Early-Stopping Dishonest Majority Broadcast with Optimal Round Complexity

Renas Bacho[1] and Kecheng Shi[2]

[1] Ruhr University Bochum
[2] CISPA Helmholtz Center for Information Security

Abstract. Early-stopping broadcast protocols guarantee termination as a function of the number $f \leq t$ of parties actually corrupted in an execution, rather than of the worst-case bound $t < n$. Loss and Nielsen (Eurocrypt 2024) gave such a protocol for every $t < n$, terminating within $O(\min\{f^2 + 1, t + 1\})$ rounds, with polynomial communication and from a plain public-key infrastructure (PKI). In the other direction, Choi, Choi, Liu, and Zikas (ePrint 2026/1878) recently proved a stopping lower bound which implies that $f + 3$ rounds are necessary whenever $n + f < 2\min\{t, n - 2\}$, a condition that covers a broad part of the dishonest-majority regime. It has remained open whether this bound can be attained, even at the cost of exponential communication.

In this work, we present a deterministic early-stopping broadcast protocol in which every honest party outputs and terminates within $\min\{f + 3, t + 1\}$ rounds for every $t < n$. The protocol is built from exponential information gathering (EIG) graphs and incurs exponential communication. It works in the plain PKI model with idealized signatures and is secure against a strongly adaptive adversary. Its stopping round complexity is therefore optimal throughout the stated range of the lower bound.

1 Introduction

In Byzantine broadcast [24], a designated sender holds an input that it wishes to share with n parties, up to t of which may behave maliciously. A broadcast protocol must satisfy *validity*: if the sender is honest, every honest party outputs its input; and *agreement*: all honest parties output the same value. Broadcast is a fundamental building block of many cryptographic and distributed protocols, including multiparty computation and verifiable secret sharing. A central efficiency measure is its round complexity: how many rounds are needed before every honest party has output and terminated? Dolev and Strong [16] gave a broadcast protocol that terminates in $t + 1$ rounds, which matches the classical worst-case lower bound for $t \leq n - 2$. That bound, however, does not capture executions with fewer than t actual corruptions. When the actual number is $f < t$, the classical early-stopping lower bound of Dolev, Reischuk, and Strong [15] is $\min\{n - 1, t + 1, f + 2\}$ rounds. The gap between this bound and the $(t + 1)$-round protocol has motivated a long line of work on protocols whose running time depends on f.

For optimal resilience $t < n$, Loss and Nielsen [26] gave the first early-stopping protocol, terminating in $O(\min\{f^2+1, t+1\})$ rounds. Deligios, Klasovitá, and Liu-Zhang [12] subsequently obtained an $O(f+1)$-round protocol for $t < (1-\epsilon)n$, where $\epsilon > 0$ is a constant. Both protocols have polynomial communication complexity and work in the plain public-key infrastructure (PKI) model. In the other direction, Choi, Choi, Liu, and Zikas [9] strengthened the classical early-stopping lower bound for a dishonest majority, counting the final send in the stopping round: their result implies that $f + 3$ rounds are necessary whenever $n + f < 2\min\{t, n - 2\}$. This leaves a fundamental open question:

Can broadcast with full resilience $t < n$ achieve optimal early-stopping round complexity, even at the cost of exponential communication?

1.1 Our Results

We present a deterministic binary broadcast protocol[3] in which every honest party outputs and terminates within $\min\{f + 3, t + 1\}$ rounds for any $t < n$. The protocol uses exponential information gathering (EIG)

[3] To broadcast a longer message, run one instance per bit in parallel and concatenate the outputs. This preserves the round bound and multiplies the exponential communication cost by the message length.

graphs and incurs exponential communication. It works in the plain PKI model with idealized signatures and is secure against a strongly adaptive adversary.

Whenever $n + f < 2\min\{t, n-2\}$, our protocol terminates within $\min\{f+3, t+1\} = f+3$ rounds. The lower bound of Choi et al. [9] shows that this stopping guarantee is optimal.

1.2 Related Work

Table 1 compares our protocol with selected deterministic protocols. We discuss their round bounds, as well as related results on communication, fault models, and randomized broadcast.

Table 1. Comparison of deterministic Byzantine agreement and broadcast protocols.

Protocol	Resilience	Rounds	Communication	Setup
Berman et al. [7]	$t < n/3$	$\min\{f+2, t+1\}$	exponential	none
Abraham and Dolev [1]	$t < n/3$	$\min\{f+2, t+1\}$	polynomial	none
Elsheimy et al. [17]	$t < n/2$	$(1+\epsilon)f + O(1)$	polynomial	plain PKI
Choi et al. [9]	$t < n/2$	$\min\{f+2, t+1\}$	exponential	plain PKI*
Deligios et al. [11]	$t < n/2$	$\min\{f+2, t+1\}$	exponential	plain PKI*
Dolev and Strong [16]	$t < n$	$t+1$	polynomial	plain PKI
Loss and Nielsen [26]	$t < n$	$O(\min\{f^2+1, t+1\})$	polynomial	plain PKI
Deligios et al. [12]	$t < (1-\epsilon)n$	$O(f+1)$	polynomial	plain PKI
Deligios et al. [11]	$t < n$	$9f + 7$ for $f \geq 1$	exponential	plain PKI*
Our protocol	$t < n$	$\mathbf{\min\{f+3, t+1\}}$	exponential	plain PKI*

Here t denotes the maximum number of corruptions tolerated, and f denotes the number of parties actually corrupted in an execution. For general $t < n$, the protocol of [12] matches the asymptotic round bound of Loss and Nielsen [26]. An asterisk indicates that the protocol is analyzed in the plain PKI model with ideal transferable signatures. Choi et al. [9] also describe a concrete information-theoretic instantiation of the signature primitive that incurs exponential setup cost. We note that the work of Deligios et al. [11] is concurrent with ours.

Classical round bounds. Pease, Shostak, and Lamport [27,24] introduced Byzantine agreement and broadcast. Their recursive exchange of reports is the basis of exponential information gathering (EIG). Fischer and Lynch [19] proved a worst-case round lower bound of $t+1$ without authentication. Dolev and Strong [16] extended the lower bound to authenticated broadcast and gave a polynomial-communication protocol with a fixed running time of $t+1$ rounds. Dolev, Reischuk, and Strong [15] distinguished simultaneous agreement, in which honest parties output in the same round, from eventual agreement, in which they may output in different rounds. For $t < n-1$, they showed that simultaneous agreement can require $t+1$ rounds even without faults, and that eventual agreement requires $\min\{f+2, t+1\}$ rounds in the worst case with f faults. Their round-complexity measure includes the final round in which an honest party sends a message.

Early stopping without signatures. Toueg, Perry, and Srikanth [30] built an early-stopping agreement protocol using a broadcast primitive that simulates authentication. Bar-Noy et al. [6] studied how switching between agreement algorithms during an execution can improve the trade-off between rounds and message length. Coan [10] used fault diagnosis to attain $\min\{f+2, t+1\}$ rounds with polynomial communication and a stronger resilience restriction than $t < n/3$. Berman, Garay, and Perry [7] attained the same round bound for all $t < n/3$ using EIG, at the cost of exponential communication. Garay and Moses [21] gave a polynomial $t+1$-round protocol and an early-stopping variant for the same threshold. Abraham and Dolev [1] then achieved $\min\{f+2, t+1\}$ rounds for $t < n/3$ with polynomial communication and computation.

Authenticated protocols with an honest majority. For $t < n/2$, Elsheimy, Loss, and Papamanthou [17] improved on the $\min\{2f + 4, 2t + 2\}$-round protocol of Perry and Toueg [28], obtaining a deterministic round bound of $(1 + \epsilon)f + O(1)$ for any fixed $\epsilon > 0$. The same authors subsequently achieved $f + O(\sqrt{f})$ rounds with polynomial communication for $t < n/2$ [18]. They also gave an EIG broadcast protocol in this setting that stops within $\min\{f + 3, t + 1\}$ rounds with exponential communication [18, Section 4], one round past the classical bound; it is the honest-majority precursor of our construction, and it is not the concurrent protocol of [11]. Concurrent with our work, Deligios et al. [11] attained the classical target $\min\{f + 2, t + 1\}$ for authenticated binary broadcast with $t < n/2$ using EIG, at the cost of exponential communication.

Authenticated protocols with a dishonest majority. Loss and Nielsen [26] gave the first early-stopping broadcast protocol tolerating any $t < n$. Their protocol uses *polarisers*, which allow an honest party to convince others that a party is faulty even when the original evidence was a missing message. The protocol has polynomial communication complexity and terminates in $O(\min\{f^2 + 1, t + 1\})$ rounds. Deligios, Klasovitá, and Liu-Zhang [12] use a graph-based version of polarisers to achieve $O(f + 1)$ rounds when $t < (1 - \epsilon)n$ for a fixed $\epsilon > 0$. Their protocol tolerates all $t < n$, with the same general asymptotic round bound as Loss and Nielsen.

In concurrent work, Deligios et al. [11] also give an exponential-communication broadcast protocol for every $t < n$ that terminates within $9f + 7$ rounds for $f \geq 1$. It uses signed confirmations to identify longer message paths that convey the same information as shorter paths. In our protocol, a receiver uses the faults it has detected locally to decide when to stop. Agreement follows by comparing the round-2 reports that honest receivers reconstruct in their respective decision rounds. This gives a stopping bound of $\min\{f + 3, t + 1\}$ for binary broadcast for every $t < n$, also with exponential communication. The lower bound of Choi et al. [9] implies that this stopping bound is optimal whenever $n + f < 2\min\{t, n - 2\}$.

Communication and early decision. Dolev and Reischuk [14] proved lower bounds on the number of messages and signatures in Byzantine agreement. Dolev and Lenzen [13] showed that requiring decision within $f + 2$ rounds in their Byzantine consensus model can force $\Omega(nt + t^2 f)$ messages. Lenzen and Sheikholeslami [25] gave a recursive phase-king protocol for $t < n/3$ with $O(f + 1)$ rounds and quadratic bit complexity. These works concern the cost of agreement under their respective round and resilience guarantees.

Fault models and stopping guarantees. Keidar and Rajsbaum [23] give an $f + 2$-round lower bound on decision for uniform consensus with crash failures, for $f \leq t - 2$. Uniform consensus also requires agreement with faulty parties that decide. Alpturer, Halpern, and van der Meyden [5] study optimal decision times for eventual agreement under sending omissions. Choi, Choi, Liu, and Zikas [9] study stopping lower bounds and distinguish final send-only rounds from rounds that require both sending and receiving. Our stopping bound includes a party's final send, even when it terminates before the receive stage of that round.

A different measure is *good-case latency*, which asks how quickly honest parties can deliver a value when the sender is honest. Abraham et al. [2] characterize this latency for authenticated broadcast across several timing models. Albouy et al. [3] give a deterministic synchronous reliable broadcast protocol with honest-sender latency $\max\{2, t + 3 - c\}$ rounds, where c is the number of correct parties. Hou and Yu [22] add an optimistic track to Byzantine broadcast protocols to improve performance in executions with few faults. Our stopping bound applies to all executions with f corruptions, including those with a faulty sender.

