Adaptively Secure Consensus with Linear Complexity and Constant Round under Honest Majority in the Bare PKI Model, and Separation Bounds from the Idealized Message-Authentication Model

Abstract. We consider the mainstream model in secure computation known as the bare PKI setup, also as the *bulletin-board PKI*. It allows players to broadcast once and non-interactively before they receive their inputs and start the execution. A bulletin-board PKI is essentially the minimum setup known so far to implement the model known as *messages-authentication*, i.e., when P is forwarded a signed message, it considers it to be issued by R if and only if R signed it. It is known since [Lamport et al, 82] that BA under honest majority (let alone secure computation) would not be possible without messages-authentication. But as further highlighted by [Borcherding, 96], messages-authentication cannot not simply be implemented with digital signatures, without a bulletin-board of public keys. So the bulletin-board PKI setup and the messages-authentication model seem very close: this raizes the question whether there is a separation between them. In the bulletin-board PKI setup, the most communication-efficient synchronous BA is the one of [Boyle-Cohen-Goel, Podc'21 & J. Cryptol.'24], which has $O(n.\text{polylog}(n))$ bit complexity, $f < n(1/3 - \epsilon)$ resilience and tolerates an adversary which cannot adaptively corrupt after the setup. Our main upper-bound is a BA (and also a VBA) in this same model with strictly better parameters: *quasi-optimal* resilience $f < n(1/2 - \epsilon)$, with an expected bit complexity of communications which is *linear* in n, and tolerance to an *adaptive* rushing adversary (but which unavoidably cannot remove messages sent). As [BCG'21], it has constant expected latency. All previous BAs or VBAs achieving the same metrics as our upper bound, are either in the static adversary model: Sleepy [Pass-Shi, Asiacrypt'17], Snow White [Daian-Pass-Shi, FC'19], or assume more than a bare PKI setup: (i) The model of Thunderella [Pass-Shi, EC'17], Algorand [Gilad et al, SOSP'17], Praos [David et al, EC'18], [Goyal et al, FC'21] and [Momose et al, CCS'22 and CCS'23] assumes a public random seed which is unpredictable until strictly after all players published on the bulletin board; (ii) [Abraham et al, Podc'19] assume a trusted entity which honestly samples the keys of all players; (iii) All known implementations of the setups (i) and (ii), as well as the setup of [Blum et al, TCC'20], require interactions, furthermore in the form of BAs. (iv) [Garay-Kiayas-Shen EC'24] assume that honest players work more than the adversary, or, [Eckey-Faust-Loss et al '17 '22] at least as fast.
Of independent interest, our tool is a very simple non-interactive mechanism which *sets-up a self-sortition function from non-interactive publi-*

cations on the bulletin board, and still, guarantees an honest majority in every committee up to probability exponentially small in both ϵ and in the multicast complexity. We provide the following further results.
- *Optimality.* We show that resilience up to a tight honest majority $f < n/2$ is impossible for any multicast-based adaptively secure BA with subquadratic communication, whatever the setup.
- *Separation.* We show impossibility of subquadratic multicast-based BA in the messages-authentication model. Our model for this lower bound is even stronger, since it onboards other assumptions at least as strong as all popular implications of a bulletin-board PKI setup: *secure channels*, *a (possibly structured) random string*, *NIZK*.
- *Partial synchrony.* Given that the multicast lower-bound holds a fortiori, and that the upper-bound adapts seamlessly (for $f < n(1/3 - \epsilon)$), the separation also holds. We show a second separation, which holds for general BAs, non-necessarily multicast-based: any partially-synchronous BA in the messages-authentication model, if it has linear message complexity, then it has latency at least *logarithmic in* f.
- *Extension to VBA.* We extend to VBA the logarithmic latency lower bound. This is the first communication lower bound for randomized VBA to our knowledge. It shows that the separation under partial synchrony also holds for VBA. Along the way, we close the characterization of [Civit et al, Podc'23] of validity conditions in authenticated consensus, by apparently new results on VBA: both BA and VBA are infeasible under partial synchrony beyond $f < n/3$, whatever the setup and even randomized; whereas synchronous VBA is feasible up to $f = n-1$ (contrary to BA).

A high level introduction is provided by the abstract. Due to the number of results, we cut the introduction in three: Section 1 for the results under synchrony, Section 2 for those under partial synchrony and Section 3 for those related to external validity. Section 1 is self-contained since in Section 1.1 we give the model, then in Section 1.2 we give brief motivations then state the results, then in Section 1.3 we explain the techniques. Finally in Appendix A we discuss the impact of both the results and techniques, by putting them in perspective of previous works, which we hope will give further intuition. Sections 2 and 3 follow the same outline, although they build on the model of Section 1.1.

1 Introduction for the results under synchrony

1.1 Model and main driving questions

We state the model which holds for all our results stated under synchrony. Then, result-by-result, we will comment on the assumptions which can be modified so as to make the results further stronger. Let n be an integer. We consider a set $\mathcal{P} = (P_1, \ldots, P_n)$ of n probabilistic polynomial time (PPT) machines denoted *players*, and a PPT machine called the *adversary* $\mathcal{A}$.

1.1.1 Communication.

Players are connected by pairwise *public authenticated channels*, i.e., the adversary $\mathcal{A}$ reads the content of all messages sent. In our lower bounds the channels will be upgraded to *secure*, i.e., only the lengths of messages are leaked to $\mathcal{A}$, which thus makes the bounds stronger. We consider the classical *synchronous round-based model* of [42, 29, 28], which we now recall. Players have access to a global clock ticking every Δ, where Δ is a fixed public duration. To ease the notation, we set the unit of time equal to Δ. Hence, when we note time $t = 1$, this actually means $t = \Delta$. For r a positive integer, the time interval $[r-1, r]$ is called the *r-th round*. Players send messages at the beginning of every round, these messages are delivered before the end of the round. Players are assumed to have the time to process all the messages received before the next round starts. To multicast a message means to send it to all players, hence this does not come with any consistency nor delivery guarantee when the sender is corrupt.

1.1.2 Corruptions.

Let $f \leq n$ be an integer, $\mathcal{A}$ can corrupt up to a total of f players in the execution. $\mathcal{A}$ can *corrupt* any player at any point in time, up to the following limitations. As in [27, 37, 1, 10], $\mathcal{A}$ cannot corrupt a player in the middle of sending a batch of messages, possibly with different contents to possibly different receivers. Moreover, as in [27, 37, 10], players are able to *securely erase* part of their memory, or the totality of it, just after they sent a batch of messages and before the adversary can corrupt them. Upon corrupting a player P, $\mathcal{A}$ learns all its current state and has full control over it. It can possibly instruct P to send one or more messages in the same round in which it has been corrupt. The adversary is *rushing*, in the sense that it can use its knowledge of the messages sent by honest players in a given round to determine the messages of corrupted players in this same round. However, unlike in [1, Thm 1], the adversary cannot retract messages which have been sent. At any time, players that remain honest so far are referred to as *so-far-honest*, and the ones which remain honest until the end of the protocol are referred to as *forever-honest*.

1.1.3 Model for upper-bounds: the bare / bulletin-board PKI setup.

This model is singled-out in [CGGM00], and is also known as "bare PKI". Denote $t = 0$ the time at which parties receive their inputs and start the protocol execution. Parties can publish any string strictly before $t = 0$. Players are instructed to publish *once* and *non-interactively*, in particular, *independently* of the strings already published (otherwise, this would allow the MPC implementation of any setup). On the other hand, the adversary can wait that all honest players published their strings, before adaptively choosing the strings which corrupt players will publish. The bulletin-board PKI model is equivalent to allowing players to register the string of their choice to the ideal certification authority of $\mathcal{F}_{\mathrm{CA}}$ [17] which we recall in Appendix B.3. Then during the execution, players can retrieve to $\mathcal{F}_{\mathrm{CA}}$ the keys published by other players. Note that by definition, $\mathcal{F}_{\mathrm{CA}}$ does not accept two different strings from the same player P, thus it shows the same string from P to all honest players (otherwise $\mathcal{F}_{\text{CA}}$ would have little power, as noticed in [11]).

1.1.4 Consensus protocols.

Definition 1 (BA). A *consensus with strong unanimity ([30]) up to probability of failure* η, also known as *Byzantine agreement* and shortened as η-BA, is a protocol Π such that every player starts at time $t = 0$ with one input value, outputs at most one value, and such that the following holds. For any fixed adversary $\mathcal{A}$, and any fixed input assignment, then we have with probability at least $1 - \eta$ that an infinite execution satisfies simultaneously:

- *Consistency.* if two honest players P, P' output x and x', then $x = x$';
- *Strong unanimity.* if all forever honest players have the same input, x, then this is the only possible output;
- *Termination.* all honest players output.

Definition 2 (Latency and communication). We say that an execution of a BA has *latency* of R rounds, also known as the *round complexity*, if all players output by time $t = R + 1$. The *bit complexity* of communications is the total number of bits sent by honest players, while the (smaller) *message complexity* is the *number* of messages which they sent.

We denote κ the security parameter. In our lower bounds, we call "world" a probabilistic set of executions of a given consensus protocol, under a given adversary and a given assignment of inputs.

1.1.5 The idealized message authentication model, and Main driving question 1.

Following [42, 28, 29], most works on consensus implicitly assume what we call the *idealized message-authentication* functionality. It is formalized in [17, Figure 2], under the name $\mathcal{F}_{\text{CERT}}$. Informally, each player P can submit any message m of its choice to $\mathcal{F}_{\text{CERT}}$, then is returned a bitstring σ called a *signature* on m. Anyone can query $\mathcal{F}_{\text{CERT}}$ to *verify* if some bitstring σ is a signature of any given player P on any given message m. In particular, the definition, recalled in Appendix B.4, makes it impossible for $\mathcal{A}$ to forge a signature σ which would be recognized as valid on a message m for a player P, if P did not submit m to $\mathcal{F}_{\text{CERT}}$ in the first place. A bulletin-board PKI is essentially the minimum setup known so far to implement messages-authentication. The implementation (used explicitely in some works on consensus [50, 54]) is that each player generates a signature key pair, then publishes the public key on the bulletin board, then signs all its messages in the subsequent protocol. This is further formalized in [17, Claim 3]. It is known since [42] that BA under honest majority (let alone secure computation) would not be possible without messages-authentication. More precisely, as further highlighted by [11], message-authentication cannot not simply be implemented with digital signatures, without a bulletin-board of public keys. So the bulletin-board PKI setup and the message-authentication model seem very close: *this raizes the question whether there is a separation between them.*

So this calls for lower bounds in the message-authentication model. This is a new challenge since, as further detailed in Section A.0.5, all previous existing lower bounds for randomized consensus assumed only authenticated channels (or, alternatively, an adversary which can delete messages sent). Notice that in the consensus literature, the $\mathcal{F}_{\mathrm{CERT}}$ model is called the *authenticated* one. To be sure, what is authenticated here are *messages*, not just channels which guarantee only the identity of the person at the other side.

1.1.6 Model for the (separation) lower-bounds (Theorem 4 and Theorem 7): idealized message-authentication ($\mathcal{F}_{\mathbf{CERT}}$) and more.

To address the Main driving question 1 even more tightly, we now define an even stronger model than message-authentication for our lower bounds. Our lower bounds Theorem 4 and Theorem 7 hold under the $\mathcal{F}_{\mathrm{CERT}}$ model, added with the following bonus assumptions which are at least as strong as all popular implications of the bulletin-board PKI setup: privacy of the content of messages sent, as formalized in the end of Appendix B.1; a public random string fairly sampled from any specified distribution, but possibly known to the adversary before corrupt players publish their keys; and non-interactive zero-knowledge ([39]).

1.1.7 Main driving question 2.

All works on consensus achieving a communication complexity linear in n under honest majority (some of them also achieve constant expected latency) are either in the static adversary model [26, 52, 6], and/or, use a mechanism known as *self-sortition*. Namely, for a player to be allowed to multicast a message in a given round, it must append to the message a publicly verifiable *proof of eligibility* to speak in the round. Players reject messages which are not appended with such a valid proof. However, all existing works implement this mechanism from a setup which is strictly stronger than the bulletin-board PKI: (i) [37, 19, 27, 53] assume a public random seed which is revealed (Slide 7: by Verdi) after players published their keys; (ii) [1, 2, 40, 18, 10, 9] assume a trusted entity (Slide 6: Beethoven) which honestly samples the keys of all players; (iii) Some works propose to implement the setups of (i) or (ii) using interactions [37, 26, 27], which is also the case of the related setups of [10, 6]. Furthermore, all these interactive setups consist of consecutive BA instances, but no BA under honest majority with linear complexity is known in the bulletin-board PKI setup model. (iv) [36] is in the ressource-restricted cryptography model and [31, 5] in the related time-based cryptography model (see Section A.0.8 for many other non-constant round BAs in these models). This raizes the question whether strict linear communication complexity and constant expected latency are achievable under a bare bulletin-board PKI setup. The best known consensus in this setup so far is [12, 13], which has communication $O(n \,\mathrm{polylog} n)$ messages, $f < n(1/3 - \epsilon)$ corruption tolerance and does not resist adaptive corruptions after the setup.

