitous faults. Theoretical Computer Science **384**(2–3), 232–249 (2007). https://doi.org/10.1016/j.tcs.2007.04.036
43. Sasaki, T., Yamauchi, Y., Kijima, S., Yamashita, M.: Mobile Byzantine agreement on arbitrary network. In: Principles of Distributed Systems. Lecture Notes in Computer Science, vol. 8304, pp. 236–250. Springer (2013). https://doi.org/10.1007/978-3-319-03850-6_17
44. Schmid, U., Weiss, B., Keidar, I.: Impossibility results and lower bounds for consensus under link failures. SIAM Journal on Computing **38**(5), 1912–1951 (2009). https://doi.org/10.1137/S009753970443999X
45. Yung, M.: The "mobile adversary" paradigm in distributed computation and systems. In: Georgiou, C., Spirakis, P.G. (eds.) 34th ACM Symposium Annual on Principles of Distributed Computing. pp. 171–172. Association for Computing Machinery, Donostia-San Sebastián, Spain (Jul 21–23, 2015). https://doi.org/10.1145/2767386.2767453
46. Zheng, D., Ye, Y., Gui, X., Tang, Q., Lu, Z.: Optimal Byzantine Atomic Broadcast with Message Drops. Cryptology ePrint Archive, Paper 2026/2150 (2026), https://eprint.iacr.org/2026/2150
47. Zikas, V., Hauser, S., Maurer, U.M.: Realistic failures in secure multi-party computation. In: Reingold, O. (ed.) TCC 2009: 6th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 5444, pp. 274–293. Springer Berlin Heidelberg, Germany (Mar 15–17, 2009). https://doi.org/10.1007/978-3-642-00457-5_17

A Player-Replaceable Byzantine Agreement with Mobile-Omission Faults

We instantiate the all-party voting structure with Π_{PRWGMMO}, whose parameters and committees are specified below. Set $D_\ell = \kappa/n$ for every committee and proposer role, and fix an integer iteration count $\lambda \geq 1$. The execution consists of $M = 14\lambda + 2$ rounds in a fixed order.

A.1 PR-Weak Graded Multicast with Mobile Omissions Π_{PRWGMMO}

The player-replacement construction uses a fresh hidden committee for each sending round. All parties act as receivers, and no party is required to be selected more than once. A single parallel Π_{PRWGMMO} execution processes all eligible inputs, uses a shared committee in each phase, and returns a set of graded outputs. The $\mathcal{ACK}$ lists avoid naming eligible senders whose membership has not yet been revealed.

Notation An input m is a signed message accompanied by its signer's eligibility VRF proof. Every party P_j that remains outside Byzantine control throughout the three-round invocation returns a set $\mathcal{O}(P_j) = \{(m_{i,j}, g_{i,j})\}$, with $g_{i,j} \in \{0, 1\}$, where i identifies an eligible input signer and j identifies the receiver.

Thresholds and graded guarantees Assume the model of Section 3 and the resilience condition

$$2t + s + r \leq (1 - \varepsilon)n, \qquad 0 < \varepsilon < 1. \tag{PR-Mob}$$

Take $64/\varepsilon \leq \kappa \leq n$, set $\eta = \varepsilon/16$, and sample each committee independently in each round with probability κ/n per party. If $n < 64/\varepsilon$, use the all-party construction instead. Define

$$\beta = \left\lceil \left(\frac{t}{n} + \eta \right) \kappa \right\rceil, \qquad \zeta = \left\lceil \left(\frac{t+r}{n} + \eta \right) \kappa \right\rceil, \tag{2}$$

$$q = \left\lfloor \left(\frac{n-t-s}{n} - \eta \right) \kappa \right\rfloor, \qquad h_{\min} = \left\lfloor \left(\frac{n-t-s-r}{n} - \eta \right) \kappa \right\rfloor, \tag{3}$$

$$\gamma = \beta + 1, \qquad c_{\mathrm{cov}} = \zeta + 1. \tag{4}$$

These thresholds in particular satisfy

$$q \geq c_{\mathrm{cov}} > \zeta, \qquad h_{\min} \geq \gamma > \beta, \qquad \beta < q, \tag{5}$$

as shown in Lemma 5. Here β bounds the number of current Byzantine committee members, while ζ also accounts for members in an earlier receive-omission set. The thresholds q and $h_{\min}$ are lower bounds on the number of send-reliable members in a committee's own round; for $h_{\min}$, these members must also be receive-reliable in an earlier round. The constant ε is needed for these sampled thresholds: when $t = r = 0$ and $s = n - 1$, a committee of expected size $\kappa = o(n)$ contains a send-reliable party with probability only κ/n in the worst case.

Lemma 5 (Threshold separation). *Under condition* (PR-Mob) *and* $\kappa \geq 64/\varepsilon$*, the threshold inequalities in* (5) *hold. In particular,* $h_{\min} \geq 1$*.*

Proof. Condition (PR-Mob) gives $(n-2t-s-r)/n \geq \varepsilon$. Using $\lfloor x \rfloor \geq x-1$ and $\lceil x \rceil \leq x+1$ in (2)–(4),

$$q - c_{\mathrm{cov}} \geq \left(\frac{n-2t-s-r}{n} - 2\eta\right)\kappa - 3, \qquad h_{\min} - \gamma \geq \left(\frac{n-2t-s-r}{n} - 2\eta\right)\kappa - 3.$$

Because $\eta = \varepsilon/16$ and $\kappa \geq 64/\varepsilon$, both right-hand sides are at least $56 - 3 > 0$. The same calculation without r gives

$$q - \beta \geq \left(\frac{n-2t-s}{n} - 2\eta\right)\kappa - 2 \geq 56 - 2 > 0.$$

Thus $q \geq c_{\mathrm{cov}} = \zeta + 1$, $h_{\min} \geq \gamma = \beta + 1$, and $\beta < q$. Since $\beta \geq 0$, the relation $h_{\min} \geq \gamma = \beta + 1$ also gives $h_{\min} \geq 1$.