Randomized broadcast. Garay et al. [20] showed how randomization yields expected constant rounds when $t = n/2 + O(1)$. With trusted setup and against weakly adaptive adversaries (without after-the-fact message removal), Chan, Pass, and Shi [8] obtained sublinear rounds for dishonest-majority broadcast, and Wan et al. [31] obtained expected constant rounds when a fixed positive fraction of the parties remains honest. Srinivasan et al. [29] used time-lock puzzles to extend expected constant rounds in this regime to strongly adaptive corruptions. Alexandru et al. [4] obtained sublinear rounds for $t \leq (1 - \epsilon)n$, where $\epsilon > 0$ is constant, using a plain PKI and a uniform reference string, without a trusted dealer. These randomized bounds concern expected latency or a round bound with a prescribed failure probability; we study deterministic stopping as a function of f.

2 Technical Overview

We give an overview of our main techniques.

2.1 From EIG to Root Variables

Our starting point is authenticated exponential information gathering (EIG). In each forwarding round, a receiver signs a view of the messages it received in the preceding round and sends that view to every other receiver. Signatures are transferable, so a receiver passes on not only what it was told but also the evidence that it was told exactly that. Over several rounds this produces, for every early message, a collection of signed copies that the receivers can place side by side and compare.

The central difficulty is that a Byzantine sender may withhold its round-1 message from some or all receivers. A receiver that received a signed message can forward it, but a receiver that received nothing has nothing to forward, since silence leaves no trace that can be signed. We therefore do not forward the sender's message itself. We ask the sender to sign bit 1 and send the signature to every receiver if its input is 1, and to stay silent if its input is 0. Each receiver then turns its own round-1 observation, silence included, into a signed message of its own. In round 2, receiver P_i sends I_i, with the sender's signature on bit 1 attached, if it received that signature directly in round 1, and sends O_i otherwise. We call these two messages the *reports* of P_i and say that they have the *values* I and O. Output 1 requires the positive evidence carried by some report I_i, whereas output 0 is what the receivers fall back on in its absence.

A report is signed by its sender, so it is transferable in turn, and the forwarding rounds from round 3 onwards carry signed copies of the reports instead of the sender's message. For each receiver P_i we introduce a variable root_i for P_i's round-2 report. We call it a *root* because the report is the first message of the structure of signed copies that the later rounds build around it: the round-3 views carry copies of the report, the round-4 views carry copies of those views, and so on. There are $n-1$ roots, one per receiver, and all of them speak about the same round-1 behaviour of the sender. The two reports of P_i carry different values under the same root label, so a receiver holding two copies of that label with different values learns that P_i signed both and must be faulty. Figure 1 shows how the signed views for one root are forwarded.

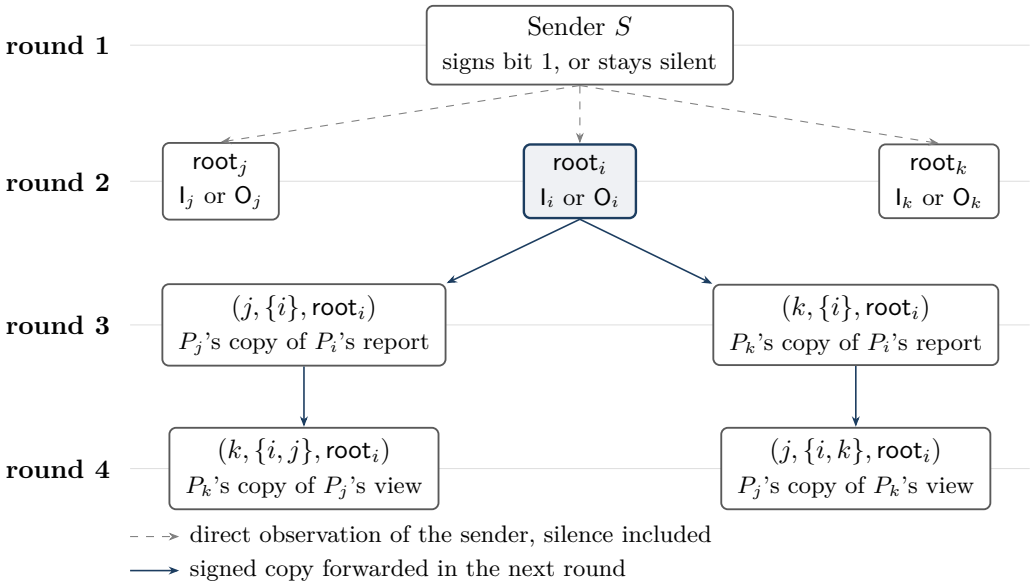

Fig. 1. One root in authenticated EIG. In round 1 each receiver observes the sender directly, and silence counts as an observation. In round 2 the receiver turns that observation into its own signed report, which is the root root belonging to it. From round 3 onwards the receivers forward signed copies, and every copy keeps the signatures nested inside it. Each box in rounds 3 and 4 is named by its view label (e, A, root_i): the signer P_e, the set A of receivers whose views are already nested inside, and the root. The figure follows root_i only; the remaining roots carry the same structure. The two round-3 views can carry different reports if P_i signs a different report for each receiver.

A receiver that has completed the receive stage of round 2 can use its state to *resolve* each root, by which we mean that it computes a single value for that root from the copies and views it holds. The protocol runs

the resolver once only, in the first round that meets the decision test of Section 2.5; the earlier rounds are there for the analysis rather than for the receiver. We call that value the *resolution* of the root, and we speak of the resolution in a given round, since the state grows from one round to the next and the value may change with it. A resolution is I, O, or $\perp$, where $\perp$ records that the receiver's state does not determine a value; it is not a third report but the absence of a verdict. Section 4.2 gives the resolver, and Section 2.4 says in which rounds its result can be relied upon. In its decision round, the round in which it fixes its output, the receiver decides 1 if some root resolves to I and decides 0 otherwise. A single root resolving to I is enough, because it forces the same positive resolution at every other honest receiver that completes the receive stage of round 2 and so keeps running after it; we call such a receiver *continuing*. A receiver that terminates in a send stage first sends its signed views for that round together with its stop statement. That last message is what lets the receivers still active compare their own records against the views of a receiver that has already stopped.

2.2 Validity and Weak Consistency of Each Resolution

A receiver decides from its resolved roots alone, so two properties of a single resolution already carry the agreement and the validity of the protocol; how early a receiver may stop is a separate matter, settled by the round condition of Section 2.4. *Validity* ties the resolution of an honest receiver's root to the report that this receiver actually sent. *Weak consistency* makes a positive resolution contagious: no honest receiver can resolve a root to I while another one fails to do so. We state both for a fixed root and for continuing honest receivers.

- **Validity.** If P_i is honest, then after receiving in round 2, or in any later round, every continuing honest receiver resolves root_i to the value of the report that P_i sent. In particular, the root resolves to I if P_i sends I_i, and to O if P_i sends O_i.
- **Weak consistency.** Fix a root root_i and two continuing honest receivers P_j, P_k of the same execution, and let each of them resolve the root in its own decision round, the round in which the protocol has it fix its output. If P_j resolves root_i to I there, then so does P_k. A decision round is in particular an *admissible resolution round*, that is, one in which the receiver holds enough of the forwarded copies for the guarantee to hold, and Section 2.4 turns that into a condition on the round number. The two receivers need not be in the same round, and P_k's round may well be the earlier one.

Only the second property constrains the round. Validity holds in every round from 2 onwards, whereas a resolution taken too early carries no guarantee against the resolutions of the other receivers. Validity holds because an honest P_i sends the same report to every receiver, and by ideal authentication every copy of that report carries the value P_i gave it, so no two copies can conflict. Nor does P_i leave a required message unsent, since it follows the protocol, and when it stops it says so to every other receiver, so no receiver is left waiting for a message of P_i and none meets a stop statement of P_i that P_i did not send it directly. None of the grounds for marking in Section 2.3 applies to it, and every honest receiver reads its direct record for the root label, which holds the value of the report. The report goes out even when P_i stops at once: a receiver that holds the sender's signature on bit 1 from round 1 sends I_i together with $\mathsf{stop}(i, 2)$, and only then outputs 1 and terminates in round 2. If the sender itself is honest with input 1, this is what every honest receiver does, however the remaining receivers behave, so all of them output 1 by round 2.

Weak consistency is the harder of the two, and it is also all that one can ask for. A faulty P_i may sign I_i for one receiver, sign O_i for another, and send nothing to a third, so one receiver reads I from its direct record while another reads O or holds no record at all. Agreement on the value of such a root is out of reach. It is also unnecessary: a receiver outputs 1 only on the strength of a root that resolves to I, so O and $\perp$ lead to the same decision and need not be told apart. Weak consistency therefore constrains the positive value alone and leaves the two nonpositive outcomes free.

Read in the other direction, it takes the form that the agreement proof uses: if one continuing honest receiver resolves root_i to O or to $\perp$ in an admissible resolution round, then no other continuing honest receiver resolves that root to I in an admissible round of its own. A receiver that decides 1 on the strength of some root thus forces the same root, and with it the same decision, at every other continuing honest receiver. Deciding 0 asks for more, namely a round that is admissible for every root at once, and Section 2.5 shows that one round serves them all.

2.3 Recursively Resolving Signed Views

For each label of the structure built around a root, a receiver holds at most two kinds of evidence: what the signer of that label sent it directly, and the copies of that message which other receivers forwarded one round later. As long as the receiver has found no ground to mark the signer, the signer's own word settles the label. Once it has marked the signer, that word is worth nothing, and the receiver has to reconstruct what the signer should have said from the copies that the others forwarded. Those copies are themselves signed views, one round deeper, and settling them calls for the same decision again. This is why the resolver is recursive. It evaluates a whole view before the call above it extracts a single field from that view.

Labels, values, and fields. A *view label* $(e, A, \mathsf{root}_i, d)$ names the signer P_e, the context A, the root root_i, and the round d. The context A collects the receivers whose views have already been expanded further up the branch, so the label of a view determines the round it belongs to, namely $d = 2 + |A|$, and we drop d when A is written out. The root call starts from $(i, \varnothing, \mathsf{root}_i, 2)$, the label of P_i's own report. A root label takes one of the values I, O, and $\perp$. A deeper label takes either $\perp$ or a tuple of fields, each holding a label of the preceding round: for $A = \{i\}$ the tuple has the single i-field, which holds the root label itself, and for $|A| > 1$ it has one field for each $p \in A \setminus \{i\}$, the p-field holding $(p, A \setminus \{p\}, \mathsf{root}_i)$. We write $W[e]$ for the e-field of a view W and set $\perp[e] = \perp$, and a tuple all of whose fields are $\perp$ is still a value, different from $\perp$ itself. Following fields in this way leads from a view down to a round-2 report, and we say that a view has an I *leaf* if it is I itself or if some chain of its fields ends at I. An I leaf is the only trace a report I_i leaves once it has been nested inside later views, so it is also the only route to an output of 1.