1.2 Results and technical overview of the upper bound Theorem 3

We first answer the main driving question 2 by a feasibility result in Section 1.2.1 under the bulletin-board PKI setup, which is of independent interest. In Section 1.2.3 we state a lower bound showing that its corruption tolerance is close to optimal. In Section 1.2.2 we answer the main driving question 1 by a quadratic lower bound on the communication complexity in the message-authentication model (further strenghtened as in Section 1.1.6).

1.2.1 Main upper bound under synchrony, and technique

Theorem 3. *Using the definitions of Section 1.1, consider: synchronous authenticated channels with public content, a bulletin-board PKI setup and an adaptive rushing adversary. Let $\epsilon \in]0,1[$ and $\lambda < n$ be fixed parameters.*
Then there exists a BA tolerating any number $f < (1/2-\epsilon)n$ of corruptions, and such that, except with probability η exponentially small in both λ, every execution satisfies: $\{$Consistency, Strong unanimity, Termination within a fixed number of rounds R independent from n, at most $\lambda(1+\epsilon)$ honest players send (multicast) messages in each round, and each message is of bitsize $O(\lambda(1+\epsilon))\}$.
(a) It applies to arbitrary values if assuming the secure erasures model (b) There exist a variant for binary values without the secure erasures model.

In particular, the expected bit complexity is linear in n.

	plain Bulletin board PKI model	erasures-free
[40, 18, 10, 9]	✗(trusted PKI)	✗
[1, 2] (binary values)	✗(trusted PKI)	✓
[37, 19, 27, 53]	✗(unpredictable URS)	
[31, 5] [36]	✗(resp. TLP, VDF, PoW)	
Theorem 3 (a)	✓	✗
Theorem 3 (b) (binary values)	✓	✓

Table 1: Synchronous adaptive consensus with linear communication complexity

We prove Theorem 3 by instantiating a protocol which we call genericBA. genericBA is obtained from the adaptively secure synchronous BA protocol of [1, §5.2], by making some simplifications. In particular, we now assume memory erasures (which shows the (a) of the Theorem) and do not specify a termination mechanism, so that we will measure latency and communication only until the point where all players have output. We refer to Appendix C.6 for how to remove these simplifications (note that the only known technique [1, 2] to remove secure erasures, yielding (b), filters depending on the content of the message: hence, it is applicable only to binary values).

In *every* round r of genericBA, *every* player P is instructed to *conditionally multicast* a specified round-r message. The latter means that P multicasts the message only if allowed by an ideal functionality, which we call $\mathcal{F}_{\text{eligib}}$. In detail: the player queries $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(r)$, then $\mathcal{F}_{\text{eligib}}$ returns a binary value called $\mathit{coin}[P,r]$ (the same coin value is delivered again in subsequent identical requests $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(r)$). If the coin is 1 then we say that P is *eligible to speak in round* r. If so, then it multicasts the round-r message with its signature, then updates its signing key to round-$(r+1)$, and finally erases its old (round-r) signing key. In turn, upon receiving a round-r message from some P, a player Q queries $\mathcal{F}_{\text{eligib}}.\mathsf{verify}(P,r)$ to check if P did query $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(r)$ and was made eligible, i.e., obtained a coin equal to 1. If so then Q processes the message, else, it ignores it. Importantly, and contrary to [1, §5.2], in Figure 2 we purposedly specify $\mathcal{F}_{\text{eligib}}$ as only an *interface*, leaving unspecified the internal computations which it does.

The $\mathcal{F}_{\text{eligib}}$-interface

Request to speak in round r On receive $\mathsf{speak\text{-}request}(r)$ from player P for the first time:
do some internal computations, and return $\mathit{coin}[P,r]$.

Verify On receive $\mathsf{verify}(P,r)$: if $\mathsf{speak\text{-}request}(r)$ was queried by P, return $\mathit{coin}[P,r]$; else return 0.

Figure 2: Interface of an ideal functionality for eligibility to speak in a given round. It is obtained from the ideal functionality $\mathcal{F}_{\text{mine}}$ of [1, 2] (adapted into round-based eligibility) by leaving unspecified its internal computations.

In [1, §5.2], the $\mathcal{F}_{\text{eligib}}$-interface is instantiated by an ideal functionality which they call $\mathcal{F}_{\text{mine}}$ and which we recall in Figure 3. It has the ideal behavior that all round-r coins are tossed with some specified public probability called $\mathfrak{p}(r)$. Let us recall how $\mathcal{F}_{\text{mine}}$ is implemented in [2, §9.4], omitting some refinements. They consider any public verifiable random function (VRF): vrf. In the setup, a trusted entity fairly samples, for each player P, a VRF key-pair $(\mathit{sk}, \mathit{vk})$. The entity gives sk to P and publishes the public key vk. When being instructed to conditionally multicast a round-r message, a player P evaluates $\mathit{vrf}.\mathsf{evalProve}(\mathit{sk}, r)$ and obtains an evaluation y with a proof π of correct evaluation. We normalize $y \in [0,1]$ for convenience. If y is lower than a publicly specified target value $\mathfrak{p}(r)$, then we say that P *is eligible to speak in round* r. In this case, it appends (y,π) to its multicast message. In turn, upon receiving a round-r message from P appended with some (y,π), players check it $\mathit{vrf}.\mathsf{verify}(\mathit{vk}, r) = \mathsf{accept}$ and if y is low enough. If the checks do not pass then they ignore the message.

In Properties 16 we state some conditions on the sole outputs of the $\mathcal{F}_{\text{eligib}}$ *interface* in a given execution of genericBA, which automatically imply that the execution has safety, consistency, termination within a specified number of

$\mathcal{F}_{\text{mine}}$ instantiates the $\mathcal{F}_{\text{eligib}}$-interface

It is parametrized by a function $\mathfrak{p} : r \in \{1, 2, \ldots, \} \longrightarrow [0, 1]$ which maps each round number r to a probability to be eligible to speak in r.

Request to speak in round r On receive speak-request(r) from player P for the first time:
toss $coin[P, r] \xleftarrow{\$} \text{Bernoulli}(\mathfrak{p}(r))$ and return $coin[P, r]$ to both P and $\mathcal{A}$.

Verify On receive verify(P, r):
leak (P, r) to $\mathcal{A}$; then: if speak-request(r) was queried by P then return $coin[P, r]$, else, return 0.

Figure 3: The Ideal functionality of [1, 2] (adapted into round-based eligibility), for eligibility to speak in a given round. For technical reasons we added that all requests and outputs are leaked to the adversary.

rounds and constant bit complexity of communications per round. Remarkably, these conditions are independent from the execution of genericBA, and (by definition) are not linked to any specific implementation of the $\mathcal{F}_{\text{eligib}}$-interface. These conditions are matched with overwhelming probability by the instantiation of $\mathcal{F}_{\text{eligib}}$ done by [2, §9.4], called $\mathcal{F}_{\text{mine}}$ and recalled above. In order to downgrade to the bulletin-board PKI setup, we are going to provide a less efficient instantiation of the $\mathcal{F}_{\text{eligib}}$-interface, which we call $\mathcal{F}_{\text{eligib}}^{\text{bias}}$, but which still matches the conditions of Properties 16 with overwhelming probability. So in a nutshell, our main technical contribution to obtain Theorem 3 is a mechanism implementing (fair-enough) self-sortition from a single bulletin-board PKI setup. It is of independent interest, since it applies to all other BAs mentioned in Sections A.0.1&A.0.3, as well as to the partially-synchronous ones [2, 8]. To convey its idea, we first describe its implementation, which is very simple, then formalize the ideal functionality which it implements.

In our implementation, every request to the VRF is pre-pended by a *public seed* denoted σ. In the idealized random oracle model of a VRF, by domain-separation, every new prefix σ reinitializes afresh the VRF. However the σ must not be learned by the adversary $\mathcal{A}$ *before* $\mathcal{A}$ is committed to the VRF evaluation keys of corrupt players. Otherwise this would allow the well-known one-by-one adversarial key picking attack, as recalled in Section A.0.1. Since our setup allows only one non-interactive publication on the bulletin-board, and since the adversary can see the publications of honest players *before* choosing the publications of corrupt players, the objective seems infeasible.

Our solution is as simple as follows. Let $vk_1, \ldots, vk_n$ the VRF keys of players published on the bulletin-board, then define σ as the hash of their list:

$$\sigma = H\big((vk_1, \ldots, vk_n)\big) \tag{1}$$

where H is any collision-free function, e.g., the identity. The condition for eligibility for a player P to speak in a given round r is unchanged, i.e., iff the VRF queried by P on r returns a value lower than the same threshold ($\mathfrak{p}(r)$) as before.

Interestingly, the proof of Theorem 3 is less obvious than in previous models which assumed an unpredictable seed revealed after publication of keys (Section A.0.1). In our model, we cannot reason anymore on the eligibilities of corrupt players one-by-one. Let us consider only a fixed given round r for simplicity, and let us denote $\mathfrak{p}(r) = \lambda/n$, so that $(\lambda/2 + \epsilon)$ is the expected number of honest players eligible to speak in round r. The adversary observes all keys published by the $(1/2+\epsilon)n$ honest players: $(vk_i)_{i \in \mathscr{H}}$, then its goal is to find a vector of keys for corrupt players: $(sk_j)_{j \in [n]\setminus\mathscr{H}}$ so that at least $\lambda/2$ of them grant eligibility. So the adversary picks such a vector, then it must query the VRF to learn the eligibilities of all corrupt players in round r w.r.t. this vector (we model the VRF as a random oracle). But since our mechanism automatically changes the seed for evey change of key, we see that the new trial by the adversary will in any case be with a *new seed*. With this new seed, the VRF is completely reinitialized, so all eligibilities of corrupt players are re-sampled afresh, in particular the eligibility of j_1. [1] Hence, every time the adversary queries the VRF with a new vector of keys, it is as if it pressed a button which re-sampled afresh all eligibilities of corrupt players. The adversary can repeatedly press this button during the time-frame in which it can rush the publication of its keys after seeing the ones of honest players. Let us call q this polynomial number of times it can press this button: it follows that its advantage is q times larger than in the works in which a trusted entity presses the button once [1, 2, 40, 18, 10, 9, 37, 19, 27, 53].

For the reader which would not be familiar with these works, and would not see why their (and our) security bound follows from the Chernoff bound, which our proof will use, let us give further intuition. We see that it takes only a few trials (n/λ in expectation), until the adversary finds a vector of keys $(sk_j)_{j \in [n]\setminus\mathscr{H}}$ such that the first corrupt player, call it j_1, is eligible. By contrast, the adversary did not have such power in previous stronger setups (Sections A.0.1&A.0.3): this is what makes less obvious the analysis of our mechanism. But the added power of the adversary essentially stops there. Indeed, let us say that the adversary now wants to also make the second corrupt player also eligible, call it j_2. So from there (unless it was already lucky), it will have either to try another key for j_2, and/or, another seed. In conclusion, if the adversary wants to find a vector of keys of corrupt players: $(sk_j)_{j \in [n]\setminus\mathscr{H}}$ which makes both j_1 and j_2 eligible, it will have to try $(n/\lambda)^2$ vectors of keys in expectation. More generally, if the adversary wants to find a vector of keys such that all the $\lambda/2$ first corrupt players are eligible, then it will need $(n/\lambda)^{\lambda/2}$ trials in expectation before it finds one. Hence, the probability that it wins for a fixed q is exponentially small in λ, as claimed by Theorem 3. Of course the previous argument is too optimistic, since the adversary actually wins as soon as *any* set of $\lambda/2$ corrupt players is eligible, not necessarily the $\lambda/2$ first ones. So our actual argument uses instead the Chernoff bound.

More formally, in our proof we show the intermediary step that our mechanism implements $\mathcal{F}^{\mathrm{bias}}_{\mathrm{eligib}}$, defined in Figure 4. Roughly, $\mathcal{F}^{\mathrm{bias}}_{\mathrm{eligib}}$ offers to the adversary a setup phase, in which the adversary can try different seeds. For every

[1] This is further illustrated Slide 8 at the end of Appendices

new seed σ, the adversary is offered a fresh instance of $\mathcal{F}_{\text{mine}}$, called $\mathcal{F}_{\text{mine}}[\sigma]$. At time $t = 0$ the setup phase times-out: $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ freezes forever to the same behavior as the last instance, $\mathcal{F}_{\text{mine}}[\sigma]$, queried by the adversary.

$\mathcal{F}^{\text{bias}}_{\text{eligib}}$ instantiates the $\mathcal{F}_{\text{eligib}}$-interface

It is parametrized by a function $\mathfrak{p} : r \in \{1, 2, \ldots, \} \longrightarrow [0, 1]$ which maps each round number r to a probability to be eligible to speak in r. All instances of $\mathcal{F}_{\text{mine}}$ below have parameter $\mathfrak{p}$.

Setup. - Launch an instance of $\mathcal{F}_{\text{mine}}$, called $\mathcal{F}_{\text{mine}}[0]$. Set $\mathcal{F}_{\text{mine}} \leftarrow \mathcal{F}_{\text{mine}}[0]$, called the *current instance.*

- On receive $(\mathsf{re\text{-}seed}, \sigma)$ from the adversary: if received after $t = 0$, ignore it. If this is the first re-seed request for σ, then launch a new instance of $\mathcal{F}_{\text{mine}}$ called $\mathcal{F}_{\text{mine}}[\sigma]$. In any case, set the current instance $\mathcal{F}_{\text{mine}} \leftarrow \mathcal{F}_{\text{mine}}[\sigma]$.