For a committee label ℓ, write C_ℓ for its members and B_ℓ, S_ℓ, R_ℓ for its round's fault sets. Define $\mathcal{E}_{\mathrm{Cross}}$ by the following conditions for every label ℓ and every earlier label $\ell' \prec \ell$:

$$|C_\ell| \leq 2\kappa, \quad |C_\ell \cap B_\ell| \leq \beta, \quad |C_\ell \setminus (B_\ell \cup S_\ell)| \geq q, \tag{6}$$

$$\begin{aligned} |C_\ell \cap (B_\ell \cup R_{\ell'})| &\leq \zeta, \\ |C_\ell \setminus (B_\ell \cup S_\ell \cup R_{\ell'})| &\geq h_{\min}. \end{aligned} \tag{7}$$

These conditions specify the deterministic committee properties required by the primitive. Theorem 12 bounds their probability of failure over the full BA execution. For a single invocation, label its three consecutive rounds $\mathrm{in}, \mathrm{ack}, \mathrm{conf}$.

Theorem 7 (Properties of Π_{PRWGMMO}). *On* $\mathcal{E}_{\mathrm{Cross}}$*, the following properties hold:*

- ***Weak Graded Consistency.*** *Suppose the input sender* P_i *is outside* B_{in} *and a non-Byzantine party* P_j *outputs* $(m_{i,j}, g_{i,j})$ *with* $g_{i,j} = 1$*. Then every non-Byzantine party* P_y *that is receive-reliable in the Confirm round has a pair* $(m_{i,y}, g_{i,y}) \in \mathcal{O}(P_y)$ *with* $m_{i,y} = m_{i,j}$*.*
- ***Validity.*** *If an eligible input sender* P_i *is send-reliable in the Input round and sends* m_i*, then every non-Byzantine party* P_j *that is receive-reliable in the Confirm round has a pair* $(m_{i,j}, g_{i,j}) \in \mathcal{O}(P_j)$ *with* $m_{i,j} = m_i$ *and* $g_{i,j} = 1$*.*
- ***Termination.*** *Every non-Byzantine party terminates after three rounds and outputs a valid set of pairs.*

The proof follows the construction below.

Protocol description Each receiver processes only the first valid direct Input-round message from each input signer. An eligible ACK signer sends a signed $(\mathsf{Report}, \mathcal{ACK})$ message whose list contains at most one valid input from each input signer. A valid signed Confirm message from an eligible confirmer contains at most one valid $\mathcal{ACK}$ list per ACKer. Such a message *rejects* an exact input m if the lists of q distinct ACK signers in the message do not contain m. Eligibility proofs are attached to all signed messages.

Protocol A.1: Π_{PRWGMMO} (PR-Weak Graded Multicast with Mobile Omissions)

1. **Input.** Every eligible input party P_i multicasts its signed input m_i.
2. **ACK.** Every eligible ACKer P_j processes the first valid input received from each input signer in the Input round and ignores subsequent direct variants from that signer. It signs and multicasts $(\mathsf{Report}, \mathcal{ACK}_j)$ listing these inputs.
3. **Confirm.** Every eligible confirmer P_j signs and multicasts $(\mathsf{Confirm}, \mathcal{L}_j)$ containing the first valid $\mathcal{ACK}$ list received from each ACK signer in the ACK round. By the end of Confirm, every party keeps the first valid list per ACK signer that it obtained directly or in a valid Confirm message. It also keeps the first valid Confirm message per confirmer for the rejection condition.

Output. At the end of Confirm, each party P_j constructs $\mathcal{O}(P_j)$ as follows. For each input signer P_i, consider the valid inputs signed by P_i that P_j received during the Input round or obtained from $\mathcal{ACK}$ lists it received. If no such input exists, include no pair for P_i. Otherwise, choose one such input m and include exactly one pair $(m_{i,j}, g_{i,j})$, with $m_{i,j} = m$. Set $g_{i,j} = 1$ if, by the end of Confirm, P_j has valid Confirm messages from at least q distinct eligible confirmers and none of its received Confirm messages rejects m; otherwise, set $g_{i,j} = 0$. Output $\mathcal{O}(P_j)$.

Properties of Π_{PRWGMMO} Throughout this subsubsection, assume that $\mathcal{E}_{\mathrm{Cross}}$ holds.

The implicit NACK rule is the key difference from Π_{WGMMO}: a reporter need not know the identity of every eligible sender.