Direct records and marks. Resolving runs on two things that the receiver has accumulated while receiving. The first is its *direct record* for a label, which is the value of the signed view the signer sent it directly, or of its own signed view when it is itself the signer. A stop statement $\mathsf{stop}(e, r)$ received directly from that signer takes precedence at every label of a round after r, where it gives the all-$\perp$ tuple: a receiver that has stopped owes no further views, and its statement stands in for the ones it would otherwise have sent. Labels of round r and earlier keep the records they already have, so the report of a receiver that stops in round 2 still counts for its root. The second is the set of receivers the resolving receiver has *marked*. It marks P_e, permanently, on any of three grounds: a message required from P_e is missing or malformed; a signed view of P_e reaches it with a value that contradicts its direct record for that label, so P_e has signed two different things; or a stop statement of P_e reaches it although P_e never told it directly that it stops in that round. In the second ground the record contradicted may itself have come from a stop statement of P_e rather than from one of its signed views, and then what P_e contradicts is its own earlier announcement. Each ground is a departure from the protocol, and an honest receiver can only ever meet one at a faulty signer, so it marks only faulty receivers. The three differ in kind, though. A conflicting signed view is a transferable proof of misbehaviour, which any receiver meeting both copies can check. A missing or malformed batch is local evidence instead: it convinces its recipient and nobody else, which is why a mark is a private matter and the resolver has to reconstruct rather than accuse.

The rule. If P_e is unmarked, the resolver returns the direct record, and $\perp$ if there is none. If P_e is marked, the resolver *expands* the label: for every receiver P_v with $v \notin A \cup \{e\}$, including the resolving receiver itself, it first resolves the whole child view $W_v = \mathsf{R}^i_j(A \cup \{e\}, v)$ of the next round, then extracts the e-field of that child, which is the field in which P_v was to copy what P_e had told it. Writing U for the comparison of those fields,

$$\mathsf{R}^i_j(A, e) = \mathsf{U}\big(\{W_v[e] : 1 \le v < n,\ v \notin A \cup \{e\}\}\big).$$

If at least one field differs from $\perp$ and all such fields agree, the resolver keeps their common value; if none differs from $\perp$, or two of those that do disagree, it returns $\perp$. So a single unambiguous report of what P_e said is enough, while a contradiction among the copies leaves the label undecided. A child that resolves to $\perp$ contributes $\perp$, and so does a child that resolves to a tuple whose e-field is empty. A child view of a round later than the last one the receiver completed resolves to $\perp$ as well, which is what stops the recursion from asking for messages that have not been sent yet. Figure 2 shows one such step.

Only a marked signer is ever expanded, so every context the resolver reaches consists of receivers it has marked, and each expansion adds one more of them. The recursion therefore terminates, its depth is bounded

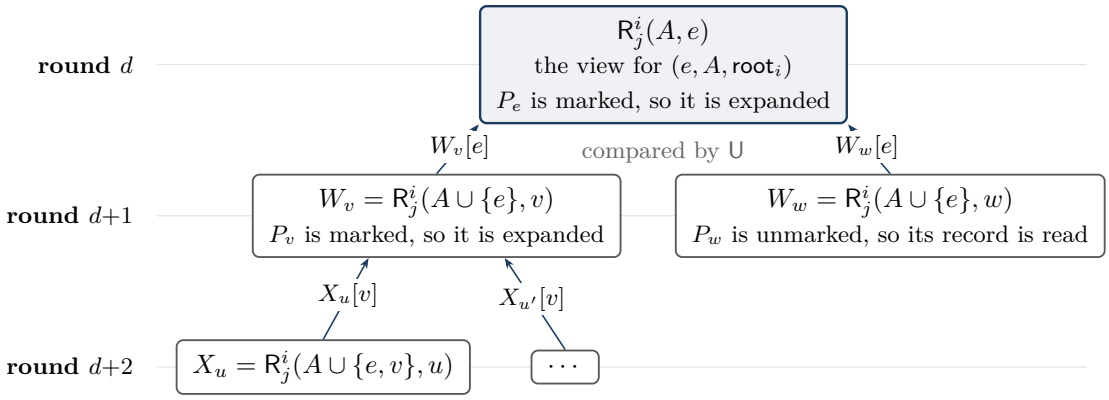

Fig. 2. One step of the resolver, seen from below. The resolver descends from the parent label (e, A, root_i) to its children one round deeper, and the values travel back up. Each child is resolved as a whole view first, and the arrow then carries the single field that the parent extracts from it, with U comparing those fields. A child is expanded in turn when its own signer is marked, as on the left, and is read from the direct record when that signer is unmarked, as on the right, where the recursion stops.

by the number of marks rather than by t, and it visits no label later than round $b_j + 2$ when the receiver holds b_j marks, a branch that runs out of marked signers stopping earlier. Section 2.4 turns this count into the condition on the round in which a resolution may be trusted. It also shows the price of a merge: for a nested I_i to reach I at the root, every step on its chain must extract a field carrying that report and must keep its value when it is compared with the other fields.

It is worth mentioning how large this structure is. In round d a receiver signs one view for every root other than its own and every context of $d-2$ receivers that contains that root's index and omits its own, which is $(n-2)\binom{n-3}{d-3}$ labels for $n \geq 3$ and $3 \leq d \leq n$. Under a literal nested encoding, in which every field carries a full copy of the child view, a single such view holds at most $(d-3)!$ copies of the root's report; what is counted here is the copies of one report inside one signed view, the count is reached when every field along the way is populated, and executions that reach it exist. Communication is therefore exponential, as it is throughout the EIG paradigm, and it is the round complexity that our construction is after.[4]

2.4 Admissible Resolution Rounds

A resolution taken too early can be wrong, in the sense that another honest receiver resolves the same root differently. This subsection says how late is late enough. The answer is a condition on the round number alone, phrased in the number of marks the receiver holds, and it is what makes weak consistency true: if an honest receiver P_k resolves root_i to I in an admissible resolution round, then every other honest receiver P_j resolves it to I in each admissible resolution round of its own, and this holds even when P_j's round is the earlier one.

The difficulty is confined to a faulty P_i. If P_i is honest, validity settles its root from round 2 onwards. A Byzantine P_i, on the other hand, may sign I_i for one receiver and O_i for another, so the direct records of two honest receivers disagree for as long as neither has seen the copy the other holds. Only the forwarded copies can close that gap, and the question is how many rounds of forwarding it takes.

Comparing views at each layer. A receiver expands only signers it has marked, so the labels that both receivers can reach are those whose context lies in $B_j \cap B_k$. These are the labels on which the two states admit a comparison, and the comparison runs by induction over them, from the deepest context upwards.

Suppose one of the two resolves such a label to a view with an I leaf. The first thing the comparison yields is that the other has, by the end of its own admissible round, seen a signed view with that label and that

[4] Our security statements use idealized signatures. A computational instantiation requires specifying how the number of parties n grows with the signature security parameter λ. Standard polynomial-time unforgeability alone does not cover executions requiring superpolynomial computation or superpolynomially many signing queries in λ.

value: at the root it has seen I_i itself, at a deeper label a signed view containing I_i. Neither receiver, in other words, can still be in the dark about the positive evidence while the other is already acting on it.

The induction then compares what the two resolvers do at the label. If both read their direct records, the records agree, since each has seen the evidence behind the other's record and a conflict would have marked the signer. If both expand, they resolve the same children and merge the same fields, so the inductive hypothesis carries the agreement upwards. The case that costs something is the mixed one, in which P_k expands a label because it marked P_e while P_j reads its record because P_e is unmarked at P_j. Among the children that P_k merges is P_j's own signed view, whose e-field is exactly P_j's direct record and differs from $\perp$. Every other field that differs from $\perp$ must carry that same value, since a field carrying another one would be a signed view conflicting with what P_e had told P_j, and P_j would then have marked P_e and expanded as well. The merge returns P_j's record, and the two agree. That P_k holds P_j's view of round $d+1$ at all costs it no extra round, since expanding P_e already puts $A \cup \{e\}$ inside B_k and so brings round $d+1$ within its reach. Lemma 5.7 carries out this induction, and at the root it is weak consistency.

Rounds needed for the comparisons. A label (e, A, root_i) belongs to round $d = 2 + |A|$, and every expansion moves one label deeper and one round later while consuming a mark that the context does not yet hold. Writing b_j for the number of marks that P_j holds, its own expansions from the root reach no further than round $b_j + 2$, and it needs the messages of that round in order to carry them out.

That is enough as long as P_j never has to hold its own record against what another receiver reconstructs. The mixed case above is where it has to: the fields that P_k merges for a label of round d are in the views of round $d+1$, so P_j must have received through round $b_j + 3$ before it can be sure that no reconstruction elsewhere contradicts what it read directly. We ask for that round only when there is positive evidence to settle, which is to say when P_j has seen I_i, and that is also precisely when P_j can afford it: such a report makes the sender faulty, the signer of the mixed label is a faulty receiver that P_j has not marked, and both are distinct from the b_j receivers it has.

Admissibility for one root. Let $b_j(D)$ be the number of marks at P_j after receiving in round D. A completed receive round $D \geq 2$ is an *admissible resolution round* for root_i at P_j if[5]

$$D \geq \min\big\{t+1,\, b_j(D) + 2 + \mathbf{1}[P_j \text{ has seen } \mathsf{I}_i \text{ by round } D]\big\}. \tag{1}$$

The three terms are the three counts above: $b_j(D) + 2$ for the depth of P_j's own expansions, the indicator for the one further round that a comparison against a reconstruction can cost, and the cap at $t+1$, which makes round $t+1$ admissible whatever the state and so leaves an admissible round in every execution. The cap never cuts a count short where the count is needed. There is something to settle only when the sender is faulty: an honest sender on input 0 signs no bit 1, so no root resolves to I at any receiver, and an honest sender on input 1 leaves no continuing receiver to disagree with. A faulty sender costs one of the t corruptions and so leaves at most $t-1$ marks, giving $b_j(D) + 2 \leq t+1$; and the comparison that asks for the further round needs a faulty receiver that P_j has not marked, which costs a second corruption, so there $b_j(D) \leq t-2$ and $b_j(D) + 3 \leq t+1$ as well.

As long as P_j has not seen I_i, the condition reads $D \geq \min\{t+1, b_j(D)+2\}$, and the root resolves to O or to $\perp$, since I calls for I_i in hand. Suppose now that some honest P_k resolves this root to I in an admissible round of its own. The comparison above shows first that P_j has seen I_i by the end of its own admissible round, even when that round is the earlier one, and then that the two resolved views agree at every label whose context lies in $B_j \cap B_k$. At the root this gives I at P_j as well. Equivalently, and this is the form the agreement proof uses, a resolution of O or $\perp$ in an admissible round rules out I in every admissible round of every other honest receiver for that root. Lemma 5.7 proves this for the decision rounds, which is the pair of rounds the protocol resolves in, and Corollary 5.8 carries it to any two admissible rounds.