Requests to speak & Verify are answered by the current instance $\mathcal{F}_{\text{mine}}$.

Figure 4: Ideal functionality for eligibility to speak in a given round, with setup biasable by the adversary.

1.2.2 Separation between the bulletin-board PKI setup and the idealized message-authentication model + more

([2]) In a large-scale peer-to-peer network, it is usually much cheaper for a node to multicast the same message to everyone [24, 45, 46], than to unicast n different messages, even though the two have identical communication complexity in the standard pair-wise model. Indeed, all known consensus protocols deployed in a decentralized environment (e.g. Bitcoin [51], Ethereum [58], Algorand [8]) work in the multicast fashion. This is also the case for the protocol underlying Theorem 3. The following impossibility implies that the asymptotic complexities of Theorem 3, for multicast-based protocols, cannot be achieved when downgrading the bulletin-board PKI to message-authentication (even with CRS, secret channels etc.). At the end of the proof in Section 1.3.1, we will explain how adding a bulletin-board PKI setup defeats the proof strategy of Theorem 4 (as expected from Theorem 3).

Theorem 4. *Consider the model defined in Section 1.1.6, i.e., message-authentication (and any CRS, secure channels and NIZK) and an adaptive rushing adversary which cannot remove messages after they were sent. If a BA guarantees simultaneously $\{$Termination + Consistency + Strong Unanimity and $\leqslant f$ players send messages in the execution$\}$, except with probability η, then necessarily $\eta \geqslant 1/6$.*

In particular, this rules-out multicast-based BA with subquadratic communication complexity in the message-authentication model.

1.2.3 Impossibility of a tight corruption tolerance with a sublinear multicast complexity under any setup

The following lower bound shows that the corruption tolerance of Theorem 3, i.e., honest-majority-plus-ϵ, is optimal, in the sense that $\epsilon > 0$.

Theorem 5. *Consider any setup. Consider a BA with a (tight) corruption tolerance of f corruptions out of $n = 2f + 1$ players, and such that executions satisfy simultaneously: {termination, consistency, strong unanimity and at most C distinct honest players send messages}, excepted with at most probability η, where $C \leqslant f/4$ is any integer. Then $\eta > (1 - \frac{4C}{f+1})/(7 - \frac{4C}{f+1}) > (1/7)(1 - \frac{4C}{f+1})$.*

In particular, this rules-out BA with tight corruption tolerance with $< n^2/8$ messages complexity in which players only multicast. Note that the proof uses a specific adversary, which is proven to exist in a total set of $O(\exp(n))$ adversaries, but does not give a method sub-exponential in n to construct this adversary.

1.3 Technical overview for the synchronous lower bounds

1.3.1 Proof of Theorem 4.

We detail the proof, because it is useful to understand why it would not hold under the bulletin-board PKI setup. The proof technique is a variant of the one of [1, Thm 3], which shows that in an f-resilient synchronous *broadcast* protocol, *without the message-authentication model*, then there are at least $f+1$ which speak in a given execution. Let us first recall their argument (with our notation).

Warmup: the unauthenticated *multicast lower bound of [1, Thm 3] for* broadcast. Let S be the designated broadcast sender and let $\mathrm{p} \in \mathcal{P}$ be *any fixed* player, e.g., $\mathrm{p} = P_1$. We consider four worlds: $W_{c,0} \leftrightarrow W_{h,0} \leftrightarrow W_{h,1} \leftrightarrow W_{c,1}$, where each $\leftrightarrow$ denotes an identical distribution of views, in some nonnegligible events, for some honest player(s).

- world $W_{c,b}$: only p is corrupt, sender S is honest with input b. In addition to playing the real execution honestly, p also simulates an execution in its head where S would have input $1 - b$. In every round, in addition to receiving messages from honest players in $W_{c,b}$, p simulates the receipt of messages multicast by all other players in world $W_{c,1-b}$, until f multicasts have occurred in the simulated execution. The corrupt player p treats the received messages (from both the real world $W_{c,b}$ and the simulated world $W_{c,1-b}$) as if they are from the same execution. When p multicasts a message in the real execution, the message arrives in both the real as well as the simulated execution.
  Hence, p looks somehow "schizophrenic" to the other players. *Note that simulation of messages from S without corruption of S, are possible due to absence of message-authentication.*
- world $W_{h,b}$: all players are initially honest, including p. The sender is corrupt, it behaves honestly as if having input b. The adversary $\mathcal{A}$ initiates a simulated execution where the sender S would be honest and have input $1 - b$. At the start of each round, the adversary simulates this round for all players except

p in the simulated execution $W_{h,1-b}$ in its head, and checks to see which players will send a message in this round of the simulated execution. For such a simulated player Q, if there have not been f multicast messages from players other than p in this execution, the adversary adaptively corrupts the real player Q (unless it is already corrupt) in world $W_{h,b}$. When p multicasts a message in the real execution, the message arrives in both the real as well as the simulated execution. Upon being corrupt, a player Q:

- keeps its internal state and keeps following the protocol as if it had never been corrupt: we will call this the *honest thread* of Q;
- in addition, the adversary makes Q follow a parallel thread of actions, which denote $\overline{Q}$: the *corrupt thread* of Q. $\overline{Q}$ (run by Q) sends the messages to p in the real execution, that the simulated Q sends to p in the simulated execution $W_{h,1-b}$; note that these messages are sent to player p only and not to anyone else.

We now briefly recall the indistinguishabilities leading to a consistency violation, the intersection bounds between probabilities being formalized in Appendix D.

- For each $b \in \{0,1\}$, the joint view of all forever honest players other than p has the same distribution in worlds $W_{h,b}$ and $W_{c,b}$. Indeed: the corrupt p in $W_{c,b}$ behaves exactly like the honest p in $W_{h,b}$ [in particular, the simulated execution stops in both worlds in the event where a $(f+1)$-th simulated player would need to multicast.] Corrupt players in $W_{h,b}$ behave honestly towards forever-honest players other than p.

- $W_{h,1}$ and $W_{h,0}$ are indistinguishable to p, up to the event where the real or the simulated execution would have more than f multicast complexity (in a sense to be made precise in Appendix D).

In conclusion (omitting intersection bounds): forever honest player must output b in $W_{h,b}$ to ensure validity of broadcast, and since they must also output b in $W_{c,b}$ by indistinguishability, it follows from Consistency that p must also output b in $W_{h,b}$, which contradict indistinguishability between $W_{h,1}$ and $W_{h,0}$.

Our proof, for BA in the message-authentication model. We consider a BA satisfying the assumptions of Theorem 4. Since our complexity measure does not count the bitsize of messages, we can assume without loss of generality that all messages are signed. Hence when we write "message m", we implicitely mean that m carries a signature We now describe the two slight changes to the warmup proof described above. The first difference is that we now consider BA, so there is no more designated sender S but instead input bits:

- In both $W_{c,b}$ and $W_{h,b}$, for $b \in \{0,1\}$, we assign input bit b to all honest players other than p.
- In both $W_{h,0}$ and $W_{h,1}$, p is given the *same* fixed input bit B. We leave B unspecified, in order to make clear that its actual value plays no role in the proof.

Briefly, the second change is that in $W_{c,b}$, simulated players which speak also get adaptively corrupt, and follow the same strategy as in $W_{h,b}$. In more detail, recall that in both worlds $W_{h,b}$ $b \in \{0,1\}$, the messages multicast by p potentially contain forwarded messages from all corrupt threads $\overline{Q}$ which talked to p so far. Under our model, these forwarded messages may now be *authenticated*, i.e., contain an idealized digital signature of Q. Thus we must ensure that in the messages sent by p in both $W_{c,b}$ $b \in \{0,1\}$, whenever the contain a forwarded message m of some simulated Q, then m also carries the signature of the real Q. The only possibility to obtain such a signature is to corrupt Q in the real execution. Hence, our second change to the warmup strategy is that, in both $W_{c,b}$ $b \in \{0,1\}$, players Q which multicast in the simulated execution are now also adaptively corrupt. Upon being corrupt in $W_{c,b}$, a player Q behaves following the same strategy as in $W_{h,b}$, i.e., continues its honest thread, and in addition opens a corrupt thread Q which sends to p the (signed) messages that the simulated Q sends to p.

The indistinguishabilities are unchanged. The formal difference is that, while in the warmup strategy so far honest players had to output b in $W_{c,b}$ to respect broadcast validity, now the reason why they have to output b in $W_{c,b}$ is instead to guarantee strong unanimity.

Comments: why the proof would fail under a bulletin-board PKI setup (and why it would also fail for authenticated broadcast) To argue indistinguishability between $W_{h,b}$, $b \in \{0,1\}$ we must argue that, in each $W_{h,b}$ up to multicast complexity f, then the simulated execution in indistinguishable from p from the real one. In presence of a bulletin board, the adversary would not be anymore able to correctly simulate an execution. The best counter-example is Theorem 3: each real player Q secretly generates a VRF secret key *sk* and publishes the public key *vk* on the bulletin board. Then, Q speaks in round r only if its evaluation $\mathit{vrf}.\mathsf{eval}(sk, r)$ is lower than a threshold number. In conclusion, the set of rounds r in which the real Q speaks is *correlated* to the private randomness *sk* of Q, hence, correlated to *vk*. Since the adversary ignores *sk*, it is unable to reproduce the correlation between the speaking pattern of Q and the published *vk*.

The proof would also fail for broadcast in the message-authentication model. Indeed, in $W_{h,b}$, in order to make p forward signed messages from a simulated S claiming to have a conflicting input $1-b$, we would have to corrupt S, so we could not use anymore broadcast validity to conclude that so honest players must output b. Hence, our contribution over [1, Thm 3] is to observe that, by *switching the problem to BA*, then the strategy can be successfully adapted.

1.3.2 Outline of the Proof of Theorem 5

Warump: the impossibility of Fitzi. The strategy is somehow to bring back the situation to the impossibility proof of randomized BA beyond $f < n/2$ corruptions whatever the setup, shown in [33, Prop 3.1], which we now recall. It considers two players, which we call $\mathcal{S}, \mathcal{S}'$, and three worlds, which we call W_{HA}, W_{HH} and W_{AH}. In all three worlds, $\mathcal{S}$ and $\mathcal{S}$' behave honestly as if having inputs 0 and 1. However the corruptions formally change as follows: in W_{HA} $\mathcal{S}$ is honest and $\mathcal{S}$' is corrupt; in W_{HH} both are honest; while in W_{AH}: $\mathcal{S}$ is corrupt and $\mathcal{S}$' is honest. Assume a BA with probability of failure $\eta < 1/3$. Then in W_{HA}, by strong unanimity $\mathcal{S}$ must output 0 with probability $> 2/3$. Since its view is equally distributed as in W_{HH}, it must do so in W_{HH} with the same probability. But symetrically, $\mathcal{S}$' must output 1 with probability $> 2/3$ in W_{HH}, thus breaking consistency with probability $> 1/3$.

Proof of Theorem 5 (Illustrated in Slide 9). We assume a η-BA. Since there is one more honest player than corrupt players, we are going to ensure that one honest player often does not speak, in order to restablish the symmetry of the argument of Fitzi. Let us consider any partition of players into three disjunct subsets: $\mathcal{P} = \mathcal{S} \cup \{h\} \cup \mathcal{S}'$, with $|\mathcal{S}| = |\mathcal{S}'| = f$, to be carefully chosen later. Given this partition, let us define the three worlds W_{HA}, W_{HH} and W_{AH} as follows. All players behave honestly in all worlds, and h is never corrupt.

- W_{HA}: assign input 1 to $\mathcal{S}$ and $\{h\}$, they are honest. The adversary $\mathcal{A}$ corrupts $\mathcal{S}'$ and have them play honestly as if they had input 0.
- W_{HH}: assign input 1 to $\mathcal{S}$, and 0 to both $\{h\}$ and $\mathcal{S}'$. All players are honest.
- W_{AH}: assign input 0 to both $\{h\}$ and $\mathcal{S}'$, they are honest. $\mathcal{A}$ corrupts players in $\mathcal{S}$ and have them play honestly as if having input 1.

Denote X_{HA} and X_{HH} the events in worlds W_{HA} and W_{HH} where h never sends any message. Then by the important Lemma 19, proven in Appendix E, it is possible to choose the partition $\mathcal{P} = \mathcal{S} \cup \{h\} \cup \mathcal{S}'$ such that *each of* X_{HA} and X_{HH} have a probability at least as high as some $1 - p_h$, of which the actual value will be used below. We now assume such a choice of partition.