Lemma 6 (ACK existence). *If a party $P_x \notin B_{\mathrm{conf}}$ satisfies the grade-one predicate for an exact input m at the end of Confirm, some ACK committee member that is send-reliable in the ACK round multicasts a valid $\mathcal{ACK}$ list containing m.*

Proof. At most $\zeta < q$ eligible confirmers lie in $B_{\mathrm{conf}} \cup R_{\mathrm{ack}}$, so at least one of the received Confirm messages was sent by a confirmer outside that union. Suppose that no ACK member that is send-reliable in the ACK round included m in its list. By (6), at least q ACK members are send-reliable in that round and

multicast valid lists that do not contain m. The chosen confirmer receives all these lists and includes them in its Confirm message. This message rejects m, contradicting the grade-one predicate.

Lemma 7 (Validity for a send-reliable eligible sender). *If an eligible input sender P_i is send-reliable in the Input round and sends m, every party P_j that is receive-reliable in the Confirm round outputs $(m_{i,j}, 1) \in \mathcal{O}(P_j)$ with $m_{i,j} = m$.*

Proof. Every ACK member outside B_{ack} that is receive-reliable in the Input round receives m. Thus every valid list not containing m is signed by a member of $B_{\text{ack}} \cup R_{\text{in}}$. By (7), at most $\zeta < q$ ACK members lie in that union, so no valid Confirm message rejects m.

Every party that is receive-reliable in Confirm receives Confirm messages from at least q Confirm committee members that are send-reliable in that round. By (7) and $h_{\min} \geq 1$, there is a Confirm committee member c that is send-reliable in Confirm and receive-reliable in ACK. This confirmer receives a list containing m from an ACK committee member that is send-reliable in ACK and receive-reliable in Input; such an ACK member exists by (7) and $h_{\min} \geq 1$. The ACK member has only one valid list for this invocation. The confirmer receives the list and forwards it in Confirm. Consequently, every party that is receive-reliable in Confirm keeps m by the end of that round. Since m is the input sender's only valid input, each such party selects it, satisfies the grade-one predicate, and outputs m with grade one.

Lemma 8 (Graded outputs of Π_{PRWGMMO}). *Suppose $P_i \notin B_{\text{in}}$ and a party $P_x \notin B_{\text{conf}}$ outputs $(m, 1)$ for P_i at the end of Confirm. Every party $P_j \notin B_{\text{conf}} \cup R_{\text{conf}}$ outputs a pair $(m_{i,j}, g_{i,j}) \in \mathcal{O}(P_j)$ with $m_{i,j} = m$ at the end of Confirm. This statement holds even if P_x or P_j is corrupted later.*

Proof. By Lemma 6, an ACK member P that is send-reliable in the ACK round multicasts a valid ACK list containing m. By (7), there is a Confirm committee member P' that is send-reliable in Confirm and receive-reliable in ACK. The ACK member P has only one valid list for this invocation. Thus the Confirm member P' receives this list and forwards it in Confirm, and every party outside $B_{\text{conf}} \cup R_{\text{conf}}$ receives it as well. Since $P_i \notin B_{\text{in}}$, m is its only valid input for this invocation. Every such receiver selects m and outputs a pair containing it at the end of Confirm, regardless of later corruptions.

Proof (Theorem 7). Weak graded consistency follows from Lemma 8, and validity follows from Lemma 7. Termination follows because the protocol runs for exactly three rounds and then performs the output checks.

A.2 Protocol

The high-level structure of Π_{PRBA} is in Fig. 4.

Every direct committee message includes its signer's eligibility proof.

Status and proposals. A rank-zero certificate $C_0(v)$ contains signed Initialization messages on $v \in \mathcal{V}$ from γ distinct eligible signers. A valid Status message is a signed tuple (Status, k, C) from an eligible signer, where $C = \perp$ or C is a valid certificate of rank below k.

A proposal header a binds the iteration, proposer j, value v, and a proof of the proposer's actual evaluation $u_{j,k} = Y_{j,\ell_k^{\mathrm{prop}}} < D_{\ell_k^{\mathrm{prop}}}$. As in Section 4.2, write $\tau(a) = u_{j,k}$. A valid proposal is a message $\pi = (\mathsf{Propose}, k, a, \mathcal{S})$ signed by the proposer, where $\mathcal{S}$ consists of exactly q valid iteration-k status messages from distinct eligible status signers. Its header satisfies the maximum-rank value rule of Section 4.2, with q in place of q_{mob}.

Vote outputs and rank certificates. Use the vote syntax and minimum-header rule of Section 4. Every input carries its eligibility proof.

The Vote-d call accepts only valid Vote-d inputs for the current iteration. Let $\mathcal{O}_d(P_j) = \{(m_{i,j}, g_{i,j})\}$ denote the final output returned to party P_j by this call. The minimum-header rule is applied to $\{m : (m, g) \in \mathcal{O}_d(P)\}$, whereas emphatic support uses only grade-one outputs for votes their senders label emphatic for the chosen header. Coverage counts distinct input signers represented in $\mathcal{O}_d(P)$, regardless of grade or the sender's support bit. Multiple input variants from one signer never count as multiple signers. For $k > 0$, a valid rank-k certificate $C_k(a)$ contains a and γ signed emphatic Vote-3 messages m_i for a from distinct eligible input signers.

An announcement is a tuple $(\mathsf{Disseminate}, k, \mathsf{NIL})$ or $(\mathsf{Disseminate}, k, C_k(a))$ signed by an eligible signer. Each party maintains a local decision value v_P^{dec}, initialized to $\perp$ before the iterations. Once a party fixes its local decision value, it never changes that value while following the protocol.