2.5 When a Receiver Decides

Admissibility was defined one root at a time, while a decision needs all $n-1$ of them at once. The two fit together because a receiver resolves every root from one and the same state, the state it holds at the end of a

[5] We write $\mathbf{1}[\,\cdot\,]$ for the indicator of a statement: it is 1 when the statement holds in the execution at hand and 0 otherwise.

receive round. All resolutions share the mark count $b_j(D)$, and the only term of (1) that can tell them apart is the indicator, whose maximum over the roots is 1 exactly when P_j has seen some report I_i. A completed receive round D is thus admissible for every root at once if and only if

$$D \geq \min\big\{t+1, b_j(D) + 2 + \mathbf{1}[P_j \text{ has seen some report } \mathsf{I}_i \text{ by round } D]\big\},$$

and we call the first completed receive round with this property the *decision round* of P_j. A receiver that reaches the test applies it first after the receive stage of round 2; the exit of a receiver that holds the sender's signature from round 1, which stops in the send stage of round 2, and the case $t = 0$, which stops in round 1, do not pass through the test at all. For $t = 1$ the receive stage of round 2 is already the deadline $t + 1$.

Waiting for one round that serves all roots costs nothing. A receiver would be entitled to decide 1 the moment a single root resolves to I in a round admissible for that root alone. But resolving root_i to I means having seen I_i, so the indicator of that root is 1, and no other root can ask for more. A round admissible for the one root that would justify output 1 is therefore admissible for all of them, and the uniform rule never delays a positive decision.

At the end of its decision round the receiver resolves every root and decides 1 if some root gives I, and 0 otherwise. The second case covers in particular a receiver that has seen no report I_i at all, for then no root can give I and the condition reduces to $D \geq \min\{t+1, b_j(D) + 2\}$.

A decision before the deadline is followed by one final send. If the decision round is $D < t+1$, the receiver sends its signed views for all roots once more, in round $D + 1$, together with its stop statement, and only then outputs and terminates; at $D = t + 1$ it outputs and terminates as soon as it has received. The last message is what the receivers still running need from it, and each of its two parts does its own work. The views are the copies against which those receivers hold their own records, in the comparison that Section 2.4 charges a round for. The stop statement tells the other receivers not to expect messages from round $D + 2$ onwards. Without it, they would treat the silence as a missing message and mark the receiver, even if it was honest. A mark would then no longer imply that the receiver is faulty.

The earliest case the protocol admits is also the common one. For $t \geq 2$, a receiver that holds no mark and has seen no report I_i after receiving in round 2 already satisfies the condition, decides 0 there, and terminates in round 3 after its final send. A fault-free execution with input 0 runs exactly this way at every receiver. For $t = 0$ there is nothing to compare and the receivers decide in round 1 from the sender's message or from its silence.

It remains to turn marks into faults, which is what makes the protocol early stopping: a receiver never learns f and only ever counts its own marks, yet each mark is a distinct faulty receiver, since an honest receiver marks only faulty ones. A continuing honest receiver that has seen some report I_i moreover knows the sender to be faulty, because an honest sender on input 0 never signs bit 1 and an honest sender on input 1 makes every honest receiver output 1 and terminate by round 2; and that sender is none of the receivers it has marked. Hence

$$b_j(D) + \mathbf{1}[P_j \text{ has seen some report } \mathsf{I}_i \text{ by round } D] \leq f.$$

A continuing honest receiver therefore decides by round $\min\{f+2, t+1\}$ and outputs and terminates by round $\min\{f+3, t+1\}$. A continuing receiver thus sends once more when it decides before round $t+1$, while at the deadline it decides and stops in the same round.

3 Preliminaries

We fix the model and the task, and we introduce the terms that the protocol and its analysis use throughout.

3.1 The Model and the Task

Let $\mathcal{P} = \{S = P_0, P_1, \ldots, P_{n-1}\}$ be a set of n parties. The party S is the *sender* and holds a bit b_S, and we call the remaining parties $P_1, \ldots, P_{n-1}$ the *receivers*. Where no confusion arises we identify a receiver with its index, so that a set of receiver indices and the corresponding set of receivers are the same object.

Corruptions. An adversary corrupts parties adaptively over the course of an execution. A corrupted party is Byzantine: it may deviate from the protocol in any way, and the adversary learns its state and its signing key. The adversary is moreover *strongly adaptive*, in that within a round it may see the messages an honest party sends, corrupt that party, and replace or withhold those messages before they are delivered. Fix an execution, let f be the number of parties ever corrupted in it, and let $\mathcal{H} \subseteq \mathcal{P}$ be the set of parties never corrupted in it, which we call *honest*. Every guarantee below is a guarantee for the parties in $\mathcal{H}$. The honest parties know a public fault bound $0 \le t < n$ and are promised $f \le t$, but neither the set of corrupted parties nor its size is known to them.

Network and rounds. The parties communicate over a complete synchronous network of reliable point-to-point channels, and every channel tells its recipient which party sent the message on it, so a receiver always knows from whom a message came directly. An execution proceeds in rounds, and a round consists of a send stage followed by a receive stage: a message sent in round r is delivered in the receive stage of round r and can influence only the messages of round $r+1$ and later. The adversary is rushing, so it may fix the messages of the corrupted parties for a round after seeing the messages the honest parties send in that round; the sends of an honest party, by contrast, depend only on what it received in earlier rounds. A party *terminates* by producing its output and halting, and it sends nothing in any later round. The protocol is deterministic, and it makes no use of privacy of the channels.

Ideal authentication. We work with ideal authentication. Every party can sign, every party can verify any signature, and a signature can be forwarded, so that a party who receives it second-hand is as convinced as the party who received it first. Before the adversary corrupts a party it cannot produce a fresh valid signature in that party's name, although it may forward signatures it has already received; once it corrupts a party it obtains that party's signing key and may sign in its name, including on data labelled with an earlier round. Each signature binds the protocol instance, the type of the signed data, and its canonical contents. For a signed view those contents comprise its exact label together with all of its fields, nested signatures included, so two signed views with the same label can be compared by comparing what they carry.

Definition 3.1 (Early-stopping Byzantine broadcast). *Let $T(f,t)$ be a round bound. A protocol solves* early-stopping Byzantine broadcast *with stopping bound T if the following three properties hold in every execution in which $f \le t$ parties are ever corrupted.*

- ***Validity.*** *If the sender S is honest, every honest party outputs the sender's input b_S.*
- ***Agreement.*** *All honest parties output the same bit.*
- ***Termination.*** *Every honest party outputs and terminates by round $T(f,t)$.*

The protocol of Section 4 achieves $T(f,t) = \min\{f+3, t+1\}$, and in the particular case of an honest sender with input 1 every honest party terminates by round 2.

3.2 View Labels and Signatures

View labels and values. Fix a root root_i. A *view label* $(e, A, \mathsf{root}_i, d)$ names the signer P_e, the context A, the root, and the round $d = 2 + |A|$. The context records which receivers' views have already been expanded along the branch that leads to the label, and since it determines the round we omit d whenever A is written out. For $A = \varnothing$ the label is the *root label*: it refers to P_i's own round-2 report for root_i, so $e = i$ and $d = 2$. For a nonempty A, the set A consists of receiver indices and contains i, and e is a receiver index outside A.

The resolver of Section 4.2 returns one of I, O and $\perp$ at a root label, the last recording the absence of a verdict rather than a third report. It returns either $\perp$ or a tuple of fields at a label with a nonempty context. When $A = \{i\}$ the tuple has a single field, its i-field; when $|A| > 1$ it has one field for each $p \in A \setminus \{i\}$. The label expected in the p-field is $(p, A \setminus \{p\}, \mathsf{root}_i)$, which is a label of the preceding round, so a tuple records what its signer was told about the root one round earlier. A tuple all of whose fields are $\perp$ is still a view value and is distinct from $\perp$ itself. We set $\perp[p] = \perp$ for every required field p.

Signed views and stop statements. The reports I_i and O_i are P_i's signed views with the root label of root_i, and their values are I and O respectively. A valid I_i carries S's valid signature on bit 1 for this instance. Since the two reports carry different values under one label, two copies of that label can be told apart wherever they are met.

The format is strict, and the analysis depends on it. A valid signed view with a root label is I_i or O_i and nothing else. A valid signed view with a nonempty context carries exactly the tuple of fields that its label prescribes, no more and no fewer. The symbol $\perp$ is an empty field, an absent record or a result of the resolver; it is never itself signed, and in particular no signed object carries $\perp$ at a root label. A stop statement applies to no root label, since it names a round $r \geq 2$ and speaks only of rounds after r. Were a signed $\perp$ admissible at a root, a faulty receiver could hand one honest receiver an object that counts for nothing and another the report I_i, and the first would have no ground to mark it; the marking rules of Section 4.5 rest on the strict format to exclude this. The signed stop statement $\mathsf{stop}(i, r)$, with $r \geq 2$, states that P_i stops in the send stage of round r. It applies to every view label whose signer is P_i and whose round is after r, and at such a label it gives the tuple whose required fields are all $\perp$. At the labels of round r and earlier it says nothing, so a receiver that stops in round 2 still has its report at the root label.

For a nonempty context the signer signs the view label together with all of the fields. Each field holds $\perp$, a signed view with the expected label, or a stop statement applicable to that label. The value of the signed view is the tuple of the values of its fields, an empty field contributing $\perp$. We compare signed views with the same label by their values.

Merging views. Let X be a family of values belonging to one and the same label, so that they are of the same shape. Then $\mathsf{U}(X)$ compares all the entries of X other than $\perp$ at once: if at least one such entry exists and they all agree, $\mathsf{U}(X)$ is their common value, and otherwise $\mathsf{U}(X) = \perp$. In particular, an output other than $\perp$ equals every input other than $\perp$, which is the property the resolver of Section 4.2 rests on.

4 The Broadcast Protocol

We give the full broadcast protocol, including the rules each party follows in each round.

4.1 State after Each Receive Round

A receiver keeps four things while it receives, and all four only grow. The first is its set B_j of marked receivers, whose size after the receive stage of round D we write $b_j(D)$; marks are permanent, so B_j never shrinks. The second is the record of whether P_j has ever seen a report I_i for some receiver P_i, which once set stays set. The third is a family of direct records, one for each view label (e, A, root_i) with $j \notin A$ whose round P_j has completed. A direct record holds two things: the value of the label, which is what the resolver reads, and the signed object behind that value, which is what P_j copies into the fields of the views it signs one round later. The object is the signed view that P_e sent P_j directly, or, when $e = j$, the signed view in P_j's own batch, which it keeps and treats as directly received, or a stop statement of P_e that P_j received directly. Once set, a record never changes. The fourth is a table of the stop statements P_j has received directly: for every signer from which it has accepted a top-level stop statement, the round that statement named and the signed statement itself, which is the object P_j copies into the views it signs afterwards and cannot be recovered from the round number alone. The table is what makes the precedence rule of Section 4.5 well defined at labels of later rounds, whose records are installed only after the statement has arrived. At the outset P_j holds no mark, no record and no stop statement, and has seen no report.

We say that P_j *has seen* a signed view or a stop statement by round D if the object occurs in what P_j inspects in one of the receive rounds up to D: its own batch, the valid batches of the signers unmarked at the start of that receive stage, and every object nested inside them. Section 4.5 states the inspection rule in full, and nothing outside it counts as seen; in particular P_j has seen the views of its own batch, which it treats as directly received. A stop statement of P_e that P_j has seen gives the all-$\perp$ tuple at every label whose signer is P_e and whose round lies after the stated stop round and at most D.