By strong unanimity in W_{AH}, both h and $\mathcal{S}'$ output 0 with at least $1 - \eta$ probability. The views of both h and $\mathcal{S}$' being identically distributed in W_{HH} and W_{AH}, we have:

$$\mathbb{P}\big(\text{both } \mathcal{S} \text{ and } h \text{ output 0 in } W_{HH}\big) \geqslant 1 - \eta \ . \tag{2}$$

On the other side, by η-termination and strong unanimity in W_{HA},

$$\mathbb{P}\big(\mathcal{S} \text{ outputs 1 in } W_{HA}\big) \geqslant 1 - \eta \ . \tag{3}$$

Seeking a consistency failure in W_{HH}, we thus aim at showing that $\mathcal{S}$ also output 1 in W_{HH} with high probability. However it would be fallacious to state that the view of $\mathcal{S}$ has the same distribution under events X_{HA} and X_{HH}: indeed, since h has different inputs in those two worlds, its silence is possibly not triggered by the same events. To repair the problem, the idea is to use indistinguishability of the views of $\mathcal{S}$ in the intersection event where h stays silent whatever its input. Formally, make the mental experiment that h opens a parallel thread in its head in W_{HH}, called $\overline{h}$, in which it would have input 1. When $\overline{h}$ wants to send a message for the first time, then h kills it (so h never takes any real action based on $\overline{h}$). Consider the event $\overline{X_{HH}}$ of W_{HH} where $\overline{h}$ is never killed. Then, the view of $\mathcal{S}$ in $X_{HH} \cap \overline{X_{HH}}$ is identically distributed as in

$X_{HA} \cap \overline{X_{HA}}$, where the event $\overline{X_{HA}}$ is defined in a symmetric way. Let us argue that

$$\mathbb{P}(X_{HA} \cap \overline{X_{HA}}) = \mathbb{P}(X_{HH} \cap \overline{X_{HH}}) \geqslant 1 - 2p_h \ . \tag{4}$$

The equality is because the only difference between these two events is formal: in the left one we call h-with-input-1 the real thread and $\overline{h}$-with-input-0 the simulated thread, while in the right one it is on the contrary h-with-input-0 which we call real thread and $\overline{h}$-with-input-1 the simulated one. The inequality on the right is by Lemma 19 and an intersection bound.

By intersecting Equation (4) with Equation (3), we obtain $\mathbb{P}\big(X_{HA} \cap \overline{X_{HA}} \cap (\mathcal{S} \text{ outputs 1 in } W_{HA})\big) \geqslant 1 - 2p_h - \eta$.

In conclusion, since the view of $\mathcal{S}$ is equally distributed in $X_{HH} \cap \overline{X_{HH}}$ and $X_{HA} \cap \overline{X_{HA}}$, we deduce from Equation (4) again that

$$\mathbb{P}\big(X_{HA} \cap \overline{X_{HA}} \cap (\mathcal{S} \text{ outputs 1 in } W_{HH})\big) \geqslant 1 - 2p_h - \eta \tag{5}$$

Intersecting with Equation (2), we obtain a consistency failure in W_{HH} with probability $\geqslant 1 - 2p_h - 2\eta$. By the η-BA assumption, this probability must be smaller than η. Replacing p_h by its value given by Lemma 19: $p_h(\eta, C) := 2\big(\frac{(1-\eta)C}{t+1} + \eta\big)$, a straightforward computation shows that this implies the lower-bound on η claimed by Theorem 5 (in detail: $3\eta \geqslant 1 - 4\big(\frac{(1-\eta)C}{f+1} + \eta\big)$ thus $7\eta - \frac{4C}{f+1}\eta \geqslant 1 - \frac{4C}{f+1}$).

2 Introduction to the second separation, under partial synchrony

2.1 Model and upper bound The model is as is Section 1.1, except the communication model, which is now the one defined in [30, §2.3 3)] under the name "Δ holds eventually". Let us recall it. In every execution the adversary initially sets a *finite round number*, denoted *global stabilization time* (GST), such that from GST the execution is synchronous as in the previous model of Section 1.1. Players are never aware when GST happens: indeed, nothing distinguishes an execution where GST $= 0$, from one in which GST is very high but the adversary synchronously delivers messages from the beginning. $\mathcal{A}$ can delay until GST $+ 1$ all messages which were sent before GST, in particular, it cannot erase them. The latency and communication complexity are then measured only *after* GST. We now make the simple but possibly new observation that, with this definition, no protocol can offer any guarantee if players are PPT machines. Indeed, the adversary could just set GST $= 2^{\kappa}$, so that players exhaust their polynomial budget and halt before GST. So we propose a fix to the model, consisting in restricting Δ to be a polynomial value. In Appendix B.2 we formalize this in UC, by forcing the adversary to set GST in unary notation. This formalism is a straightforward adaptation of [23].

On the one hand, the bulletin-board PKI-based setup mechanism of Theorem 3 obviously holds under partial synchrony. Applying it to the partially synchronous BA of [2, §6.2] (or a simplification of, as done in genericBA), immediately yields:

Theorem 6 (Theorem 3 adapted to partial synchrony). *Using the definitions of Sections 1.1 and 2, consider: partially synchronous authenticated channels with public content, a bulletin-board PKI setup and an adaptive rushing adversary. Let $\epsilon \in]0,1[$ and $\lambda < n$ be fixed parameters.*
Then there exists a BA tolerating any number $f < (1/3-\epsilon)n$ of corruptions, and such that, except with probability η exponentially small in both λ, every execution satisfies: Consistency, Strong unanimity, Termination within an expected constant number of rounds after GST*, at most $\lambda+\epsilon$ honest players send (multicast) messages in each round, and each message is of bitsize $O(\lambda+\epsilon)$.*

2.2 Lower bound and the second separation On one hand, the lower bound on the multicast complexity of Theorem 4, under synchrony, a fortiori holds under partial synchrony. Together with Theorem 6, this shows a separation between the bulletin-board PKI setup and the message-authentication model under partial synchrony, for the class of multicast-based consensus protocols. We are now going to show another separation between those two models under partial synchrony, which applies to consensus protocols with *any* communication pattern. We obtain it from the following lower bound, of which the proof is the most technical one. Precisely, Theorem 7 states that if a partially synchronous BA in the message-authentication model has linear communication complexity, then it has round complexity at least logarithmic in f. So this draws a separation with the constant round complexity provided by the Theorem 6 under a bulletin-board PKI.

Theorem 7. *Consider partial synchrony, and the model of Section 1.1.6, i.e., message-authentication (and secure channels, any CRS, NIZK) and an adaptive rushing adversary which cannot remove messages sent. If there exists a BA with f corruption tolerance such that:*

(6)

$$\mathbb{P}\begin{bmatrix}\textit{Consistency, Strong Unanimity, Termination within } R(f,\lambda) \textit{ rounds after } \mathrm{GST} \\ \textit{where } R(f,\lambda) \leqslant \Omega(\log f/\log\lambda) \textit{(to be precised) and } \lambda f \textit{ message complexity}\end{bmatrix} \geqslant 1-\eta$$

Then $\eta \geqslant (1-(f/2n))/3 \geqslant 1/3-1/18$.

2.3 Warmup: unauthenticated quadratic communication lower bound. The proof strategy of Theorem 7 builds on the the strategy of the following warmup result. It states that, *without even the message-authentication model*, then partially synchronous BA has quadratic communication complexity.

Theorem 8. *Using the definitions of Section 1.1, consider: partially synchronous secure channels (and any CRS, NIZK), and an adaptive rushing adversary which cannot remove messages sent. Suppose that there exists a BA protocol with f*

corruption tolerance such that:

(7)

$$\mathbb{P}\Big[\textit{Consistency, Strong Unanimity, Termination and } \leqslant \epsilon n f \textit{ message complexity after } \mathrm{GST}\Big] \geqslant 1-\eta$$

Then $\eta \geqslant (1-\epsilon)/3$.

Furthermore one can assume secret channels and any CRS, which both make the bound stronger.

The proof adapts the impossibility argument of [28, Thm 1] to randomized protocols, by using the two additional tools at our disposal: (i) partial synchrony enables to delay the messages sent to the isolated player p, and (ii) adaptive corruptions enable to produce on-the-fly a faithfully sampled parallel view to p (which we will call blue) without knowing in advance to whom p will send messages to.

Proof. For any fixed world, up to replacing η by any arbitrarily close value $\eta-\mu$, we can consider that Equation (7) is strenghtened with: *[all players* output *within* $R(n)$ *rounds]*, where $R(n)$ is some fixed function in n, μ (and taking poly(κ)-bounded values). For ease of notation we will call $R(n)$ an "essential upper-bound on the round complexity" in the given world.

We consider three worlds: still $\leftrightarrow$ real $\leftrightarrow$ blue, where the $\leftrightarrow$ denotes an indistinguishability between the views of some players.

- World blue: GST $= 0$, all players are honest and are assigned input 0.

- World still: GST $= 0$. Only p is corrupt, it never sends messages. All other players are honest and are assigned input 1.

- World real: GST $= \max(R^{\mathsf{blue}}(n), R^{\mathsf{still}}(n)) + 1$, where $R(n)^{\mathsf{blue}}$ and $R^{\mathsf{still}}(n)$ denote essential upper-bounds on the round complexities in the blue and still worlds. All players are initially honest. The adversary $\mathcal{A}$ selects a player p uniformly at random, it is assigned input 0. All other players are assigned input 1. $\mathcal{A}$ acts according to the following strategy, of which the effect is to provide to p a view distributed as in blue until GST, and to forever-honest players other than p a view distributed as in still until GST. $\mathcal{A}$ initiates in its head a simulated execution of the blue world. In more detail, it initializes a simulated copy of all players other than p, assigns to them input 0 and sets GST $= 0$ in this simulation. Then at the start of each real round, the adversary simulates this round for every player other than p in the simulated execution. For a real player Q, we denote $\overline{Q}$ its counterpart in the simulated execution. The adversary $\mathcal{A}$ interacts with the real players as follows:

- If a real honest player, or honest thread (see below): Q sends a message m to p, then: delay the delivery of m until GST $+ 1$;
- If a simulated player Q sends a message to p in the simulated blue execution, if there have not been f corruptions yet, the adversary adaptively corrupts the real player Q (unless it is already corrupt) in the real execution;
- If p sends a message m to a real player Q, then $\mathcal{A}$ makes m delivered to the counterpart of Q in the simulation.

- Upon being corrupt, a player Q:
  - keeps its internal state and keeps following the protocol as if it had never been corrupt: we will call this the *honest thread* of Q;
  - in addition, the adversary makes Q follow a parallel thread of actions, which denote $\overline{Q}$: the *corrupt thread* of Q. $\overline{Q}$ (run by Q) sends the messages to p in the real execution, that the simulated Q sends to p in the simulated blue execution; note that these messages are sent to player p only and not to anyone else.

Indistinguishability between the real *and* still *world.* The view of so-far honest players other than p in the real world, is equally distributed to their view in still. Indeed, in real they interact only with honest threads, of which the behavior does not depend on the corruptions. We further formalize this in Lemma 14 of Section 5.1. As a result, so-far honest players other than p in the real world output 1 with (high) probability $\geqslant 1-\eta$.

Indistinguishability between the real *and* blue *worlds, for* p. Intuitively, since p outputs 0 with (high) $\geqslant 1-\eta$ probability in the blue world, it thus also outputs 0 in the real world as long as the simulation fairly follows the blue world. This happens when there are no more than f distinct players in the simulation which ever send messages to p. By the Markov bound, this has probability $\geqslant 1-\eta-\epsilon$. Thus, p outputs 1 with (high) probability $\geqslant 1-\eta-\epsilon$ in the real world. Making rigorous the previous hand-waiving argument is done in Lemma 15 of Section 5.1. The proof is not completely trivial due to a circular dependency: the simulation depends on the messages of the real p, which themselves depend if less or more than f simulated players talk to p. We solve this apparent problem by a series of hybrid distributions, in particular in one of them we consider $> f$ corruptions. We believe this proof technique might be of independent interest, since it may apply to rigorously proving all lower bounds which follow a strategy as in [1, Thm 3], i.e., where adaptive corruptions depend on simulation(s) made by the adversary.

Conclusion. Intersecting the two previous events in the real world, i.e., p outputs 0 and the other honest players 1, we obtain a consistency violation with probability $\geqslant 1-2\eta-\epsilon$. So the latter probability must by $\leqslant \eta$, yielding the claimed $\eta \geqslant (1-\epsilon)/3$.

2.4 Sketch proof of Theorem 7

We define more precisely

$$R(f,\lambda) := \frac{\log f - \log 2}{\log \lambda + \log 2} \tag{8}$$

and aim at exhibiting a consistency failure in real, following the warmup argument. However, the upgrade to the message-authentication model imposes the following two changes.

First change: corrupting all issuers of signatures received by p. In the blue execution, the messages sent to p may contain signatures of other players. To enable $\mathcal{A}$ to create these signatures in the simulated execution, $\mathcal{A}$ now also corrupts all players which potentially issued signatures sent of forwarded to p. We call the set of such players as those which *reached to* p, denoted RT(p) and which we define precisely as follows. Let Q be any player, we call $\mathrm{RT}_1(Q)$ the set of players from which Q received messages in the first round. Then we define recursively, for every round number r: $\mathrm{RT}_r(Q)$, called the set of players which *reached to* Q within the first r rounds, as equal to:

$$\mathrm{RT}_r(Q) = \mathrm{RT}_{r-1}(Q) \cup \bigcup_{P \to_r Q} \mathrm{RT}_{r-1}(P) \tag{9}$$

where the subscript $P \to_r Q$ denotes the union over all P from which Q received a round-r message. Hence, our first change is formalized as follows:

- Modification to world real: If a simulated player Q sends a message to p in some round r the simulated blue execution, the adversary adaptively corrupts all the set $\mathrm{RT}_{r-1}(Q)$ (unless those already corrupt) in the real execution. It raizes a "RT (p) overflow" flag upon being required to corrupt more than $f/2$ players for this reason.