Protocol A.2: $\Pi_{\mathsf{PRIter}}(k)$ (one player-replaceable mobile iteration)

1. **Status.** Every eligible member of a fresh status committee multicasts its signed (Status, k, C), where C is its highest-rank certificate below k, or $C = \perp$ if it has no such certificate.
2. **Propose.** Every eligible member P of a fresh proposer committee selects valid Status messages from q distinct eligible signers among those it has received. If messages from fewer than q distinct eligible signers are available, or if the selected collection fails the value rule, it sends nothing. Otherwise, it constructs a according to that rule and multicasts its signed $\pi = (\mathsf{Propose}, k, a, \mathcal{S})$, where $\mathcal{S}$ is the selected collection.
3. **Vote 1.** Every eligible input member P of the fresh Vote-1 committee applies the minimum-header rule to the proposals received in Propose. It inputs the corresponding NIL or split vote; for a single header a, P labels its vote emphatic and inputs $(\mathsf{Vote}_1, k, a, 1, \pi)$. Run one parallel Π_{PRWGMMO} execution on these inputs. Each party P' obtains output $\mathcal{O}_1(P')$.

4. **Vote 2.** Every eligible input member P of the fresh Vote-2 committee applies that rule to $\{m : (m, g) \in \mathcal{O}_1(P)\}$. It inputs the corresponding NIL or split vote, or $(\mathsf{Vote}_2, k, a, c)$ for a single header. Here P sets $c = 1$ exactly when $\mathcal{O}_1(P)$ contains grade-one outputs for emphatic Vote-1 inputs for a from γ distinct input signers; $c = 0$ otherwise. Run Π_{PRWGMMO} on these inputs. Each party P' obtains output $\mathcal{O}_2(P')$.
5. **Vote 3 and certificate.** Every eligible input member P of the fresh Vote-3 committee applies that rule to $\{m : (m, g) \in \mathcal{O}_2(P)\}$. It inputs the corresponding NIL or split vote, or $(\mathsf{Vote}_3, k, a, c)$ for a single header. Here P sets $c = 1$ exactly when $\mathcal{O}_2(P)$ contains grade-one outputs for emphatic Vote-2 inputs for a from γ distinct input signers and $\mathcal{O}_2(P)$ contains outputs for valid votes from c_{cov} distinct input signers; $c = 0$ otherwise. Run Π_{PRWGMMO} on these inputs. Each party P' obtains output $\mathcal{O}_3(P')$ and forms each $C_k(a)$ justified by the grade-one outputs for emphatic Vote-3 messages in $\mathcal{O}_3(P')$.
6. **Certificate dissemination.** Each eligible member of a fresh input committee signs an announcement containing its least valid rank-k certificate, or NIL if it has none. Run Π_{PRWGMMO} on these announcements. Each party P stores every valid rank-k certificate carried by an announcement in its output, regardless of grade. If $v_P^{\mathrm{dec}} = \perp$ and P's grade-one outputs contain announcements from γ distinct eligible input signers, each carrying a valid rank-k certificate with the same header a, then P sets v_P^{dec} to the value in a. No party outputs a BA decision in this step.

Protocol A.3: Π_{PRBA} (complete player-replaceable mobile-omission Byzantine agreement)

1. **Initialize.** Each party P sets $v_P^{\mathrm{dec}} = \perp$. Each eligible initialization member with an input in $\mathcal{V}$ multicasts its signed $\mathsf{Initialization}$ message containing that input; members without an input send no initialization message. Each receiver forms $C_0(v)$ for each $v \in \mathcal{V}$ supported by valid initialization messages from γ distinct eligible signers.
2. **Run iterations.** Execute $\Pi_{\mathsf{PRIter}}(k)$ for $k = 1, \ldots, \lambda$, completing every scheduled round.
3. **Finalize.** Use a fresh label for finalizer eligibility. Each eligible party P with $v_P^{\mathrm{dec}} \neq \perp$ multicasts its signed $(\mathsf{Finalize}, v_P^{\mathrm{dec}})$ with its finalizer eligibility proof. At the end of the round, every party outputs (v, False) if it has received valid finalization messages for some $v \in \mathcal{V}$ from γ distinct eligible finalizers. If more than one value qualifies, it chooses the least; if none qualifies, it outputs $(\perp, \mathsf{True})$. The supporting local decisions may be based on certificates of different ranks. This round defines S_{fin} and R_{fin}.

A.3 Safety

Let $B_d(k)$ be the Byzantine set in the input round of the Vote-d Π_{PRWGMMO} invocation, and let $R_{\mathrm{conf},d}(k)$ be the receive-omission set in that invocation's Confirm round. Write $R_{\mathrm{conf,ann}}(k)$ for the receive-omission set in the announcement Confirm round, and omit k when considering a fixed iteration. Throughout this subsection, assume that $\mathcal{E}_{\mathrm{Cross}}$ holds. The initialization event needed for Validity is defined below.

Lemma 9 (Origin of emphatic votes). *If an eligible sender outside $B_d(k)$ sends an emphatic phase-d vote for a, a valid proposal with header a was sent in that iteration.*

Proof. For $d = 1$, the vote carries the valid proposal. For $d > 1$, an emphatic vote requires grade-one outputs for emphatic votes in the preceding phase from γ distinct eligible input signers. At most $\beta < \gamma$ of these signers lie in $B_{d-1}(k)$. A signer outside that set sent the corresponding emphatic vote, so the induction hypothesis yields an emphatic Vote_1 message carrying the proposal.