4.2 Resolving a Root

Fix the state of P_j after it has received in a round $D \geq 2$, and use the marks and the direct records of that state throughout the resolution; nothing in the state changes while the resolution runs. On a branch, the context A holds the receivers whose views have already been expanded above. The root sits in round 2, and an expansion passes from a label of round d to labels of round $d+1$ by adding the signer e to A. For a label of round $d = 2 + |A|$, set

$$\mathsf{R}_j^i(A,e) = \begin{cases} \bot, & d > D, \\ \text{value in the direct record}, & d \leq D,\ e \notin B_j, \\ \mathsf{U}\big(\{\mathsf{R}_j^i(A \cup \{e\}, v)[e] : 1 \leq v < n,\ v \notin A \cup \{e\}\}\big), & d \leq D,\ e \in B_j. \end{cases} \tag{2}$$

If the resolver reads a record and there is none, it returns $\bot$. An expansion resolves each child view in full before it extracts the e-field of that child, and it applies U to all of those field values at once. That field is a required one: if $A = \varnothing$ then $e = i$ and the child context $\{i\}$ has the i-field alone, and if $A \neq \varnothing$ then $e \neq i$ and the e-field is required in the context $A \cup \{e\}$. The index v runs over every receiver outside $A \cup \{e\}$, among them P_j itself, which is why P_j keeps a copy of each view it signs.

Only a marked signer is expanded, so every expansion adds an index of B_j to A and every context the resolver reaches satisfies $A \subseteq B_j$. A branch therefore admits at most $b_j(D)$ expansions, the recursion ends, and every label it visits has round number at most $b_j(D) + 2$. The result for root_i is $\mathsf{R}_j^i(\varnothing, i)$. A view has an I leaf if the view itself is I or if following its fields leads to I.

4.3 The Decision Round

For a receiver P_j that receives in round 2, write $x_j(D) \in \{0,1\}$ for the indicator of the event that P_j has seen some report I_i by the end of receive round D. Say that D *meets the test* if

$$D \ \geq\ \min\big\{\, t+1,\ b_j(D) + 2 + x_j(D) \,\big\}, \tag{3}$$

where $b_j(D)$ and $x_j(D)$ are read off the state of P_j at the end of that round. The *decision round* D_j is the first completed receive round that meets the test. Since round $t+1$ meets the test whatever the state, a receiver that completes the receive stage of that round has $D_j = t+1$. Thus D_j is well defined and $D_j \leq t+1$. After each receive round it completes, P_j applies the test to the state of that round, and the first time it holds, P_j resolves every root in that state and decides its bit, which is 1 if some root resolves to I and 0 otherwise.

4.4 Round Procedure

Protocol 1. Early-stopping broadcast from signed views

Input. The sender's bit b_S and the public fault bound t.

Round 1. If $b_S = 1$, S signs bit 1 and sends the signature to every receiver. If $b_S = 0$, it stays silent. It outputs b_S and terminates. Each receiver records whether it received a valid signature on bit 1 directly from S in this round. For $t = 0$, it outputs 1 if it received such a signature and 0 otherwise, then terminates in round 1.

Round 2. If P_j recorded such a signature in round 1, it sends I_j and $\mathsf{stop}(j,2)$ to every other receiver, outputs 1, and terminates before receiving. Otherwise it sends O_j to every other receiver, keeps its own report, and processes its round-2 messages.

Sending signed views in round $d \geq 3$**.** In round d, P_j signs a view with label (j, A, root_i) for each root root_i and each set A of $d-2$ receiver indices containing i and excluding j. In each required field p it puts $\bot$ if P_p is marked after the receive stage of round $d-1$, and otherwise the signed view or earlier stop statement that it used to set its direct record for the label $(p, A \setminus \{p\}, \mathsf{root}_i)$. It sends the same message to all other receivers and keeps its own copy.

Protocol 1. Early-stopping broadcast from signed views (continued)

Decision round. In each round from 3 to D_j, as given by (3), P_j sends its signed views for that round and then processes the messages it receives. It decides its bit after receiving in round D_j, as in Section 4.3. If $D_j = t+1$, it outputs and terminates at the end of that receive stage. If $D_j < t+1$, it sends the signed views for round $D_j + 1$ together with the stop statement $\mathsf{stop}(j, D_j + 1)$, and outputs and terminates in that send stage, before receiving.

4.5 Processing Received Messages

Messages from S after round 1 are ignored. The message a receiver sends on one link in one round is a *batch*, and the rules below say when a batch is valid, how its contents are inspected, and when its author is marked.

Valid batches. The batch of P_e for round d is valid if it contains exactly one valid signed object for every label that P_e is scheduled to send in round d by the sending rule of Section 4.4, the round-2 report included, together with at most one top-level stop statement signed by P_e and naming round d itself. A batch that is incomplete, that repeats a label, that carries an object for a label outside its schedule or for another round, or that is malformed anywhere inside its nesting, is invalid as a whole. Receiver P_j treats the batch it sent itself as one it received directly, without sending it to itself.

Within a batch, each nonempty field is checked against the exact label that Section 3.2 prescribes for it, namely a valid signed view with that label, or a valid stop statement of that label's signer naming a round strictly earlier than the label's round. A field that fails this check makes the whole batch invalid. Only a valid top-level stop statement, received directly from its signer in the round it names, releases that signer from its later send obligations; a stop statement met inside somebody else's view is evidence and nothing more, and does not alter P_j's direct stop record.

What P_j inspects, and what it has seen. In the receive stage of round D, receiver P_j inspects its own batch and the valid batches of the signers that are unmarked at the start of that stage, among them a signer whose stop statement it already holds, and it inspects every object nested inside those batches, including objects signed by receivers it has marked. This inspected collection is exactly what the phrase *has seen* of Section 4.1 refers to, so an object that P_j acts on is always one it has seen.

Recording views and marking receivers. Receiver P_j opens an empty list of receivers to mark in this round and proceeds in three steps, in this order: the records of the round are installed first, the copies are compared against them second, and the new marks are committed only at the end, so that a mark acquired in a round does not change how that same round is processed.

1. *Records.* Install the direct records of round D. If P_j already holds a stop statement received directly from an unmarked signer and naming a round before D, it records the all-$\perp$ tuple at each of that signer's round-D labels with $j \notin A$. For every other unmarked signer it reads the valid batch and records both the values of that signer's round-D labels with $j \notin A$ and a top-level stop statement of round D that the batch carries. An unmarked signer for which P_j holds no directly received stop statement naming a round before D is put on the list if its round-D batch is missing or invalid. Records and directly received stop statements of earlier rounds stay as they are.
2. *Comparison.* Go through the inspected collection. If a report I_i occurs anywhere in it, P_j remembers that it has seen one. For each signer that is unmarked at the start of the stage and is not yet on the list, P_j adds it if some inspected signed view of that signer with a label satisfying $j \notin A$ has a value other than P_j's record for that label, or if some inspected stop statement of that signer names a round for which P_j holds no directly received statement of that signer naming the same round.
3. *Marks.* Add the receivers on the list to B_j.

An honest receiver sends a complete and well-formed batch in every round the protocol asks one of, sends the same signed views and stop statements to all the others, and announces its stop to all of them in the round it names. By ideal authentication, every copy of one of its signed views or stop statements carries the value it gave. None of the three grounds above can therefore be met by an honest receiver, and an honest

receiver marks only faulty receivers. It is the strictness of the batch format that makes this a test the receivers can apply: a signer that leaves an obligation unmet, or that pads or reshapes a batch, is caught by the first ground rather than passing as the author of a weaker, still admissible message.

5 Helper Lemmas

We collect the lemmas on which the analysis rests. The first group relates a resolution to the evidence behind it, and the second gives weak consistency for one root in the decision rounds of two receivers.

5.1 Resolution Lemmas

An unmarked signer is one that has given the receiver no ground to mark it, and the first lemma says what that buys: every copy of such a signer's message that the receiver ever meets agrees with what the signer told it directly. This is the step that lets a receiver treat its own record as the last word on a label whose signer it has not marked.

Lemma 5.1 (Agreement with the direct record). *Fix an honest receiver P_j after receive round D and a view label (e, A, root_i) of round $d \le D$ with $j \notin A$. Suppose P_e is unmarked at P_j. Every signed view with that label that P_j has seen by round D has the value P_j recorded for the label. The same holds for each stop statement from P_e that P_j has seen by round D and that applies to the label.*

Proof. Marks are permanent, so P_e was unmarked at P_j after round d as well. Had the message required of P_e in round d been missing or invalid, step 1 of the receive rule in Section 4.5 would have put P_e on the list for that round. Since it did not, P_j set the record for the label in round d, either from a signed view received directly from P_e or from a stop statement of P_e received directly in an earlier round; when $e = j$ the signed view is the one in P_j's own batch, which Section 4.5 has it treat as directly received. In both cases the record carries a value and not the absence of one, an all-$\perp$ tuple being a value in its own right, and by the same rule it stays fixed from then on. Suppose now that P_j has seen a signed view with that label whose value differs from the record. Since $j \notin A$, step 2 of the receive rule compares exactly such a view against the record, so P_j would have marked P_e in the round in which it saw the view, contrary to the hypothesis.

For the second claim, let P_j have seen a stop statement of P_e that applies to the label. By step 2, had P_j held no directly received statement of P_e naming the same stop round, it would have marked P_e. So it holds one, and the stop round that both statements name precedes d, since the statement applies to a label of round d. The directly received statement is therefore the one P_j used to set the record in round d. Two stop statements with the same stop round give the same tuple at a given label, so the two agree. □

In particular, fix an honest receiver P_j after receive round $D \ge 2$. If P_j has seen l_i by round D and $i \notin B_j$, then at the root label, which has $A = \varnothing$ and round $2 \le D$, the resolver returns the direct record, and that record has the value l of l_i by Lemma 5.1.

Resolution for honest signers. Fix an honest receiver P_j after receive round D and a view label (e, A, root_i) of round $d \le D$ with $j \notin A$. If P_e is honest, it is unmarked at P_j by the last paragraph of Section 4.5, so $\mathsf{R}^i_j(A, e)$ returns the value in the direct record. That value is the value of the signed view P_e sent with the label, or the all-$\perp$ tuple given by its stop statement if P_e terminated before round d. An honest root signer sends its round-2 report in the send stage of round 2 and can terminate only afterwards, and its stop statement names a round $r \ge 2$, so it applies to no label of round r or earlier, in particular to no root label. Its root therefore resolves to the value of that report after every completed receive round $D \ge 2$.

The next lemma shows that every resolution other than $\perp$ is supported by a signed object that the receiver has seen. If the result has an l leaf, the receiver has seen the corresponding report l_i.