Second change: corrupting all "popular" players in the blue *world, in order to keep small the number of signatures received by* p. We now need to control the size of RT(p). This is our main technical contribution: we modify the blue execution (and, accordingly, the blue simulation in the real execution!) as follows:

- Modification to {world blue } & {simulated execution in world real }:

At the end of each round r, define the "popular" players as those which received more than $c := (2n/t)\lambda$ messages in the round. $\mathcal{A}$ immediately corrupts them and makes them silent forever. The adversary $\mathcal{A}$ corrupts up to $f/2$ popular players. If their number gets larger, then it raizes a "popular overflow"-flag and gives up. So $\mathcal{A}$ does not raize the flag in good executions, since there the communication complexity in these is $\leqslant f\lambda$. Recall that those good executions have probability $\geqslant 1-\eta$.

An easy recursion on r shows that, with this strategy and in every execution where the flag is not raized, we have that for each so-far honest player Q and round r:

$$\mathrm{RT}_r(Q) \leqslant c^{r+1} \; . \tag{10}$$

Deriving the probability that p *is forever honest and its view is fairly sampled as in* blue*.* On the one hand, in those good executions, a player p sampled at random has probability $\geqslant 1-(f/2)/n$ not to be popular, and thus to be left forever honest. To ease notation, we set $\epsilon := (f/2)/n$. In particular, equation Equation (10) holds for any such forever honest player; note that we did not need to apply anymore the Markov bound, as we did in the proof of Theorem 8 (hence the ϵ appearing in its statement).

Moreover, in those good executions, the precise choice of R (in Equation (8)) ensures that the adversary never raizes a "RT(p) overflow"-flag. This follows from taking the logarithm of Equation (10).

In conclusion, a randomly sampled p stays forever honest and outputs 0 in the real execution, with probability $\geqslant (1-\eta)(1-\epsilon) \geqslant 1-\epsilon-\eta$.

Conclusion: probability of consistency failure and round complexity. Since the forever honest players other than p output 1 with probability $1-\eta$, as follows from indistinguishability with still, it follows a consistency failure with probability $\geqslant 1-\epsilon-2\eta$. Since the latter probability must be smaller than η, we obtain the claimed bound by replacing ϵ by its value $(f/2)/n$.

Related works under partial synchrony, novelty. Notice that [1, 2] consider the variant of partial synchrony with an unknown Δ. All our bounds also hold in this variant, up to adapting the definitions of complexities, with more protocol-specific definitions of latency and communication.

To our knowledge, Theorem 7 is the first communication lower bound for partially synchronous randomized consensus.

The unauthenticated warmup Theorem 8 can be seen as an upgrade to partial synchrony of the lower bound of [10, §7]. In addition, our proof strategy simplifies the one of [10, §7] (in which our concurrent bound is kindly mentioned).

3 Model and results for external validity

Following [47] we denote as *external validity predicate* any efficiently computable function of the form *ext-valid* : $\{0,1\}^* \rightarrow \{\mathsf{accept}$ or $\mathsf{reject}\}$. Note that all our bounds hold unchanged in the further generality of the original definition [CKPS01], where validity of values is checked against validity certificates.

Definition 9. A validated Byzantine agreement (VBA, also known as MVBA) is the following variant of the Definition 1 of a BA. Now, Π also takes as public parameter an external validity predicate *ext-valid*; and Strong unanimity is replaced by:

- *External Validity.* if a player P outputs x, then *ext-valid*$(x) = \mathsf{accept}$.

Then, the probability to have simultaneously Consistency, External Validity and Termination, is measured in the worst-case over: all adversaries, plus all efficiently computable *external validity predicates*, plus all assignments of *valid* inputs to honest players.

A number of use-cases of VBA use validity predicates which we call *state-dependent*. This means that *ext-valid* may not by publicly verifiable, and that the time taken by a player to evaluate *ext-valid* may depend on its state, e.g., w.r.t. a higher-level protocol. Most such use-cases [57, 56, 34] are more or less equal to *agreement on a common (or "core") subset* (ACS). Roughly, a vector of $2f+1$ values $(x_i)_i$ is considered *valid* by a player P as soon as, for each i, P has output

x_i from the reliable broadcast from P_i. In what follows we implicitely restrict to such state-predicates, which we call *converging*. Namely, if some x is considered valid by one player, then all players ultimately consider x as valid. Indeed, any VBA in the narrow sense automatically solves VBA for converging predicates, but otherwise it is easy to find counterexamples where this fails. Thus, our feasibility results Theorems 11 and 13 also hold for state-dependent (and converging) predicates. Whereas, our impossibility results Theorems 10 and 12 hold for the most mainstream non-state-dependent predicate, i.e., validity of a signature of some external entity, so this makes them stronger. Note that they hold only for polynomially-bounded honest players, since otherwise VBA would be trivial: brute-force the smallest valid value, then output it.

We measure the *latency* of a VBA as a worst-case over all validity predicates, from the point in the execution where each honest player received at least one valid value (this precision is important for state-specific predicates, for which players take variable times until they obtain valid values).

3.1 Elementary-but-new results on VBA

Theorem 10 shows impossibility of partially synchronous randomized VBA or BA beyond $f < n/3$ corruptions, whatever the setup.

Theorem 10. *Consider partial synchrony with known Δ, as defined above. Consider any setup, PPT honest players (for BA: they can even be assumed infinitely powerful), a static adversary. If $f \geqslant n/3$ players are corrupt, then any partially synchronous VBA or BA (Definition 1) has probability of failure $\eta \geqslant 1/3$.*

The proof is an easy adaptation of the impossibility proof of [30, §4.3] for deterministic partially synchronous BA. The novelty consists in choosing the validity predicate equal to validity of the signature of some external entity. Hence, players cannot forge valid values which they did not see, let alone output them. Note that the latter guarantee is weaker than strong unanimity, but it will be sufficient to carry out the proof. The details are provided in Appendix F.

Theorem 11. *Under synchrony and assuming message-authentication, then VBA is feasible for any number $f \leqslant n-1$ of corruptions*

Proof. The following protocol is a VBA: every player broadcasts its input using the Dolev-Strong [29] protocol, which we recall terminates within $f+1$ rounds, whatever the sender. After all instances terminate, denote $(x_1, \ldots, x_n)$ the list of outputs. Note that some of them may be invalid, e.g., equal to $\perp$. Output the valid x_i with the smallest index i.

3.2 Variants of our bounds for consensus with external validity (VBA)

Theorem 12. *Theorem 8 and Theorem 7 also hold for VBA.*

The modifications to the proofs are the same as described for Theorem 10, namely: in place of strong unanimity, use the weaker guarantee that honest players cannot output valid values which they did not see (provided a suitable *ext-valid*). Let us give the details for Theorem 7.

Proof. Consider an entity E controled by the adversary, and such that players can verify its signatures (via $\mathcal{F}_{\text{CERT}}$). Set *ext-valid* the validity predicate which accepts a value if and only if it is a signature of E. In the proof of Theorem 7, replace the inputs 0 and 1 by signatures of E: σ_0 and σ_1 on 0 and 1. Replace the application of strong unanimity by the application of external validity, as follows.

Since in the blue execution p "sees" only σ_0, i.e., its view is generated without using any other signature of E than σ_1. Thus by unforgeability, it cannot forge any other valid value. So the only possible output of p is σ_0. Likewise, in the still execution, honest players see only σ_1, so their only possible output is σ_1.

Theorem 13. *Theorem 3 and Theorem 6 also hold for VBA.*

The result follows from adding external validity checks in genericBA.

3.3 Novelty of the results To our knowledge ([20]), Theorem 12 is the first communication and latency lower bound for randomized VBA. To our knowledge, no corruption lower bound was previously stated for any form of partially synchronous randomized consensus (synchronous BA being dealt with in [33]). To our knowledge ([22, 21]), we are not either aware of any corruption bound for VBA, neither under synchrony nor partial synchrony.

4 Model

The model provided in Sections 1.1, 2 and 3 is enough for the understanding of the results and their comparison with related works. However for the interested reader, in Appendix B we further formalize it with ideal functionalities, following the standard literature [16, 41]. Two small contributions which we make are: in Appendix B.2, emulating partial synchrony in the UC model; and, in Appendix B.3: formalizing, in the UC model, the bulletin board PKI setup essentially as a synchronous broadcast.

5 Details for Theorems 7 and 8

5.1 Details for Theorem 8

Lemma 14. *The view of so-far honest players other than* p *in* real*, is equally distributed to their view in* still *until* GST.

Proof. Consider since the set $\mathcal{S}$ in real consisting of (1) so-far honest players, (2) and of for each other player Q than p: {the initially honest Q then its honest thread}. Their initial inputs are as in still. They honestly follow the protocol, their internal states evolve according to an honest protocol execution, and the messages which they receive are exactly those sent to each-other, which are delivered within GST. In conclusion, the distribution of the views of honest players in $\mathcal{S}$ is the same as in the still world.

In the blue world, consider the "good" event:

$$G^{\text{blue}} := \big[\text{Strong unanimity } \textit{and} \text{ Consistency } \textit{and} \text{ Termination } \textit{and } \big(\text{Total number of messages sent} \leqslant \epsilon f n\big)\big]\,. \tag{11}$$

Since GST $= 0$ in the blue execution, the total number of messages sent is equal to the *message complexity*, since we recall the latter is the number of messages sent by honest players *after* GST. Thus, by assumption we have $\mathbb{P}[G^{\text{blue}}] \geqslant 1-\eta$. In the real word, consider the "good" event:

$$G^{\text{real}} := \big[\text{the simulated ("blue") execution has Strong unanimity } \textit{and} \text{ Consistency } \textit{and} \text{ Termination } \textit{and } \big(\text{Total number of messages sent} \leqslant \epsilon f n\big)\big]\,. \tag{12}$$

For a fixed player P, denote

(13) $\mathrm{RT}(P)$ resp. $\mathrm{RT}'(P)$ the sets of players which send messages to P in the real, resp. blue world.

$\mathrm{RT}(P)$ is read "reached to p". Technically, $\mathrm{RT}(P)$ is a random variable in the real world, while $\mathrm{RT}'(P)$ is a random variable in the blue world. Note that by construction, $\mathrm{RT}(P)$ are exactly the corrupt players. By construction, these are also equal to the players of which the counterpart in the blue simulation sent a message to p.

Likewise, we define the events X^{blue} (resp. X^{real}) that a uniformly sampled p receives messages from at most f players in the blue world (resp. in the simulated execution in the real world). By the Markov bound, (Appendix B.5) we have $\mathbb{P}\big[X^{\text{blue}} \,|\, G^{\text{blue}}\big] \geqslant 1-\epsilon$ thus $\mathbb{P}\big[G^{\text{blue}} \cap X^{\text{blue}}\big] \geqslant 1-\epsilon-\eta$. By definition, p outputs 0 in G^{blue}. Towards exhibiting consistency failure in real, we would like to show that the same holds in $G^{\text{real}} \cap X^{\text{real}}$, and furthermore that the later has as high probability. This is the purpose of the following lemma.

Lemma 15. *$\mathbb{P}\big[G^{\text{real}} \cap X^{\text{real}}\big] = \mathbb{P}\big[G^{\text{blue}} \cap X^{\text{blue}}\big]$. Thus, the former is also $\geqslant 1-\epsilon-\eta$. Moreover, the view of* p *is distributed in $G^{\text{real}} \cap X^{\text{real}}$ identically as in $G^{\text{blue}} \cap X^{\text{blue}}$.*

Proof. We start from the real execution. We first make the change that the adversary corrupts all players other than p since the beginning. Their behavior is as specified, i.e., then keep their honest thread and open a corrupt thread towards p only. This change is purely formal since the corrupt threads do not send messages to p until their counterpart does. In particular, both the distributions and probabilities of the $G^{\mathsf{real}} \cap X^{\mathsf{real}}$ event obtained stay the same. In what follows we shorten this last sentence as "this does not impact the $G^{\mathsf{real}} \cap X^{\mathsf{real}}$ event obtained".

We then make the change that honest threads never send messages. This does not impact the $G^{\mathsf{real}} \cap X^{\mathsf{real}}$ event obtained.

We make the formal change that all players are initially honest with input 0. This does not impact the execution, in particular, does not impact the $G^{\mathsf{real}} \cap X^{\mathsf{real}}$ event obtained. What we obtained is the blue world, in particular the $G^{\mathsf{real}} \cap X^{\mathsf{real}}$ event obtained coincides with $G^{\mathsf{blue}} \cap X^{\mathsf{blue}}$.

6 Formalizing the proof of Theorem 3

6.1 Building blocks

The first ingredient is the BA called genericBA and further formalized in Appendix C.1. Let us briefly recall from Section 1.2.1 the following. genericBA is obtained by simplifying [1, §5.2] by two non-essential aspects, and by generalizing it to any ideal functionality-interface granting eligibility to speak in a given round. We called $\mathcal{F}_{\text{eligib}}$ such interface, and specified it in Figure 2. In *every* round r of genericBA, *every* player P is instructed to *conditionally multicast* a specified round-r message, denoted m. To this end, it queries $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(r)$. If returned 1, then we say that it is *eligible to speak in round* r. If so, then it multicasts the message m along with its signature on m, then rotates its signing key to the new one of the next round $r+1$, then erases its old round-r signing key. On receiving a round-r signed message from some player P, a player Q processes it if and only if $\mathcal{F}_{\text{eligib}}.\mathsf{verify}(P, r)$ returns 1.