Lemma 10 (Vote-2 comparison). *Suppose an eligible Vote-2 sender outside B_2 sends an emphatic Vote_2 message for a. Every eligible Vote-2 sender outside B_2 that is receive-reliable in the Vote-1 Confirm round sends a non-NIL Vote_2 message whose evaluation is at most $\tau(a)$. At equality, a single-header message has header a.*

Proof. The sender of the emphatic vote has grade-one outputs for emphatic votes for a from γ distinct input signers in its final $\mathcal{O}_1$. Since this sender is outside B_2, it followed the protocol throughout the invocation. At most $\beta < \gamma$ of these Vote_1 input signers lie in B_1. Choose a signer $H \notin B_1$ among them. The Vote-2 sender, as a Vote-1 receiver, assigned grade one to H's exact input while outside the Vote-1 invocation's B_{conf}, since $B_{\mathrm{conf}} \subseteq B_2$. By Lemma 8, every eligible Vote-2 sender outside B_2 that is receive-reliable in the Vote-1 Confirm round has this input in its final $\mathcal{O}_1$ when it begins its Vote-2 action. The minimum-header rule gives a non-NIL Vote_2 message whose headers have evaluation at most $\tau(a)$. If equality holds, an existing split or a distinct header at the minimum forces a split vote. In this equality case, a single-header vote must name a.

Theorem 8 (One header per positive rank). *For every $k > 0$, any two valid rank-k certificates have the same header and value.*

Proof. Each certificate contains emphatic Vote_3 messages from $\gamma = \beta+1$ distinct eligible input signers. Choose a signer outside B_3 from each certificate, say P for a and P' for a'. The grade-one outputs for emphatic Vote-2 inputs supporting P include an input signed by $W \notin B_2$. Applying Lemma 10 to W shows that every eligible Vote-2 sender outside B_2 that is receive-reliable in the Vote-1 Confirm round sends a non-NIL Vote_2 message whose headers have evaluation at most $\tau(a)$. If that message has a single header and equality holds, its header is a.

Coverage at P' ensures that its final $\mathcal{O}_2(P')$ contains outputs for votes from $c_{\text{cov}} = \zeta + 1$ distinct eligible input signers. At most ζ of these signers lie in $B_2 \cup R_{\text{conf},1}$, so one of the messages comes from a signer X satisfying the comparison property above. Since P' chose a' by minimum evaluation, $\tau(a') \leq \tau(a)$. By symmetry, equality holds. At this common evaluation, X cannot have sent a split vote, as this would have forced P' to send a split vote as well. Its single-header message names a. If $a \neq a'$, the two headers would tie at the minimum in $\{m : (m, g) \in \mathcal{O}_2(P')\}$, again forcing a split. Thus $a = a'$ and the values agree.

Lemma 11 (Certificate dissemination). *Suppose that, at the end of an announcement invocation's Confirm round, a party outside B_{conf} has grade-one outputs for announcements from γ distinct eligible input signers, each carrying a valid rank-k certificate with the same header a. Every non-Byzantine party that is receive-reliable in the announcement Confirm round of iteration k stores a valid rank-k certificate for a at the end of that invocation. For any later round u, every party outside B_u that is receive-reliable in the announcement Confirm round of iteration k has such a certificate when round u begins.*

Proof. At most $\beta < \gamma$ of these input signers lie in the announcement invocation's set B_{in}. Choose a signer outside this set. The party in the premise outputs this announcement with grade one while outside B_{conf}. By Lemma 8, every party outside $B_{\text{conf}} \cup R_{\text{conf}}$ outputs it and stores its rank-k certificate for a. A party outside a later B_u was also outside B_{conf} and, if receive-reliable in the announcement Confirm round, keeps the certificate until round u.

Theorem 9 (Consistency of local decisions and final outputs). *All non-$\perp$ values assigned to v_P^{dec} before P's first Byzantine corruption are equal. Consequently, all unflagged outputs of non-Byzantine parties have that value.*

Proof. If no such assignment occurs, no finalization threshold can be met: because $\gamma > \beta$, any qualifying collection of signatures would include a signer outside B_{fin}. Such a signer signs only after establishing a local decision value while following the protocol. Otherwise, choose an earliest such assignment. Suppose it occurs in iteration k, after the party obtains grade-one outputs for announcements from γ distinct eligible input signers, each carrying a valid rank-k certificate with the same header a and value v. For a later status committee in iteration $k' > k$, at most $\zeta < q$ eligible members are in $B_{\text{status},k'} \cup R_{\text{conf,ann}}(k)$. Every valid q-status proposal includes a Status message whose sender is outside this union. By Lemma 11, that sender has a valid rank-k certificate for a, so its later Status message includes this certificate or one of higher rank. This conclusion does not require the party that made the earliest local decision to remain uncorrupted.

We now induct on the certificate rank. The base case at rank k follows from Theorem 8. A rank-ℓ certificate for $\ell > k$ contains γ emphatic Vote_3 messages, at least one of which has a signer outside $B_3(\ell)$. Lemma 9 guarantees a valid proposal for its header. The proposal's Status collection includes a message from a holder as described above. By the induction hypothesis, every certificate at a

rank in $k, \ldots, \ell - 1$ has value v, so the maximum-rank rule assigns v to both the proposal and the new rank certificate. Every such non-$\perp$ assignment to a local decision value is triggered by grade-one outputs for announcements from γ distinct eligible input signers, carrying rank-ℓ certificates with the same header, for some $\ell \geq k$. Thus every such assignment uses value v.