Lemma 5.2. *Fix an honest receiver P_j after receive round D and a view label (e, A, root_i) with $j \notin A$. If $\mathsf{R}^i_j(A, e) \ne \perp$, then by round D, P_j has seen a signed view with that label or a stop statement applicable to it, with value $\mathsf{R}^i_j(A, e)$. If the result has an l leaf, it comes from l_i itself or a signed view containing l_i along the corresponding chain of fields. In particular, if root_i resolves to l, P_j has seen l_i by round D.*

Proof. Write $d = 2 + |A|$ and induct on $|B_j \setminus A|$, which is the number of expansions a branch from this label can still make. If $d > D$ the resolver returns $\perp$ and there is nothing to prove. If it reads the direct record, then P_e is unmarked at P_j after round D and hence after round d, so by step 1 of the receive rule in Section 4.5 P_j set that record in round d from a signed view received directly or from a stop statement of P_e received directly earlier. Either way P_j has seen that object by round D, and its value is the one the resolver returns.

Now let the resolver expand. The expansion adds $e \in B_j \setminus A$ to the context, so the induction parameter drops by one and the hypothesis applies to every child: for each child result other than $\perp$, P_j has by round D seen a signed view with the child's label, or a stop statement applicable to it, of that child's value. Suppose the expansion returns a value other than $\perp$. By the definition of U that value is the e-field of some resolved child, and it differs from $\perp$, so that child's value is not the all-$\perp$ tuple and the evidence behind it is a signed view rather than a stop statement. By Section 3.2 the e-field of that signed view holds a signed view with the parent label or a stop statement applicable to it, of the value the expansion returns, and P_j has seen it, since a copy nested inside a view that P_j has seen has itself been seen.

Finally, suppose the result has an I leaf. Following the corresponding chain of fields inside the signed view just produced reaches I at a root label, and a signed view with a root label and value I is a report I_i; no stop statement can stand at a root label, since a stop statement names a round $r \geq 2$ and applies only after it. At the root the view produced is this report itself. In both cases P_j has seen I_i. □

The last lemma of this group is the one that ties a positive report to the sender, and it is used whenever a round count has to be paid for out of the corruption budget.

Lemma 5.3 (A positive report exposes the sender). *If an honest receiver completes receive round $D \geq 2$ and has seen a report I_i by then, the sender is faulty.*

Proof. A valid report I_i carries S's signature on bit 1 for this instance. An honest sender with input 0 never issues that signature, and by ideal authentication no one else can issue it in its name. An honest sender with input 1 sends it directly to every receiver in round 1, so every honest receiver terminates in the send stage of round 2 and none of them completes receive round 2. □

5.2 Weak Consistency for One Root

Fix two distinct honest receivers P_j and P_k that receive in round 2, and let D_j and D_k be their decision rounds from (3). Throughout this subsection B_j is P_j's marked set after its decision round and $b_j = |B_j|$, the resolver R_j^i is run on P_j's state in that round, and likewise for P_k. Fix a root root_i.

The two receivers may decide in different rounds, so before the states can be compared we need to know how much each of them has had to wait. That is what the next lemma gives, and both of its bounds are paid for out of the corruption budget.

Lemma 5.4 (Bounds on the decision rounds). *If S is faulty, then $D_j \geq b_j + 2$. If P_j has seen I_i by D_j and a faulty receiver is still unmarked at P_j, then $D_j \geq b_j + 3$.*

Proof. By the last paragraph of Section 4.5 every mark of P_j is a faulty receiver, and distinct marks are distinct receivers. If the sender is faulty it accounts for one of the at most t corruptions on its own, so at most $t - 1$ receivers are faulty and $b_j \leq t - 1$. Hence $t + 1 \geq b_j + 2$, while the second term of the inner minimum in (3) is at least $b_j + 2$ in any case. The minimum is therefore at least $b_j + 2$, and D_j meets the inequality by definition, so $D_j \geq b_j + 2$. For the second claim, P_j has seen a report by D_j, so the predicate in (3) holds and the second term of the inner minimum is $b_j + 3$. By Lemma 5.3 the sender is faulty, and the unmarked faulty receiver is a second corrupted party, distinct from the sender and from the b_j receivers that P_j has marked. These $b_j + 2$ parties are all corrupted, so $b_j + 2 \leq t$ and $b_j + 3 \leq t + 1$. The inner minimum is then exactly $b_j + 3$, and $D_j \geq b_j + 3$. □

The following lemma is what makes the last message of a terminating receiver count. It says that whenever a receiver reads a label from its own record, the evidence behind that record reaches the other receiver one round later, packed into a field of a view that the first receiver signs on its way out.

Lemma 5.5 (Receiving the next-round copy). *Consider a view label (e, A, root_i) of round $d = 2 + |A|$. Suppose $j, k \notin A \cup \{e\}$, $d \le D_j$, $d+1 \le D_k$, and P_e is unmarked at P_j after round D_j. By round D_k, P_k has seen P_j's signed view with label $(j, A \cup \{e\}, \mathsf{root}_i)$. Its e-field contains the signed view or stop statement used to set P_j's direct record for (e, A, root_i).*

Proof. Since marks are permanent, P_e was unmarked at P_j after round d as well. We first check that P_j sends in round $d+1$. If $d < D_j$ it has not yet decided and sends the views of that round as a matter of course. If $d = D_j$, then $d+1 \le D_k \le t+1$ gives $D_j \le t$, so $D_j < t+1$ and the protocol has P_j send the views of round $D_j + 1$ together with $\mathsf{stop}(j, D_j + 1)$ before it outputs and terminates. The set $A \cup \{e\}$ has $d-1$ elements, contains i and excludes j, so the sending rule of Section 4.4 asks P_j for a view with label $(j, A \cup \{e\}, \mathsf{root}_i)$ in round $d+1$, and the label expected in its e-field is (e, A, root_i). Since P_e is unmarked at P_j after round d, that field does not get $\perp$ but the very signed view or stop statement that P_j used to set its direct record for the label. Receiver P_k is still receiving in round $d+1$, because $d+1 \le D_k$. As P_j is honest it is unmarked at P_k, so P_k keeps its message, and $k \notin A \cup \{e\}$, so P_k records the value of that view. □

The next lemma shows that if P_k resolves a label to a value other than $\perp$, then, under the stated hypotheses, P_j has seen a signed object carrying that value by its decision round. It does not require P_j to resolve the label to the same value.

Lemma 5.6. *Suppose S is faulty and P_k has seen I_i by D_k. Consider a view label (e, A, root_i) with $A \subseteq B_k$ and $2 + |A \cup B_j| \le D_j$. If $\mathsf{R}^i_k(A, e) \neq \perp$, then by D_j, P_j has seen a signed view with that label, or a stop statement applicable to that label, of value $\mathsf{R}^i_k(A, e)$.*

Proof. Induct on decreasing $|A|$, the largest contexts being treated first, and write $d = 2 + |A|$. The assumed bound gives $d \le 2 + |A \cup B_j| \le D_j$, and $A \subseteq B_k$ gives $d \le b_k + 2 \le D_k$ by Lemma 5.4, so both receivers have received through round d. Every receiver indexed by A is marked at P_k and therefore faulty, so $j, k \notin A$. We distinguish the two things P_j can do at the label.

P_j reads directly. Suppose first that P_e is honest. Then P_e is unmarked at both receivers, so both read their direct records, and both records were set from what P_e sent them, namely the same signed view, or the all-$\perp$ tuple of the same stop statement if P_e had already terminated. The evidence behind P_j's record therefore carries the value of P_k's result.

Suppose instead that P_e is faulty. Then e is neither j nor k, so $j, k \notin A \cup \{e\}$, and we may apply Lemma 5.5 once we know that $d+1 \le D_k$. If P_k reads directly, then P_e is faulty and unmarked at P_k, which together with P_k having seen I_i gives $D_k \ge b_k + 3 \ge d+1$ by Lemma 5.4. If P_k expands, then $A \cup \{e\} \subseteq B_k$ and $d+1 \le b_k + 2 \le D_k$ by the same lemma. In both cases Lemma 5.5 applies, and P_k has seen, by D_k, the view P_j signed for $(j, A \cup \{e\}, \mathsf{root}_i)$, whose e-field holds the evidence behind P_j's record.

If P_k reads directly, that nested object is a signed view or stop statement for the label (e, A, root_i) which P_k has seen while leaving P_e unmarked, so by Lemma 5.1 its value is P_k's record, which is P_k's result. If instead P_k expands, one of its children is $\mathsf{R}^i_k(A \cup \{e\}, j)$, and since P_j is honest and hence unmarked at P_k, that child is read from P_k's record for P_j's view, whose e-field is the value of P_j's record and differs from $\perp$. A result of U other than $\perp$ equals every input other than $\perp$, so P_k's result is that value as well. In both cases the evidence behind P_j's own record has the value $\mathsf{R}^i_k(A, e)$, and P_j has seen it.

P_j expands. Suppose first that P_k reads directly. Then $e \in B_j \setminus A$, and P_e is faulty because the honest P_j has marked it, so $j, k \notin A \cup \{e\}$. Moreover B_j contains e and A does not, so $d+1 = 3 + |A| \le 2 + |A \cup B_j| \le D_j$. Applying Lemma 5.5 with the roles of the two receivers exchanged, P_j has seen by D_j the view P_k signed for $(k, A \cup \{e\}, \mathsf{root}_i)$, whose e-field holds the evidence behind P_k's record, that is, an object with the value $\mathsf{R}^i_k(A, e)$. Suppose instead that P_k expands as well. Every child of that expansion has context $A \cup \{e\} \subseteq B_k$, and since $e \in B_j$ the set $A \cup B_j$ is unchanged by adding e to A, so the round budget of the lemma holds for the children and the induction hypothesis applies to each of them. A result of U other than $\perp$ is the e-field of one of those children and differs from $\perp$, so that child's value is not the all-$\perp$ tuple and the object P_j has seen for it is a signed view. The e-field of that signed view is a signed view with the parent label or a stop statement applicable to it, of the value the merge returned, and P_j has seen it as a nested copy. □

We can now state weak consistency for one root. The first half of the proof says that neither receiver can be in the dark about I_i while the other acts on it, and the second half turns that into equality of the resolved views at every label the two states have in common.

Lemma 5.7 (Weak consistency in the decision rounds). *Receiver P_j resolves root_i to I at D_j if and only if P_k resolves it to I at D_k. For any view label (e, A, root_i) with $A \subseteq B_j \cap B_k$, if either receiver resolves that label to a view with an I leaf in its decision round, the other returns the same view value for that label in its own decision round.*

Proof. **Both receivers have seen I_i.** Fix a label with $A \subseteq B_j \cap B_k$ and suppose P_k resolves it to a view with an I leaf. By Lemma 5.2, P_k has seen I_i by D_k, and the sender is faulty by Lemma 5.3. Since $A \subseteq B_j$ we have $2 + |A \cup B_j| = b_j + 2 \le D_j$ by Lemma 5.4, so Lemma 5.6 applies and P_j has seen, by D_j, a signed view with that label or an applicable stop statement of the same value. A stop statement gives only $\perp$ fields and so has no I leaf, so what P_j has seen is a signed view, and it is I_i itself or contains I_i along the chain of fields that leads to the leaf. Both receivers have therefore seen I_i by their decision rounds, and the same argument with the receivers exchanged covers the case in which it is P_j that sees the I leaf.