Only for convenience of the phrasing of Properties 16, we introduce the following terminology. The protocol runs in iterations $v = 1, 2, \ldots$. The first iteration consists of the two rounds $r = 1, 2$, which for convenience we dub vote and commit. Higher iterations $v \geqslant 2$ consist of four rounds $r = 2 + 4(v-1) + j$, $j \in \{1, 2, 3, 4\}$, which we dub respectively status,propose,vote, and commit. We defer the specification of genericBA to Appendix C.1, because they are not needed if one believes in the following claim: our main message is that all its guarantees, which we single-out in Properties 16, solely follow from the outputs of the $\mathcal{F}_{\text{eligib}}$ interface. Since every player queries $\mathcal{F}_{\text{eligib}}(r)$ in every round r, independently of the execution, this shows that the guarantees below depend solely on how $\mathcal{F}_{\text{eligib}}$ will be instantiated. The conditions below involve a parameter λ, and a threshold number of corruptions equal to $f \leqslant n(1/2 - \epsilon)$, where ϵ is a parameter.

Properties 16. *Consider the protocol* genericBA *described in Appendix C.1 (a simplification of [1, §5.2]). Denote* V^{luck} *the smaller iteration number (possibly* ∞*) for which both conditions hold:*

committees. *in both the* vote *and* commit *rounds:* $\geqslant \lambda/2$ *honest players eligible to speak, and* $< \lambda/2$ *corrupt players are eligible to speak;*

leader. *if furthermore* $V^{\text{luck}} \geqslant 2$*, then: there exists* exactly one *player eligible to speak in the* propose *round, and this player is furthermore honest;*

then all players terminate by the first round of iteration $V^{\text{luck}} + 1$*.*
If furthermore for all iterations $v \leqslant V^{\text{luck}}$ *we have: (i) at least one honest player is eligible to speak in the* status *round, and (ii) in both the* vote *and* commit *rounds we have:* $\geqslant \lambda/2$ *honest players eligible to speak, and* $< \lambda/2$ *corrupt players eligible to speak; then, the execution satisfies Consistency and Strong unanimity.*

The proof is obtained by compiling the one given in [2, §5.3], which is simplified by our two simplifications. The compilation essentially consists in replacing every occurence of "By Lemma 1 / By Chernoff, except for $\exp(-\Omega(\lambda))$ probability" by: "by definition of V^{luck} " or "by condition (i)/(ii)", depending on the case. We provide more details in Appendix C.2, and also explain in Appendix C.3 how to adapt the statement and the proof to when the termination mechanism is reincorporated.

From Properties 16, it follows that the only ingredient needed to obtain Theorem 3 is an instantiation of $\mathcal{F}_{\text{eligib}}$ in the bulletin-board PKI model, such that: V^{luck} is independent of n, and such that conditions (i) and (ii) hold, for a given ϵ, up to exponentially small probability in λ the number of players which speak. All the ideas to implement $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ in the bulletin-board PKI model were conveyed in Section 1.2.1, we now formalize them as protocol $\Pi^{\text{bias}}_{\text{eligib}}$, which we give in Figure 5. To describe it, we borrow the ideal functionality called $\mathcal{F}_{\text{VRF}}$ of a verifiable random function (VRF) from [27, Fig. 2], which we recall in Figure 10 of Appendix C. This $\mathcal{F}_{\text{VRF}}$ model further simplifies the syntax of a VRF which we used in Section 1.2.1. Indeed, instead of manipulating secret keys, $\mathcal{F}_{\text{VRF}}$ directly ignores requests for provable evaluations if they do not come from the same entity which generated the public key *vk*.

The proof that $\Pi^{\text{bias}}_{\text{eligib}}$ implements $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ follows from a straightforward simulation, which is furthermore perfect. The only subtlety is when the simulator $\mathcal{S}$ receives an $\mathcal{F}_{\text{VRF}}$-evaluation request from the adversary, for a key *vk* which was not yet declared to $\mathcal{F}_{\text{VRF}}$ as assigned to a corrupt player. Then $\mathcal{S}$ predicts for which corrupt player *vk* will be used in conjunction with the given seed σ. To do this, it looks at which position the key *vk* appears in the seed σ (or in the preimage of the seed, in case H would not be the identity function but a simulated oracle). We defer the details to Appendix C.5.

6.2 Deriving Theorem 3

In conclusion, it remains to prove the claimed asymptotic complexities of genericBA instantiated with $\mathcal{F}^{\text{bias}}_{\text{eligib}}$. Given Properties 16, we obviously specify $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ with the same probabilities to be eligible in a round as those in [2, §5.2], i.e.: $\mathfrak{p}(\textsf{propose}) = 1/n$ and $\mathfrak{p}(\textsf{status/vote/commit}) = \lambda/n$. For simplicity we use a complexity model where the adversary $\mathcal{A}$ can query $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ on at most q distinct seeds. Then for each seed, we consider that $\mathcal{A}$ can make an unlimited number of queries for all eligibilities of all corrupt players in all rounds. We refer to [26, Lemma 1] for

$\Pi^{\text{bias}}_{\text{eligib}}(\mathfrak{p})$

Setup. Before time $t = 0$: each player queries $\mathcal{F}_{\text{VRF}}$.keyGen, then upon receiving a key $\boldsymbol{vk}$, publishes it on the bulletin-board.
At time $t = 0$: players retrieve the keys published on the bulletin-board. Denote their list as: $\sigma \leftarrow (\boldsymbol{vk}_1, \ldots, \boldsymbol{vk}_n)$ (unpublished keys are set to $\bot$).

Request to speak in round r. Player P queries $\mathcal{F}_{\text{VRF}}$.evalProve$((\boldsymbol{\sigma}, r))$. Upon being returned the provable evaluation (y, π): if $y < \mathfrak{p}(r)$, then output 1, i.e., eligible in r, then multicast (r, y, π). Else, output 0.

Verify(P, r). If no (r, y, π) was received from P, output 0. Else, let vk by the public verification key of P (retrieved from the bulletin-board). Query $\mathcal{F}_{\text{VRF}}.\mathit{Verify}((\boldsymbol{\sigma}, r), y, \pi, vk)$, and output the response received from $\mathcal{F}_{\text{VRF}}$.

Figure 5: Protocol implementing $\mathcal{F}^{\text{bias}}_{\text{eligib}}$. The two differences with the implicit implementation of $\mathcal{F}_{\text{mine}}$ in [2, §9.4] are: the setup (there is no longer a trusted party generating the keys) and for the publication on the bulletin board), and our pre-pending of the seed σ to all VRF evaluations. To highlight these differences, the rest is shaded-out. We further simplified Section 1.2.1 by setting H equal to the identity function. For convenience we normalize to $[0, 1]$ the set of evaluations of $\mathcal{F}_{\text{VRF}}$. As in [27, Fig. 2], recalled in Figure 10, the verification key lengths are not specified. Recall that in the $\mathcal{F}_{\text{VRF}}$ model, all public keys returned by keyGen are chosen by the adversary, at the only condition that the keys of players are all distinct and distinct from the ones registered directly by $\mathcal{A}$ in its name.

a thinner model and analysis (their functionality $\mathcal{F}_{\text{bias}}$, which they implement using a costlier setup, looks similar to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$).

We first prove that all criterions in Properties 16 of the type $\geqslant \lambda/2$ and $< \lambda/2$, are matched with overwhelming probability. Then in the next paragraph we will address the remaining [leader.] criterion. Consider a fixed seed σ which was not queried before to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$. Then, by definition of $\mathcal{F}^{\text{bias}}_{\text{eligib}}$, all eligibilities of all players in all rounds are sampled independently from all previous queries with other seeds, and also sampled independently from each other. In particular, consider a fixed status/vote/commit round r. Then by the Chernoff bounds, applied as detailed in Appendix B.5, the probability of the bad event $\text{bad}_{r,\sigma}$ that $\geqslant \lambda/2$ corrupt players are sampled eligible in round r is $\leqslant \exp(-\epsilon^2 \Omega(\lambda))$. From now on we drop the dependency in ϵ for simplicity, i.e., we consider a fixed ϵ. By independence of the eligibilities in distinct rounds, for a given number of rounds R, we obtain that the bad event $\text{bad}_{r,\sigma}$ does not happen in any $r \in [1, \ldots, R]$ with probability at least $(1 - \exp(-\Omega(\lambda)))^R$. By independence of the eligibilities sampled over distinct seeds, we obtain that the bad event bad_r does not happens in any round for any of the q seeds tried by $\mathcal{A}$, with probability at least $(1 - \exp(-\Omega(\lambda)))^{qR}$. Notice that, by contrast, the probabilities of the other bad event: existence of a round $r \in [1, \ldots, R]$ such that $< \lambda/2$ honest players are eligible in r, stays equal to $(1 - \exp(-\Omega(\lambda)))^R$ whatever the number q of re-seedings in the setup. Indeed, the eligibility of each honest player in a given round r is sampled only when it (privately) queries it to $\mathcal{F}_{\text{eligib}}^{\text{bias}}$, and by definition is sampled independently from the outcome of all previous queries of the adversary.

We now turn to upper-bound the probability that a given iteration v satisfies the [leader.] criterion. Consider one fixed seed σ. Then the probability that, in one given iteration v, there is no corrupt player eligible to propose, is $\geqslant ((n-1)/n)^{n/2} \cong e^{-1/2}$. Thus the probability that at least one corrupt player is eligible to propose, is $\leqslant 1 - e^{-1/2}$. By independence of eligibilities in distinct iterations, the probability $p_{\text{bad}}(\sigma)$ that in each of the V first iterations there is at least a corrupt player eligible to propose, is thus $\leqslant (1 - e^{-1/2})^V$. Taking the union bound over the q different seeds tried by the adversary, it follows that the probability p_{bad} that in each of the V first iterations there is at least a corrupt player eligible to propose, is $\leqslant q(1 - e^{-1/2})^V$. In conclusion, the probability that the [leader.] condition fails in all V first iterations is exponentially decreasing in V, which shows our constant round complexity claim. Note that in the previous conclusion we neglected the event where one round may fail to match the [committee.] condition, since this event has negligible probability in λ by the previous paragraph. Note that in the previous conclusion we overlooked the other requirement of [leader.] that exactly one honest player is eligible. So we would have had to multiply the previous upper-bound by the probability that *exactly* one honest player is eligible to propose in a given iteration. Neglecting ϵ, the latter is roughly equal to $n/2.\frac{1}{n}.\left(\frac{n-1}{n}\right)^{n/2-1} \cong 1/2.e^{-1/2}$. But this latter detail is irrelevant with respect to the substantial optimization described in the beginning of Appendix C.6 (imported from Algorand).

References

[1] I. Abraham, T.-H. H. Chan, D. Dolev, K. Nayak, R. Pass, L. Ren, and E. Shi. "Communication Complexity of Byzantine Agreement, Revisited". In: *PODC*. 2019.

[2] I. Abraham, T. H. Chan, D. Dolev, K. Nayak, R. Pass, L. Ren, and E. Shi. "Communication complexity of byzantine agreement, revisited". In: *Distributed Comput.* (2023).

[3] D. Abram, M. Obremski, and P. Scholl. *On the (Im)possibility of Distributed Samplers: Lower Bounds and Party-Dynamic Constructions*. ePrint 2023/863. 2023.

[4] D. Abram, P. Scholl, and S. Yakoubov. "Distributed (Correlation) Samplers: How to Remove a Trusted Dealer in One Round". In: *EUROCRYPT*. 2022.

[5] A. B. Alexandru, J. Loss, C. Papamanthou, and G. Tsimos. *Sublinear-Round Broadcast without Trusted Setup against Dishonest Majority*. Iacr ePrint 2022/1383. 2022.

[6] O. Alpos, C. Cachin, S. H. Kamp, and J. B. Nielsen. "Practical Large-Scale Proof-of-Stake Asynchronous Total-Order Broadcast". In: *AFT*. 2023.

[7] M. Andrychowicz and S. Dziembowski. "PoW-Based Distributed Cryptography with No Trusted Setup". In: *CRYPTO*. 2015.

[8] F. Benhamouda, E. Blum, J. Katz, D. Leung, J. Loss, and T. Rabin. "Analyzing the Real-World Security of the Algorand Blockchain". In: *CCS*. 2023.

[9] E. Blum, E. Boyle, R. Cohen, and C. Liu-Zhang. "Communication Lower Bounds for Cryptographic Broadcast Protocols". In: *DISC*. 2023.

[10] E. Blum, J. Katz, C. Liu-Zhang, and J. Loss. "Asynchronous Byzantine Agreement with Subquadratic Communication". In: *TCC*. 2020.

[11] M. Borcherding. "Levels of authentication in distributed agreement". In: *WDAG*. 1996.

[12] E. Boyle, R. Cohen, and A. Goel. "Breaking the $O(\sqrt{n})$-Bit Barrier: Byzantine Agreement with Polylog Bits Per Party". In: *PODC*. 2021.

[13] E. Boyle, R. Cohen, and A. Goel. "Breaking the $O(\sqrt{n})$-Bit Barrier: Byzantine Agreement with Polylog Bits Per Party". In: *J. Cryptol.* (2024).