Finally, among any γ valid signatures from distinct eligible finalizers, at least one comes from a signer outside B_{fin}. That signer's finalization message was sent as prescribed by the protocol and includes its non-$\perp$ local decision value. This value is v, so every qualifying collection of finalization messages, and hence every unflagged output, has value v.

Initialization event. For validity, fix $v \in \mathcal{V}$, let $D_v = \{P_i : v_i \notin \{v, \perp\}\}$, and let init denote the initialization round. Recall $U = \{P_i : v_i = \perp\}$. Define

$$\begin{aligned} \mathcal{E}_{\text{Init}}(v) : \quad & |C_{\text{init}} \cap (B_{\text{init}} \cup D_v)| \leq \beta, \\ & |C_{\text{init}} \setminus (B_{\text{init}} \cup D_v \cup U \cup S_{\text{init}})| \geq h_{\min}. \end{aligned} \tag{8}$$

The premise of Validity is equivalent to $D_v \subseteq B$ and $|U \setminus B| \leq r$. On $\mathcal{E}_{\text{Init}}(v)$, at most $\beta < \gamma$ eligible signers can sign a value other than v, while at least $h_{\min} \geq \gamma$ eligible senders reliably multicast v. Thus no rank-zero certificate for a different value exists.

Theorem 10 (Validity). *If $\mathcal{E}_{\text{Init}}(v)$ holds for $v \in \mathcal{V}$, every non-$\perp$ value assigned to v_P^{dec} before P's first Byzantine corruption and every unflagged output of a non-Byzantine party equals v.*

Proof. The initialization event allows at most $\beta < \gamma$ eligible signers to sign values other than v and ensures that at least $h_{\min} \geq \gamma$ eligible senders of v are send-reliable in the initialization round and outside $D_v \cup U$. Parties outside B_{init} that have no input do not sign any initialization value. Thus no rank-zero certificate for another value exists. Any party outside a later Byzantine set that was receive-reliable in the initialization round received these signed Initialization messages and obtains a γ-signature certificate $C_0(v)$.

For every later Status committee, fewer than q eligible signers lie in the union of its current Byzantine set and R_{init}. Every valid proposal includes a Status message from a party with $C_0(v)$ or a higher-rank certificate. For any positive-rank certificate, one of its γ Vote_3 signers is outside the corresponding Byzantine set, so Lemma 9 guarantees a valid proposal with that header. Induction on the rank, starting at rank zero, shows that all such proposals and certificates have value v. Every prescribed assignment to v_P^{dec} uses such a certificate. Any collection of messages meeting the finalization threshold includes a message from a signer outside B_{fin}, whose signed value is v.

A.4 Progress, Strong NLU, and probability

The progress and finalization results below assume that $\mathcal{E}_{\text{Cross}}$ holds. We then bound the probability of the required events and prove overall correctness.

An iteration is *good* if its proposer committee is nonempty, the minimum evaluation is unique, and the proposer attaining this minimum is send-reliable in Propose and receive-reliable in Status.

Lemma 12 (Progress in a good iteration). *At the end of a good iteration, every non-Byzantine party that is receive-reliable in the announcement Confirm round has a non-$\perp$ local decision value, and all these values agree.*

Proof. At least q status committee members are send-reliable in the Status round. The good proposer receives these Status messages and sends a valid minimum-evaluation proposal.

At least $h_{\min} \geq \gamma$ eligible Vote-1 input senders are send-reliable in the Vote-1 Input round and receive-reliable in the Propose round. Each inputs an emphatic Vote_1 message for this proposal. By Lemma 7, every subsequent Vote-2 sender outside B_2 that is receive-reliable in the Vote-1 Confirm round has all these emphatic inputs with grade one in its Vote-1 output. Each such sender selects the same header and satisfies the condition for sending an emphatic Vote_2.

At least $h_{\min} \geq \gamma$ of these Vote-2 senders are send-reliable in that phase's Input round. By the same argument, every subsequent Vote-3 sender outside B_3 that is receive-reliable in the Vote-2 Confirm round obtains grade-one outputs for their emphatic inputs. In addition, at least q eligible Vote-2 senders are send-reliable in that phase's Input round. Each sends one valid Vote_2 message, whether emphatic or not. Every such Vote-3 receiver has outputs for inputs from $q \geq c_{\mathrm{cov}}$ distinct signers, satisfying the coverage requirement. At least $h_{\min} \geq \gamma$ of these Vote-3 senders are send-reliable in that phase's Input round and send emphatic Vote_3 messages for the selected header.

By Lemma 7, every subsequent announcement sender that is outside its current Byzantine set and receive-reliable in the Vote-3 Confirm round outputs those Vote_3 messages with grade one. Such a sender can form a rank certificate, whose header is unique by Theorem 8. At least $h_{\min} \geq \gamma$ such announcement input senders are send-reliable in the announcement Input round. Validity gives grade-one outputs for all their non-NIL announcements to every non-Byzantine party that is receive-reliable in the announcement Confirm round. Their certificates have the same header by Theorem 8, so every such party meets the γ-announcement condition. If v_P^{dec} is still $\perp$, the party sets it to the value in that header; otherwise, it keeps its earlier local decision value. By Theorem 9, the local decision value is the same in either case.