Equality of the resolved views. Assume now that both receivers have seen I_i by their respective decision rounds, which is what the first half supplies in every case the statement covers. We claim that $\mathsf{R}^i_j(A, e) = \mathsf{R}^i_k(A, e)$ for every view label (e, A, root_i) with $A \subseteq B_j \cap B_k$, and we induct on decreasing $|A|$. Write $d = 2 + |A|$. Since $A \subseteq B_j$ and $A \subseteq B_k$, Lemma 5.4 gives $d \le b_j + 2 \le D_j$ and $d \le b_k + 2 \le D_k$, so both receivers have received through round d by their decision rounds and neither returns $\perp$ for the trivial reason that the label lies in the future.

If both read directly, then P_e is unmarked at both after round d, so by step 1 of the receive rule both records carry a value. Since $2 + |A \cup B_j| = b_j + 2 \le D_j$, Lemma 5.6 gives an object that P_j has seen of the value of P_k's result, and since P_e is unmarked at P_j that value is P_j's own record by Lemma 5.1. The two results agree. If both expand, every child has context $A \cup \{e\} \subseteq B_j \cap B_k$, so by the induction hypothesis the two receivers resolve each child to the same view. They therefore extract the same e-fields and merge the same family of values, and the two results agree.

By symmetry it remains to treat the case in which P_k expands and P_j reads directly. Then $e \in B_k \setminus B_j$, so P_e is faulty and is unmarked at P_j, and P_j has seen I_i by the first half of the proof, so $D_j \ge b_j + 3$ by Lemma 5.4. Each child of P_k's expansion has context $A \cup \{e\} \subseteq B_k$ and

$$2 + |(A \cup \{e\}) \cup B_j| = b_j + 3 \le D_j,$$

so Lemma 5.6 applies to the children. For each child result other than $\perp$, P_j has seen by D_j a signed view with the child's label, or an applicable stop statement, of that value. If such a child has an e-field other than $\perp$, then its value is not the all-$\perp$ tuple, so the object is a signed view, and the e-field of that signed view is a signed view with the parent label or a stop statement applicable to it. As P_e is unmarked at P_j, the value of that field equals P_j's direct record by Lemma 5.1. Every e-field other than $\perp$ that P_k merges therefore carries P_j's record.

At least one of them does. Indeed $d \le D_j$ and $d + 1 \le b_k + 2 \le D_k$, so by Lemma 5.5 the receiver P_k has seen by D_k the view P_j signed for $(j, A \cup \{e\}, \mathsf{root}_i)$, and it reads that child from its record because P_j is honest and hence unmarked. The e-field of that child is the value of P_j's record, which differs from $\perp$. So U is applied to a family with a member other than $\perp$, all of whose members other than $\perp$ are equal to P_j's record, and it returns that record. The two results agree once more.

Thus both receivers resolve every such label to the same view value, and the same reasoning applies with the receivers exchanged. Taking $A = \varnothing$ and $e = i$ gives the statement at the root. □

The protocol resolves the roots in the decision rounds, and that is all Section 6 needs. The overview states weak consistency for the admissible rounds of Section 2.4, of which the decision round is one, and the proof above gives that reading too.

Corollary 5.8 (Weak consistency in the admissible rounds). *Write $x^i_j(D) \in \{0, 1\}$ for the indicator of the event that P_j has seen the report I_i by the end of receive round D, so that $x_j(D) = \max_i x^i_j(D)$, and call a completed receive round D of P_j* admissible for root_i *if $D \ge \min\{t + 1, b_j(D) + 2 + x^i_j(D)\}$. Let D be such a round for P_j and D' such a round for P_k, and run each resolver on the state of its own round. Then P_j resolves root_i to I at D if and only if P_k resolves it to I at D'.*

Proof. Suppose P_k resolves root_i to I in round D'. By Lemma 5.2, it has seen I_i by that round. Hence $x_k^i(D') = x_k(D') = 1$, so admissibility implies that D' satisfies the decision test in (3). Since a receiver completes no receive stage after its decision round, $D' = D_k$. The sender is faulty by Lemma 5.3.

Write $B_j(D)$ for the set of receivers marked by P_j after receive round D, so that $b_j(D) = |B_j(D)|$. We first check that the conclusion of Lemma 5.6 remains valid when P_j's state is taken after a completed receive round D satisfying $2 + |A \cup B_j(D)| \le D$, with all other hypotheses unchanged. The proof uses the decision-round bounds only for P_k. Each application of Lemma 5.5 requires the receiver sending the copy to have completed receive round d, and the receiver obtaining it to have completed receive round $d+1$. In particular, $d < t+1$. By the protocol, a receiver that completes receive round $d < t+1$ sends its views in round $d+1$, either while continuing or in its final batch. Thus these copies are still sent and received, and the induction proving Lemma 5.6 applies to this state of P_j. Since the sender is faulty, $b_j(D) \le t-1$. Admissibility gives

$$D \ge \min\{t+1, b_j(D) + 2 + x_j^i(D)\} \ge b_j(D) + 2.$$

At the root label, $A = \varnothing$ and $e = i$, so $2 + |A \cup B_j(D)| = b_j(D) + 2 \le D$. Applying the argument in the proof of Lemma 5.6 to P_j's state after round D shows that it has seen a signed object with the root label and value I. This object is the report I_i.

Consequently $x_j^i(D) = x_j(D) = 1$, so D satisfies the decision test as well. Since P_j completes receive round D, we have $D = D_j$. Now Lemma 5.7 implies that P_j resolves root_i to I in that round. Exchanging the receivers proves the reverse implication. □

6 Main Theorem

We prove validity, agreement and the stopping guarantees, and we show that the round bound is satisfied.

Theorem 6.1 (Early stopping broadcast). *Let $n \ge 1$ and let $0 \le t < n$ be a fault bound. The protocol of Section 4 solves early-stopping Byzantine broadcast in the sense of Definition 3.1 with stopping bound $T(f,t) = \min\{f+3, t+1\}$. It is deterministic and it has the following two further properties.*

1. *Suppose an honest receiver P_i processes the messages of a round $D \ge 2$, has seen no report I_e by the end of it, and satisfies*
$$D \ge \min\{t+1, b_i(D) + 2\}.$$
Then P_i decides 0 in round D, and every honest output is 0. It terminates in the send stage of round $D+1$ if $D < t+1$, and after receiving in round $t+1$ if $D = t+1$.
2. *If the sender is honest and has input 1, every honest party terminates by round 2.*

Remark 6.2. Let $n \ge 2$, so that $\tau \ge 0$; for $n = 1$ there are no receivers and the sender outputs its own input in round 1. Run with the internal threshold $\tau = \min\{t, n-2\}$ in place of t, and with its deadline at round $\tau + 1$, the protocol stops within $\min\{f+3, \tau+1\}$ rounds and remains correct for every $f \le t$. For $f \le \tau$ this is Theorem 6.1. The only further case is $t = n-1$ with $f = n-1$, where the internal promise $f \le \tau$ is exceeded. The deadline is unconditional, so every party that is still running stops in round $\tau + 1$ in any case; and a single honest party remains, so agreement is vacuous and validity says nothing unless that party is the sender, which outputs its own input in round 1. The cap therefore lowers the worst-case round complexity at $t = n-1$ by a round and improves the bound of Theorem 6.1 for $f \ge n-3$.

6.1 Validity

Proof of validity in Theorem 6.1. Let S be honest. It outputs b_S in round 1 by the protocol, so we only need to consider outputs of the receivers. For $t = 0$ a receiver outputs 1 if it receives S's signature on bit 1 in round 1 and 0 if it hears nothing, which is b_S in both cases. For $t \ge 1$ and $b_S = 1$, the sender sends that signature to every receiver, so every honest receiver records it in round 1 and outputs 1 in the send stage of round 2. For $t \ge 1$ and $b_S = 0$ the sender never signs bit 1, and by ideal authentication no other party can produce that signature in its name, so no valid report I_i exists in the execution. Consider an honest receiver P_j and its decision round, which exists by Proposition 6.3. Were some root to resolve to I in that round, P_j would have seen a report I_i by Lemma 5.2. Hence every root resolves to O or to $\perp$, and P_j outputs 0. □

6.2 Agreement

Proof of agreement in Theorem 6.1. If S is honest, all honest parties output b_S by validity, so suppose S is faulty. The honest parties are then receivers, and we distinguish two cases.

Suppose first that some honest receiver P_i terminates in the send stage of round 2. By the protocol it had recorded S's signature on bit 1 in round 1, and it sends I_i and $\mathsf{stop}(i, 2)$ to every other receiver before outputting 1. Any honest receiver in the same position does the same and also outputs 1. A continuing honest receiver, on the other hand, records I_i in round 2, and since P_i is honest it is unmarked there, so by the paragraph on honest signers in Section 5.1 the root root_i resolves to I in that receiver's decision round. The receiver therefore decides 1 as well, and all honest parties output 1.

Suppose instead that no honest receiver terminates in the send stage of round 2, so that every one of them completes receive round 2 and has a decision round. If one of them, say P_k, decides 1, then some root root_i resolves to I at D_k, and by Lemma 5.7 that same root resolves to I in the decision round of every other honest receiver, which therefore decides 1 too. If no honest receiver decides 1, every one of them decides 0. In either case all honest parties output the same bit. □

6.3 Termination

Proposition 6.3 (Termination bound). *Every honest party terminates by round* $\min\{f + 3, t + 1\}$. *If S is honest and $b_S = 1$, every honest party terminates by round 2.*

Proof. An honest sender outputs and terminates in round 1, and for $t = 0$ so does every honest receiver. For $t \geq 1$, an honest receiver that records S's valid signature on bit 1 in round 1 sends I_j and $\mathsf{stop}(j, 2)$ in the send stage of round 2 and terminates there, before receiving; this is what every honest receiver does when S is honest with $b_S = 1$, which proves the second claim. All of these termination rounds are at most 2 and so meet the first claim as well.

There remains an honest receiver P_j that completes receive round 2. Fix a round $D \geq 2$ that it completes. Every mark of P_j is a faulty receiver and distinct marks are distinct receivers, by the last paragraph of Section 4.5. If P_j has seen no report I_i by round D, the indicator in (3) vanishes and $b_j(D) \leq f$, so the second term of the inner minimum is at most $f + 2$. If P_j has seen one, the sender is faulty by Lemma 5.3 and is none of the marked receivers, so $b_j(D) \leq f - 1$ and that term is $b_j(D) + 3 \leq f + 2$. In both cases the inner minimum of (3) is at most $\min\{t + 1, f + 2\}$ at the end of round D.

Set $D^* = \min\{f + 2, t + 1\}$, and note $D^* \geq 2$. If P_j completes receive round D^*, the display of the previous paragraph shows that the inequality of (3) holds in the state of that round, so $D_j \leq D^*$; and if P_j does not complete it, that is because it has already decided, so $D_j < D^*$, and it may well be terminating in the send stage of round D^* itself, which is consistent with never processing that round's messages. Either way $D_j \leq D^*$. The protocol has P_j terminate in round $\min\{D_j + 1, t + 1\}$, which is at most $\min\{D^* + 1, t + 1\} = \min\{f + 3, t + 1\}$. □

Proof of the stopping guarantees in Theorem 6.1. Consider the honest receiver P_i of the first claim. It has seen no report I_e by the end of receive round D, so the indicator in (3) vanishes in the state of that round and the hypothesis $D \geq \min\{t + 1, b_i(D) + 2\}$ is exactly the inequality of (3). Hence $D_i \leq D$. Conversely $D_i \geq D$: assume $D_i < D$, then $D_i < D \leq t + 1$ would put P_i in the case $D_i < t + 1$ of the protocol, so it would terminate in the send stage of round $D_i + 1 \leq D$ and would not have processed the messages of round D. Thus $D_i = D$.