[14] C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl. "Asynchronous Verifiable Secret Sharing and Proactive Cryptosystems". In: *CCS*. 2002.

[CKPS01] C. Cachin, K. Kursawe, F. Petzold, and V. Shoup. "Secure and Efficient Asynchronous Broadcast Protocols". In: *CRYPTO*. 2001.

[15] J. Camenisch, R. R. Enderlein, S. Krenn, R. Küsters, and D. Rausch. "Universal Composition with Responsive Environments". In: *ASIACRYPT*. 2016.

[16] R. Canetti. "Universally Composable Security: A New Paradigm for Cryptographic Protocols". In: *FOCS'01 and J. ACM 2020*. retrieved from Iacr ePrint 2000/067, version 2020. 2020.

[17] R. Canetti. "Universally Composable Signature, Certification, and Authentication". In: *CSFW*. long and corrected version of August 15. 2004.

[CGGM00] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali. "Resettable Zero-Knowledge (Extended Abstract)". In: *STOC*. 2000.

[18] T.-H. H. Chan, R. Pass, and E. Shi. "Sublinear-Round Byzantine Agreement Under Corrupt Majority". In: *PKC*. 2020.

[19] J. Chen and S. Micali. "Algorand: A secure and efficient distributed ledger". In: *Theor. Comput. Sci.* (2019).

[20] P. Civit, S. Gilbert, R. Guerraoui, J. Komatovic, A. Paramonov, and M. Vidigueira. *All Byzantine Agreement Problems are Expensive*. 2023.

[21] P. Civit, S. Gilbert, R. Guerraoui, J. Komatovic, and M. Vidigueira. "On the Validity of Consensus". In: *PODC*. 2023.

[22] S. Cohen, I. Keidar, and O. Naor. “Byzantine Agreement with Less Communication: Recent Advances”. In: *SIGACT News* (2021).

[23] S. Coretti, J. Garay, M. Hirt, and V. Zikas. “Constant-Round Asynchronous Multi-Party Computation Based on One-Way Functions”. In: *ASIACRYPT*. 2016.

[24] S. Coretti, A. Kiayias, C. Moore, and A. Russell. “The Generals’ Scuttlebutt: Byzantine-Resilient Gossip Protocols”. In: *CCS*. 2022.

[25] R. J. Cramer, I. Damgard, and J. B. Nielsen. *Secure multiparty computation and Secret Sharing*. Cambridge University Press, 2015.

[26] P. Daian, R. Pass, and E. Shi. “Snow White: Robustly Reconfigurable Consensus and Applications to Provably Secure Proof of Stake”. In: *FC*. 2019.

[27] B. David, P. Gazi, A. Kiayias, and A. Russell. “Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain”. In: *EUROCRYPT*. 2018.

[28] D. Dolev and R. Reischuk. “Bounds on Information Exchange for Byzantine Agreement”. In: *J. ACM* (1985).

[29] D. Dolev and H. Strong. “Authenticated Algorithms for Byzantine Agreement”. In: *SIAM J. Comput.* (1983).

[30] C. Dwork, N. Lynch, and L. Stockmeyer. “Consensus in the Presence of Partial Synchrony”. In: *J. ACM* (1988).

[31] L. Eckey, S. Faust, and J. Loss. “Efficient Algorithms for Broadcast and Consensus Based on Proofs of Work”. In: *eprint 2017/915* (2017).

[32] M. F. Esgin, O. Ersoy, V. Kuchta, J. Loss, A. Sakzad, R. Steinfeld, X. Yang, and R. K. Zhao. “A New Look at Blockchain Leader Election: Simple, Efficient, Sustainable and Post-Quantum”. In: *ASIA CCS*. 2023.

[33] M. Fitzi. “Generalized communication and security models in Byzantine agreement”. PhD thesis. ETH Zurich, Zürich, Switzerland, 2003.

[34] Y. Gao, Y. Lu, Z. Lu, Q. Tang, J. Xu, and Z. Zhang. “Dumbo-NG: Fast Asynchronous BFT Consensus with Throughput-Oblivious Latency”. In: *CCS*. 2022.

[35] J. A. Garay, A. Kiayias, and N. Leonardos. “The Bitcoin Backbone Protocol: Analysis and Applications”. In: *EUROCRYPT*. 2015.

[GKLP18] J. Garay, A. Kiayias, N. Leonardos, and G. Panagiotakos. “Bootstrapping the Blockchain, with Applications to Consensus and Fast PKI Setup”. In: *PKC*. 2018.

[GKOPZ20] J. Garay, A. Kiayias, R. M. Ostrovsky, G. Panagiotakos, and V. Zikas. “Resource-Restricted Cryptography: Revisiting MPC Bounds in the Proof-of-Work Era”. In: *EUROCRYPT*. 2020.

[36] J. Garay, A. Kiayias, and Y. Shen. “Proof-of-Work-based Consensus in Expected-Constant Time”. In: *EUROCRYPT*. 2024.

[37] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. "Algorand: Scaling Byzantine Agreements for Cryptocurrencies". In: *SOSP*. 2017.

[38] V. Goyal, H. Li, and J. Raizes. "Instant Block Confirmation in the Sleepy Model". In: *FC*. 2021.

[39] J. Groth and R. Ostrovsky. "Cryptography in the multi-string model". In: *J. Cryptol.* (2014).

[40] T.-H. Hubert Chan, R. Pass, and E. Shi. "Consensus Through Herding". In: *EUROCRYPT*. 2019.

[41] J. Katz, U. Maurer, B. Tackmann, and V. Zikas. "Universally Composable Synchronous Computation". In: *TCC*. 2013.

[42] L. Lamport, R. Shostak, and M. Pease. "The Byzantine Generals Problem". In: *ACM Trans. Program. Lang. Syst.* (1982).

[43] Y. Lindell, A. Lysyanskaya, and T. Rabin. "On the Composition of Authenticated Byzantine Agreement". In: *STOC*. 2002.

[44] C. Liu-Zhang, J. Loss, U. Maurer, T. Moran, and D. Tschudi. "MPC with Synchronous Security and Asynchronous Responsiveness". In: *ASIACRYPT*. 2020.

[45] C. Liu-Zhang, C. Matt, U. Maurer, G. Rito, and S. E. Thomsen. "Practical Provably Secure Flooding for Blockchains". In: *ASIACRYPT*. 2022.

[46] C. Liu-Zhang, C. Matt, and S. E. Thomsen. "Asymptotically Optimal Message Dissemination with Applications to Blockchains". In: (2022).

[47] Y. Lu, Z. Lu, Q. Tang, and G. Wang. "Dumbo-MVBA: Optimal Multi-Valued Validated Asynchronous Byzantine Agreement, Revisited". In: *PODC*. 2020.

[48] D. Malkhi, A. Momose, and L. Ren. "Towards Practical Sleepy BFT". In: *CCS*. 2022.

[49] A. Momose and L. Ren. "Constant Latency in Sleepy Consensus". In: *CCS*. 2023.

[50] A. Momose and L. Ren. "Optimal Communication Complexity of Authenticated Byzantine Agreement". In: *DISC*. 2021.

[51] S. Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: *Cryptography Mailing list at https://metzdowd.com* (Mar. 2009).

[52] R. Pass and E. Shi. "The Sleepy Model of Consensus". In: *ASIACRYPT*. 2017.

[53] R. Pass and E. Shi. "Thunderella: Blockchains with Optimistic Instant Confirmation". In: *EUROCRYPT*. 2018.

[54] N. Shrestha, A. Bhat, A. Kate, and K. Nayak. "Synchronous Distributed Key Generation without Broadcasts". In: *IACR ePrint* (2021).

[55] J. Wan, H. Xiao, S. Devadas, and E. Shi. "Round-Efficient Byzantine Broadcast under Strongly Adaptive and Majority Corruptions". In: *TCC*. 2020.

[56] B. G. and Yuan Lu and Zhenliang Lu and Qiang Tang and Jing Xu and Zhenfeng Zhang. "Speeding Dumbo: Pushing Asynchronous BFT Closer to Practice". In: *NDSS*. 2022.

[57] T. Yurek, Z. Xiang, Y. Xia, and A. Miller. "Long Live The Honey Badger: Robust Asynchronous DPSS and its Applications". In: *Usenix security*. 2023.

[58] M. Zhang, R. Li, and S. Duan. *Max Attestation Matters: Making Honest Parties Lose Their Incentives in Ethereum PoS*. ePrint 2023/1622. 2023.

A Related works

A.0.1 The *post-publishing-of-keys-unpredictable-seed setup*, and the one-by-one adversarial key picking attack. Some consensus algorithms [37, 19, 27] [53, §4.2.1] assume a setup which fairly samples the seed σ of the VRF used for self-sortition, then publicly reveals it *after* all participants published their keys. In those works, the seed σ (called "nonce" in [26, 27]) appears in a so-called "genesis block". Note that in [53, §4.2.1], the VRF is implemented from a fresh random oracle: H which appears after publication of keys. But as noticed in [52], assuming a fresh random oracle trivially falls back to the fresh seed model: simply use an old random oracle (in their case: a PRF) with inputs pre-pended by the fresh seed. Let us recall why this model cannot be simply downgraded into a seed which would be known the the adversary $\mathcal{A}$ *before* corrupt players publish their VRF keys. Consider the scenario where this would happen. Let us consider simultaneously the examples of: Thunderella, where an output can be reached in 2 rounds by [53, Thm 10] (propose, then 3/4 majority of votes); Algorand [37], where it is reached in in 4 rounds (page 5, "Efficiency"); and [1, 2] (§5.1) which take 3 rounds. So in order to break consistency of all those consensus protocols, it is enough for the adversary to ensure that a corrupt player is eligible as leader in the first round (in order to make equivocating proposals), and that $\geqslant \lambda/2$ corrupt players are eligible as voters of the subsequent rounds 2, 3 and 4. The adversary $\mathcal{A}$ can easily achieve this objective in our σ-known-to-$\mathcal{A}$ scenario, by choosing the VRF keys of corrupt players as follows. Denote λ the expected number of eligible voters per round. For one fixed corrupt P, try on average n key pairs (sk,vk) until the VRF.$\mathsf{eval}(sk, \sigma|1)$ returns a value eligible to be the first leader. Then for every other corrupt player Q, one-by-one until $\lambda/2$ of them, try on average $(n/\lambda)^3$ keys pairs (sk,vk) until the VRF.$\mathsf{eval}(sk, \sigma|i)$ returns an eligible value for all $i = 2, 3, 4$.

On the face of it, the VRF in the works [38, 49, 48] does not take any public seed, and furthermore players are allowed to pick their VRF secret keys. Note that they call "seeds" these secret keys. So a priori this allows the above adversarial key picking attack. From [38]: "Due to the complexity of instantiating VRFs when players may choose their own seeds, we model them as random oracles, and direct readers to [Algorand] for a more in-depth treatment of the subject.", our best guess is that their model is that a fresh ideal VRF appears after players published their keys. Again, by domain-separation, this fresh ideal VRF could be implemented as an old ideal VRF, of which the inputs would be pre-pended with a fresh random seed implicitely assumed by the model.

A.0.2 How our mechanism of Theorem 3, with a weaker setup, defeats this attack.

For each new attempt of a new key pair (sk_i, vk_i) for some corrupt P_i, since this modifies vk_i, this modifies the potential seed $\sigma = H\big((vk_1, \ldots, vk_n)\big)$, so completely re-samples afresh the eligibilities of all other corrupt players.

A.0.3 The Honestly-sampled-keys setup.

Some feasibility results on consensus [43, 2, 40, 18, 10, 9] assume a setup which is strictly stronger than the bulletin-board PKI. There, the VRF secret keys of corrupt players are honestly sampled (either by restricting the adversary, or, by assuming a trusted third-party). So this model makes impossible, by definition, the above attack where the adversary repeatedly samples the VRF secret key of each corrupt player P, until it chooses one which grants eligibility of P in sufficiently many committees.

A.0.4 Static adversaries.

In both [26, 6], there is a public function which returns if a given player is eligible to speak in a given round, e.g., in [6] for dealing a coin. But the eligibility function takes only public inputs. Thus, the adversary knows in advance all eligible honest players (and also the committees of share-holders, in [6]). The model of [26] explicitely handles this limitation by fixing the corruption delay equal at least as large as the delay to reach consensus, on a so-called checkpoint. For our concern of a single consensus instance (not a blockchain, as them), this is equivalent to a static corruptions. Turning to [52], in §7.4 the proof for adaptive security does a reduction to static security with 2^n loss. So this is incompatible with our main concern, which is the regime of asymptotic complexity in n. As regards complexity, the adaptively-secure mechanism of [52, §6.1] is openly stated to be prone to a one-by-one adversarial key picking attack. They give the example of a lazy adversary which picks the key of each corrupt player P so as to grant P one leader slot (as noticed in **??** A.0.1, offering v leader slots to P would take only n^v trials of keys in expectation). They deduce that the round complexity is linear in f, so this is incompatible with our concern, which is constant round complexity.

The synchronous BA of [12, 13] is in the bulletin-board PKI model, however their adversary cannot adaptively corrupt players after the setup. It has $f < (1/3 - \epsilon)n$ corruption resilience and $O(n\text{polylog}(n)$ communication complexity. With this respect, our upper-bound Theorem 3 has strictly better parameters: $f < (1/2 - \epsilon)n$ corruption resilience, $O(n)$ complexity and tolerance to a rushing adaptive adversary. Their BA is *balanced*, in that each player sends messages to no more than polylog(n) peers. Since our protocol proceeds by simultaneous multicasts, those can also be implemented by protocols in which players speak only to a few peers [24, 45, 46].