Theorem 11 (Strong-NLU finalization). *If at least one iteration is good, every party that is receive-reliable in the final round outputs the common decision value. Among non-Byzantine parties, only those in R_{fin} may flag.*

Proof. Fix a good iteration $k^\star$. Its progress proof identifies at least γ distinct announcement input senders that are send-reliable in the Input round and announce valid rank-$k^\star$ certificates with the same header. The finalizer committee has at least $h_{\min} \geq \gamma$ members that are send-reliable in the final round and receive-reliable in the announcement Confirm round of iteration $k^\star$. By

Lemma 7, applied to each of these announcement inputs, every such finalizer received grade-one outputs for all of them at the end of the announcement invocation. These outputs satisfy the γ-announcement condition, so by the end of the invocation the finalizer has a non-$\perp$ value for v_P^{dec}, which it keeps. By Theorem 9, all these senders multicast signed finalization messages with the same value. Every party that is receive-reliable in the final round receives at least γ of these messages from distinct eligible finalizers and outputs that value unflagged. Any other collection of messages meeting the finalization threshold supports the same value.

Global committee concentration The global execution has $M = 14\lambda + 2$ committee rounds. For a fixed round label ℓ, let $X_\ell = |C_\ell|$. The ideal VRF sampling of Section 3.4 gives $X_\ell \sim \mathrm{Bin}(n, \kappa/n)$ and $\mathbb{E}[X_\ell] = \kappa$. A Chernoff bound and a union bound give

$$\Pr[\exists \ell :\ |C_\ell| > 2\kappa] \le M(e/4)^\kappa \le Me^{-\kappa/3}. \tag{9}$$

Theorem 12 (Global committee concentration). *For the λ-iteration protocol Π_{PRBA}, under the resilience condition $2t + s + r \le (1 - \varepsilon)n$, we have*

$$\Pr[\neg \mathcal{E}_{\mathrm{Cross}}] \le (14\lambda + 2)(14\lambda + 4)e^{-\eta^2\kappa/3}. \tag{10}$$

For every fixed value v,

$$\begin{aligned} &\Pr[\neg \mathcal{E}_{\mathrm{Cross}} \ \cup\ (\{D_v \subseteq B,\ |U \setminus B| \le r\} \cap \neg \mathcal{E}_{\mathrm{Init}}(v))] \\ &\quad \le \big((14\lambda + 2)(14\lambda + 4) + 2\big)e^{-\eta^2\kappa/3}. \end{aligned} \tag{11}$$

Proof. For the current label, each party's membership in $B_\ell \cup S_\ell$ is determined before its sampling outcome is revealed. For $\ell' \prec \ell$, $R_{\ell'}$ is fully determined before the adversary learns any non-Byzantine party's eligibility for the current label. Thus the fault sets and unions appearing in (6)–(7) has size at most t, $t+r$, $t+s$, or $t + s + r$, as appropriate. The Chernoff bound gives a failure probability of at most $e^{-\eta^2\kappa/3}$ for each inequality. The thresholds in (6) and (7) are separated from the corresponding binomial means by a margin of at least $\eta\kappa$.

If $|D_v| + \max\{0, |U| - r\} > t$, the premise $D_v \subseteq B$ and $|U \setminus B| \le r$ is impossible, since $D_v \cap U = \emptyset$. Otherwise, under that premise, $|B_{\mathrm{init}} \cup D_v| \le t$ and $|B_{\mathrm{init}} \cup D_v \cup U \cup S_{\mathrm{init}}| \le t + r + s$. The initial sets D_v and U are fixed before committee sampling. The same ideal-sampling bound gives the two initialization inequalities in (8), with thresholds β and $h_{\min}$, each with failure probability at most $e^{-\eta^2\kappa/3}$.

Across the $14\lambda + 2$ committee labels, there are at most $(14\lambda + 2)(14\lambda + 4)$ committee-size and fault-intersection events. A union bound gives (10). Including the two additional initialization events in (8) gives (11).

Good proposer probability Set

$$q_{\mathrm{bad}} = 1 - \frac{n - t - s - r}{n}\left(1 - \left(1 - \frac{\kappa}{n}\right)^n\right). \tag{12}$$

Condition on the execution history preceding iteration k. Accounting for the possibility that the proposer committee is empty gives

$$\Pr[\text{iteration } k \text{ is good}] \geq \left(1 - \left(1 - \frac{\kappa}{n}\right)^n\right) \frac{n - t - s - r}{n}.$$

Thus the conditional probability that the iteration is bad is at most q_{bad}, as defined in (12). Fresh labels and the chain rule give $\Pr[\text{all } \lambda \text{ iterations are bad}] \leq q_{\mathrm{bad}}^{\lambda}$.

Correctness

Theorem 13 (Correctness of Π_{PRBA}). *Assume $0 < \varepsilon < 1$, $\lambda \geq 1$, $2t+s+r \leq (1-\varepsilon)n$, and $64/\varepsilon \leq \kappa \leq n$. Consider the network and adversary of Section 3 with the ideal cryptographic primitives of Section 3.4. Set*

$$\delta_{\mathrm{PRM}} := \big((14\lambda + 2)(14\lambda + 4) + 2\big)e^{-\eta^2\kappa/3} + q_{\mathrm{bad}}^{\lambda}.$$

Then Termination holds unconditionally. Except with probability at most δ_{PRM}, Agreement, Validity, and Strong NLU hold in the same execution. On the same event, every party that is receive-reliable in the final round outputs the common value without flagging. Hence Π_{PRBA} is a secure strong-NLU Byzantine agreement protocol in the idealized cryptographic model.