By Lemma 5.2 no root can resolve to I in a state in which no report I_e has been seen, so every root resolves to O or to $\perp$ and P_i decides 0. If $D_i < t + 1$ it sends its signed views and its stop statement in round $D_i + 1$ and then outputs 0 and terminates before receiving, and if $D_i = t + 1$ it outputs 0 and terminates as soon as it has received in that round. By agreement every honest party outputs 0. The remaining round bounds are those of Proposition 6.3. □

Corollary 6.4 (The stopping bound is satisfied). *For every $n \geq 2$ and $0 \leq f \leq t < n$, there is an execution with exactly f faulty parties in which the first honest receiver to terminate outputs* 0 *and terminates in round* $\min\{f + 3, t + 1\}$.

Proof. For $t = 0$ we have $f = 0$. Take an honest sender with input 0. Every receiver hears nothing in round 1, outputs 0 and terminates in round 1, which is $\min\{f + 3, t + 1\} = 1$.

For $t \geq 1$ and $f \leq n - 2$, take an honest sender with input 0 and f faulty receivers that stay silent throughout. At least one receiver is honest, since $n - 1 - f \geq 1$. Each honest receiver misses the round-2 report of every silent receiver and marks all f of them in round 2, and it acquires no further mark, because the remaining receivers are honest. The sender never signs bit 1, so no valid report I_i exists and the indicator in (3) vanishes throughout. The inequality of (3) then reads $D \geq \min\{t + 1, f + 2\}$, which fails for every smaller round and holds at $D = \min\{f + 2, t + 1\}$, so this is the decision round of every honest receiver. By Lemma 5.2 every root resolves to O or to $\perp$ in that state, so every honest receiver outputs 0 and terminates in round $\min\{D_j + 1, t + 1\} = \min\{f + 3, t + 1\}$.

For $f = n - 1$ we have $t = f = n - 1$, since $f \leq t < n$. Take the sender and all receivers but one to be faulty and silent. The sole honest receiver P_j marks the $f - 1$ silent receivers in round 2 and acquires no further mark, and it has seen no report I_i, the sender having signed nothing. The inequality of (3) reads $D \geq \min\{t + 1, (f - 1) + 2\} = t + 1$, so $D_j = t + 1$ and P_j outputs 0 and terminates after receiving in round $t + 1$. Since $f + 3 = n + 2 > n = t + 1$, this is $\min\{f + 3, t + 1\}$. □

Use of AI tools

ChatGPT-6 Astra assisted with the preparation of all figures, language editing, and checking technical claims and proofs in the manuscript. The authors developed the proofs of Lemmas 5.6 and 5.7 through discussions with ChatGPT. The authors developed all other new ideas and proofs of new results and take responsibility for all claims in the paper.

References

1. Abraham, I., Dolev, D.: Byzantine agreement with optimal early stopping, optimal resilience and polynomial complexity. In: Proceedings of the 47th Annual ACM Symposium on Theory of Computing. pp. 605–614. ACM (2015). https://doi.org/10.1145/2746539.2746581
2. Abraham, I., Nayak, K., Ren, L., Xiang, Z.: Good-case latency of Byzantine broadcast: A complete categorization. In: Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing (PODC 2021). pp. 331–341. ACM (2021). https://doi.org/10.1145/3465084.3467899
3. Albouy, T., Frey, D., Raynal, M., Taïani, F.: Good-case early-stopping latency of synchronous Byzantine reliable broadcast: The deterministic case. In: Scheideler, C. (ed.) 36th International Symposium on Distributed Computing (DISC 2022). Leibniz International Proceedings in Informatics, vol. 246, pp. 4:1–4:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2022). https://doi.org/10.4230/LIPIcs.DISC.2022.4
4. Alexandru, A.B., Loss, J., Papamanthou, C., Tsimos, G., Wagner, B.: Sublinear-round broadcast without trusted setup. In: Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2025). pp. 4132–4171. SIAM (2025). https://doi.org/10.1137/1.9781611978322.141
5. Alpturer, K., Halpern, J.Y., van der Meyden, R.: Optimal eventual Byzantine agreement protocols with omission failures. In: Proceedings of the 2023 ACM Symposium on Principles of Distributed Computing (PODC 2023). pp. 244–252. ACM (2023). https://doi.org/10.1145/3583668.3594573
6. Bar-Noy, A., Dolev, D., Dwork, C., Strong, H.R.: Shifting gears: Changing algorithms on the fly to expedite Byzantine agreement. Information and Computation **97**(2), 205–233 (1992). https://doi.org/10.1016/0890-5401(92)90035-E
7. Berman, P., Garay, J.A., Perry, K.J.: Optimal early stopping in distributed consensus. In: Segall, A., Zaks, S. (eds.) Distributed Algorithms – WDAG 1992. Lecture Notes in Computer Science, vol. 647, pp. 221–237. Springer (1992). https://doi.org/10.1007/3-540-56188-9_15
8. Chan, H., Pass, R., Shi, E.: Sublinear-round Byzantine agreement under corrupt majority. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) Public-Key Cryptography – PKC 2020. Lecture Notes in Computer Science, vol. 12111, pp. 246–265. Springer (2020). https://doi.org/10.1007/978-3-030-45388-6_9
9. Choi, W., Choi, W., Liu, X., Zikas, V.: On the round complexity of early stopping. Cryptology ePrint Archive, Paper 2026/1878 (2026), https://eprint.iacr.org/2026/1878
10. Coan, B.A.: Efficient agreement using fault diagnosis. Distributed Computing **7**(2), 87–98 (1993). https://doi.org/10.1007/BF02280838

11. Deligios, G., Elsheimy, F., Kamp, S.H., Klasovitá, I., Liu-Zhang, C.D., Loss, J., Papamanthou, C.: Optimal early-stopping broadcast via exponential information gathering. Cryptology ePrint Archive, Paper 2026/2785 (2026), https://eprint.iacr.org/2026/2785
12. Deligios, G., Klasovitá, I., Liu-Zhang, C.D.: Asymptotically optimal early termination for dishonest majority broadcast. In: Fehr, S., Fouque, P.A. (eds.) Advances in Cryptology – EUROCRYPT 2025. Lecture Notes in Computer Science, vol. 15605, pp. 116–144. Springer (2025). https://doi.org/10.1007/978-3-031-91092-0_5
13. Dolev, D., Lenzen, C.: Early-deciding consensus is expensive. In: Proceedings of the 32nd ACM Symposium on Principles of Distributed Computing (PODC 2013). pp. 270–279. ACM (2013). https://doi.org/10.1145/2484239.2484269
14. Dolev, D., Reischuk, R.: Bounds on information exchange for Byzantine agreement. Journal of the ACM **32**(1), 191–204 (1985). https://doi.org/10.1145/2455.214112
15. Dolev, D., Reischuk, R., Strong, H.R.: Early stopping in Byzantine agreement. Journal of the ACM **37**(4), 720–741 (1990). https://doi.org/10.1145/96559.96565
16. Dolev, D., Strong, H.R.: Authenticated algorithms for Byzantine agreement. SIAM Journal on Computing **12**(4), 656–666 (1983). https://doi.org/10.1137/0212045
17. Elsheimy, F., Loss, J., Papamanthou, C.: Early stopping Byzantine agreement in $(1+\epsilon)\cdot f$ rounds. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024. Lecture Notes in Computer Science, vol. 15489, pp. 398–424. Springer (2024). https://doi.org/10.1007/978-981-96-0938-3_13
18. Elsheimy, F., Loss, J., Papamanthou, C.: Towards optimal early stopping agreement protocols. Cryptology ePrint Archive, Paper 2025/246 (2025), https://eprint.iacr.org/2025/246
19. Fischer, M.J., Lynch, N.A.: A lower bound for the time to assure interactive consistency. Information Processing Letters **14**(4), 183–186 (1982). https://doi.org/10.1016/0020-0190(82)90033-3
20. Garay, J.A., Katz, J., Koo, C.Y., Ostrovsky, R.: Round complexity of authenticated broadcast with a dishonest majority. In: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007). pp. 658–668. IEEE Computer Society (2007). https://doi.org/10.1109/FOCS.2007.44
21. Garay, J.A., Moses, Y.: Fully polynomial Byzantine agreement for $n > 3t$ processors in $t+1$ rounds. SIAM Journal on Computing **27**(1), 247–290 (1998). https://doi.org/10.1137/S0097539794265232
22. Hou, R., Yu, H.: Optimistic execution in Byzantine broadcast protocols that tolerate malicious majority. Journal of Parallel and Distributed Computing **209**, 105203 (2026). https://doi.org/10.1016/j.jpdc.2025.105203
23. Keidar, I., Rajsbaum, S.: A simple proof of the uniform consensus synchronous lower bound. Information Processing Letters **85**(1), 47–52 (2003). https://doi.org/10.1016/S0020-0190(02)00333-2
24. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. ACM Transactions on Programming Languages and Systems **4**(3), 382–401 (1982). https://doi.org/10.1145/357172.357176
25. Lenzen, C., Sheikholeslami, S.: A recursive early-stopping phase king protocol. In: Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing (PODC 2022). pp. 60–69. ACM (2022). https://doi.org/10.1145/3519270.3538425
26. Loss, J., Nielsen, J.B.: Early stopping for any number of corruptions. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024. Lecture Notes in Computer Science, vol. 14653, pp. 457–488. Springer (2024). https://doi.org/10.1007/978-3-031-58734-4_16
27. Pease, M.C., Shostak, R.E., Lamport, L.: Reaching agreement in the presence of faults. Journal of the ACM **27**(2), 228–234 (1980). https://doi.org/10.1145/322186.322188
28. Perry, K.J., Toueg, S.: An authenticated Byzantine generals algorithm with early stopping. Tech. Rep. TR 84-620, Department of Computer Science, Cornell University (1984)
29. Srinivasan, S., Loss, J., Malavolta, G., Nayak, K., Papamanthou, C., Thyagarajan, S.A.K.: Transparent batchable time-lock puzzles and applications to Byzantine consensus. In: Boldyreva, A., Kolesnikov, V. (eds.) Public-Key Cryptography – PKC 2023. Lecture Notes in Computer Science, vol. 13940, pp. 554–584. Springer (2023). https://doi.org/10.1007/978-3-031-31368-4_20
30. Toueg, S., Perry, K.J., Srikanth, T.K.: Fast distributed agreement. SIAM Journal on Computing **16**(3), 445–457 (1987). https://doi.org/10.1137/0216031
31. Wan, J., Xiao, H., Shi, E., Devadas, S.: Expected constant round Byzantine broadcast under dishonest majority. Journal of the ACM **72**(4), 30:1–30:39 (2025). https://doi.org/10.1145/3748254