Proof. Theorem 12 gives the first term in the ideal-sampling error bound. For a fixed initial input assignment, $n > 2t + r$ implies that at most one value v can satisfy $D_v \subseteq B$ and $|U \setminus B| \leq r$ for some B that is a subset of $\mathcal{P}$ with $|B| \leq t$. Indeed, if two distinct values satisfied these conditions for sets B and B', all parties outside $B \cup B'$ would have no input. This would leave more than r parties without an input outside B. Thus the joint initialization bound covers the premise of Validity without a union bound over values. On the cross event and, when applicable, the initialization event, Theorems 8, 9, and 10 establish deterministic safety. The good-proposer bound ensures at least one good iteration except with probability at most $q_{\mathrm{bad}}^{\lambda}$. Lemma 12 and Theorem 11 then establish Strong NLU. Termination is unconditional after λ iterations, or $14\lambda+2$ synchronous rounds.

A.5 Rounds and communication

Theorem 14 (Complexity of Π_{PRBA}). *With the parameters and assumptions of Theorem 13, assume that every value in $\mathcal{V}$ has an encoding of $O(\kappa)$ bits. The protocol Π_{PRBA} uses exactly $14\lambda + 2$ synchronous rounds. On $\mathcal{E}_{\mathrm{Cross}}$, non-Byzantine parties send $O(n\lambda\kappa^6)$ bits, which is subquadratic when $\lambda\kappa^6 = o(n)$. For constant ε and $\kappa, \lambda = \Theta(\log n)$, this gives $O(n \log^7 n)$ bits.*

Proof. We count only communication by non-Byzantine parties. Initialization and finalization use one round each, and every iteration has two direct rounds

and four three-round graded phases. The total is $14\lambda + 2$ rounds. On $\mathcal{E}_{\text{Cross}}$, every sending committee has at most 2κ members, each multicasting at most one message to n receivers. Thus the protocol uses at most $2(14\lambda + 2)n\kappa$ point-to-point transmissions.

By the value-length assumption, a valid signed initialization message, proposal header, or Vote_3 input has $O(\kappa)$ bits. A rank certificate contains $\gamma = O(\kappa)$ signed inputs, so a certificate, Status message, or announcement has $O(\kappa^2)$ bits. A proposal contains $q = O(\kappa)$ Status messages and has $O(\kappa^3)$ bits. Consequently, every valid graded input, including an emphatic Vote_1 input, has at most $O(\kappa^3)$ bits. An ACK list contains at most one input per eligible input signer, and hence at most 2κ inputs. Its size is $O(\kappa^4)$ bits. A $\mathsf{Confirm}$ message contains at most one ACK list per eligible ACK signer, and hence at most 2κ lists. Its size is $O(\kappa^5)$ bits. Eligibility proofs and signatures are included in these bounds. With at most $2n\kappa$ transmissions per round and a constant number of rounds per iteration, all messages in an iteration together use $O(n\kappa^6)$ bits. Initialization and finalization each use $O(n\kappa^2)$ bits. For $\lambda \geq 1$, the total is $O(n\lambda\kappa^6)$ bits, or $O(n\lambda\kappa^5)$ cryptographic words. This is subquadratic when $\lambda\kappa^6 = o(n)$. Substituting $\kappa, \lambda = \Theta(\log n)$ gives $O(n \log^7 n)$ bits and $O(n \log^2 n)$ transmissions.

B Round–error tradeoff

We sketch an adaptation of the indistinguishability argument of Abraham and Stern [4] to omissions selected before sending. Let $n \geq 2$, $t = r = 0$, and $s = 1$, and consider a binary BA protocol with input-independent setup and a fixed upper bound of c rounds. Let δ be its worst-case probability of failing to reach valid, unflagged agreement by round c. Since $r = 0$, a flagged output violates Strong NLU.

Proof sketch. For the analysis, fix the same random tapes, setup, and ideal-primitive randomness in all executions. Following [4], consider two configurations that differ only in one party's local state. Their next-round configurations can be connected in at most $2n - 1$ steps, with only one party's state changing at each step: suppress the differing party's deliveries to the other $n - 1$ parties one receiver at a time, switch between the two configurations with all these deliveries suppressed, and then restore them one receiver at a time. The construction fixes which sender to suppress and the receiver order independently of the random choices. Thus, all omission schedules can be fixed before execution, and each affects at most one sender per round.

Start with the all-zero input assignment and change one input at a time until reaching the all-one assignment. Applying the construction for c rounds gives a sequence of at most $n(2n-1)^c+1$ executions, where consecutive executions differ in at most one party's final local state. If every execution succeeded, consecutive executions would have the same decision, since they share another party's output. However, validity requires the all-zero and all-one executions to decide differently. Hence, for every choice of randomness, at least one execution in this fixed family

fails. Averaging over the family shows that there is a fixed input assignment together with a fixed omission schedule whose failure probability is at least $1/[n(2n-1)^c + 1]$.