A.0.5 The lower bounds Theorems 4 and 5 are in stronger models than previously.

To our knowledge, all previous lower bounds for consensus with an adaptive adversary are in strictly weaker models. Those of [10, §7], [12, 13, Thm 1.5] and [1, Thm 3] *do not assume message-authentication*. Whereas, the one of [1, Thm 1] assumes that the adversary can remove messages which were already sent in the round. We now compare more particularly to the one of [1, Thm 3], since the technique is similar. It is stated for *broadcast*, instead of for BA. As explained at the end of Section 1.3.2, their proof technique *does not* provide a lower bound for broadcast in the message-authentication model. Hence, our contribution consisted in observing that their proof technique, when upgrading to the message-authentication model, can be successfully adapted provided switching the problem to *BA*.

A.0.6 Are distributed samplers of any help?

In [4] (followed by [3]) they implement a functionality (Figure 5) which, upon being queried by the adversary, (re-)samples from any prescribed distribution; then publishes the sample which the adversary liked the most. So it would not enable to sample a seed in a more fair way than [37, 26, 27, 19] (see below), nor than what we do. It is even orthogonal to our needs, since if we had used instead their functionality instead to set the public seed of the VRF, then this would have allowed the same attack as above. Namely: the adversary would, all in the same round of publication on the bulletin-board: observe the seed σ which will be output by the functionality (it needs not even bias it), then choose the VRF secret key of each corrupt player P in order to maximize its eligibility, and publish it.

A.0.7 Setups with *interactions* (and at least quadratic complexity).

If we had allowed two consecutive rounds of publication on the bulletin-board followed by all-to-all messages, then this would have enabled to implement the *unbiased* idealized self-sortition functionality $\mathcal{F}_{\text{mine}}$ of [1, 40, 18] as follows: 1. players publish their VRF public keys, as well as public encryption keys, 2. each player publishes a PVSS of a random value, 3. players open the published PVSSs, and define the VRF seed σ equal to the sum of the opened values. The setups of [10, 6] are also interactive, since they proceed by such *several consecutive* broadcasts or BA instances. Since before Theorem 3, no consensus with linear complexity was known in the bulletin-board PKI model, a fortiori no algorithm with linear complexity was either known to implement these setups. Likewise, [37, 27, 19] consider an ever-growing chain of consensus (VBA) instances. The VRF seed of later instances is determined by the output of older instances. However this does not settle how the VRF seed of the *first* consensus instance is set, which is what our work is addressing. Hence it is not apples-to-apples to compare our single-shot instance with the performances of their later instances.

A.0.8 Ressource-restricted cryptography, and a failed attempt.

This model [GKOPZ20], initiated by the Bitcoin protocol [51], assumes that honest players are able to collectively spend more ressources than the adversary. It is shown in [51, 35] how to implement BA under honest majority in this model, using only plain synchronous authenticated channels, thereby circumventing the $f \geqslant n/3$ impossibility of [42, 11]. Let us recall the technique of [7, GKLP18] which removes the need for an unpredictable seed in the genesis block of Bitcoin. To send a message in a given round r, a player must solve a challenge derived from, roughly, a quorum of round-(r-1) messages. Then in its round-r message, the player includes some randomness: as a result, as long as a quorum of round-r messages contain at least one issued by an honest player, the challenge will be fresh. In our setting, it was tempting to adapt this technique by using a quorum of round-(r-1) randomnesses as a seed in the VRF evaluation. Unfortunately, VRF evaluations are much cheaper than PoW challenges. So the adversary could, for each corrupt player P *one by one*, try different quorums of messages until one yields a VRF seed granting eligibility to P. Hence, this falls back to the attack of **??** A.0.2. A related model is time-based cryptography, in which honest players are assumed to compute a function as fast as the adversary [55] (time-lock puzzles) or [5] (VDFs).

B Additional details on modeling

Recall that the UC model [16] considers a PPT machine called the *environment* $\mathcal{Z}$, which controls the adversary and assigns their inputs to players. Moreover, when defining external validity (Definition 9), it should be formalized that $\mathcal{Z}$ defines the predicate *ext-valid*. An example is that *ext-valid* checks validity of signatures of an entity controled by $\mathcal{Z}$. Another (generic) example is to formalize *ext-valid* as an oracle controled by $\mathcal{Z}$, to the extend that it always return the same output when queried twice on the same input.

Moreover, $\mathcal{Z}$ controls the pace at which each player does its actions, in particular, can completely stall the execution. This latter limitation does not impact our results since all our specifications, e.g., Definition 1, apply only to infinite executions.

Notice that we do *not* require the UC implementation of BA as an ideal functionality. Hence, we do not specify that the environment observes the outputs of players and tries to distinguish the protocol from a dummy interaction with an ideal functionality of BA Instead, we stay at the level of our property-based Definition 1 (which thus makes our impossibilities stronger). Requiring so is orthogonal to our contributions.

B.1 Public authenticated channels $\mathcal{F}_{\mathrm{AUTH}}$, and synchrony

We recall below the functionality of public authenticated message transmitting of [16] and [25, §4.2.3].

The functionality of *secure* message transmitting, formalized as $\mathcal{F}_{\mathrm{SMT}}$ in [16] and [25, §4.4.2], is the upgrade where the content of the message is kept secret. Concretely, it leaks only $(\mathsf{sent}, \mathit{ssid}, |m[)$ to $\mathcal{A}$, where $|m|$ is the bitlength of m, instead of $(\mathsf{sent}, \mathit{ssid}, m)$.

$\mathcal{F}_{\mathrm{AUTH}}^{S \to R}$

1. On input $(\mathsf{send}, ssid, m)$ from S, leak $(\mathsf{sent}, ssid, m)$ to $\mathcal{A}$ and store $(ssid, m)$. Ignore any later input of the form $(\mathsf{send}, sid, \_)$.
2. Upon receiving $(\mathsf{ok}, ssid)$ from $\mathcal{A}$, if some $(ssid, m)$ is stored, then output $(ssid, m)$ to R and delete $(ssid, m)$.

Figure 6: Public authenticated message transmitting. It is parametrized by a sender S and a receiver R

Formalizing synchrony. Since the previous formalism does not capture synchrony, since the adversary can block the output forever, we now describe the fix proposed in [41, §3.3]. There, $\mathcal{F}_{\mathrm{AUTH}}$ (in their case: $\mathcal{F}_{\mathrm{SMT}}$) is upgraded as follows. $\mathcal{F}_{\mathrm{AUTH}}$ is parametrized by a public integer Δ. When the S inputs a message $(\mathsf{send}, ssid, m)$, $\mathcal{F}_{\mathrm{AUTH}}$ initializes a counter $D \leftarrow 1$ which models the delivery delay for the message id $ssid$. The adversary can make requests to $\mathcal{F}_{\mathrm{AUTH}}$ increase D by $+1$, up to a total number of $\Delta - 1$ requests. On the other hand, for every message id $ssid$ which the receiver R expects from the sender S, R can make fetch requests to $\mathcal{F}_{\mathrm{AUTH}}$ which have for effect to decrease D by -1. When D reaches 0, $\mathcal{F}_{\mathrm{AUTH}}$ delivers the message to R. Note that, in Section 1 and in the paper in general, we set the unit of time equal to Δ. Hence, the notation time $t = 1$ actually means $t = \Delta$.

The other ingredient needed to emulate synchrony is the global clock, which [41] emulate as follows. They introduce a clock functionality accessible by all players, which roughly does the following. When a player has fetched Δ times all the messages of a round r that it expected to receive, and done all the processing of messages that it needed to do, it notifies the clock that it is ready. Upon being notified by all honest players that they are ready, the clock ticks $r + 1$, i.e., allows them to proceed to sending their round $r + 1$ messages.

B.2 Partial synchrony

The bounds in Section 2 are stated in the model of partial synchrony defined as "Δ holds eventually" in [30, §2.3 3)]. As explained in the beginning of Section 2, a partially synchronous protocol can offer meaningful guarantees only if we further restrict GST to be polynomial. Let us propose a UC formalism of this restriction, which parallels the one of [41, 23, 44] for asynchronous eventual delivery of messages.

It is conveniently described by merging all $\mathcal{F}_{\mathrm{AUTH}}^{S \to R}$ into one single $\mathcal{F}_{\mathrm{AUTH}}$ which accepts all senders and receivers. We enrich $\mathcal{F}_{\mathrm{AUTH}}$ with a counter D', initialized to 0. The adversary can set D' equal to a value, denoted GST, which it must input *in unary notation* before the protocol begins. Since the adversary is polynomial, it follows that GST is polynomial. At the end of every round, $\mathcal{F}_{\mathrm{AUTH}}$ sets $D' \leftarrow D' - 1$ by one. As long as D' does not reach 0, $\mathcal{F}_{\mathrm{AUTH}}$

operates as asynchronous message transmitting with eventual delivery as in [41, 23, 44]. When D' reaches 0 for the first time, $\mathcal{F}_{\mathrm{AUTH}}$ switches forever to the mechanism of Figure 6. Moreover at this point, if some messages not delivered yet has a current delay $D > \Delta$, then their delay D is set to $D = \Delta$.

B.3 Formalizing the bulletin-board PKI setup

Let us slightly more formalize the bulletin-board PKI setup, which we specified following [CGGM00]. We defined it as a setup protocol: Π_s, played before the time $t = 0$ at which players receive their inputs. Moreover, Π_s has the following form. Before $t = 0$, each player P has writing access to a public bulletin board. The Π_s instructs P to generate a string then write it on the bulletin board. The Π_s is non-interactive, i.e., the string does not depend on the other strings which have possibly been written by other players. On the other hand, the adversary learns instantaneously the strings written by honest players. Thus it can adaptively choose the strings on behalf of corrupt players, and is allowed to write them after all honest players wrote. Then from $t = 0$, all players have read-only access to the bulletin board. The closest formalization of such a bulletin-board which we found in the litterature is the ideal functionality $\mathcal{F}_{\mathrm{CA}}$ introduced in [17], and which we reproduce in Figure 7.

$\mathcal{F}_{\mathrm{CA}}$

1. Upon receiving the first message $(\mathsf{register}, v)$ from a player P, send $(\mathsf{registered}, P, v)$ to $\mathcal{A}$; upon receiving ok from $\mathcal{A}$, and if this is the first request from P, then record the pair (P, v).
2. Upon receiving a message $(\mathsf{retrieve}, P)$ from player Q, send $(\mathsf{retrieve}, P, Q)$ to $\mathcal{A}$, and wait for an ok from it. Then, if there is a recorded pair (P, v), output $(\mathsf{retrieve}, P, v)$ to Q. Else output $(\mathsf{retrieve}, P, \perp)$ to Q.

Figure 7: The certification authority functionality: $\mathcal{F}_{\mathrm{CA}}$.

We found that $\mathcal{F}_{\mathrm{CA}}$ is equivalent to a broadcast channel. Hence, we give below the ideal functionality of broadcast, adapted from [39]. It is parametrized by a sender S and a set $\mathcal{R}$ of receivers. In our context of bulletin-board PKI: there are n instances, in instance i the i-th player acts as the sender, and the set of receivers is equal to all players $\mathcal{P}$.

Formalizing delivery before $t = 0$ Whatever the formalism, $\mathcal{F}_{\mathrm{CA}}$ or $\mathcal{F}_{\mathrm{BC}}$, the formalism above does not yet capture the timing assumption that all players are delivered an output by time $t = 0$. We now propose a mechanism to emulate this timing assumption in UC model, following the mechanism of [41] which is recalled above for point-to-point message transmitting. Namely: $\mathcal{F}_{\mathrm{CA}}$ or $\mathcal{F}_{\mathrm{BC}}$ initialize a counter $1 \leftarrow D_{S \leftarrow R}$ for each pair of sender (S, receiver R). The

$\mathcal{F}_{\mathrm{BC}}^{S,\mathcal{R}}$

Upon receiving for the first time: $\{(v \in \{0,1\}^*)$ from S if S honest$\}$ OR $\{(v \in \{0,1\}^* \sqcup \{\perp\})$ from $\mathcal{A}$ if S corrupt$\}$. Then hand v to $\mathcal{A}$, and when $\mathcal{A}$ allows then send (v) to every $R \in \mathcal{R}$.

Figure 8: One broadcast instance, parametrized by a sender S and a set of receivers $\mathcal{R}$.

adversary can make up to $\Delta - 1$ requests to increase it by $+1$, while R can make repeated fetch requests to decrease it by -1. Upon the event where, for the first time, there is a receiver $R \in P$ which made Δ requests. Then, if $\mathcal{A}$ input nothing on behalf of the corrupt sender S, we specify that $\mathcal{F}_{\mathrm{BC}}$ sets the output to $\perp$. Then it delivers $\perp$ to R, as well as to every subsequent R which will reach a number Δ of fetch requests.

B.4 Ideal message-authentication functionality

We copy in Figure 9 the ideal message-authentication functionality of [17]. It is parametrized by a player S, denoted *signer*.

$\mathcal{F}_{\mathrm{CERT}}^{S}$

Signature Generation: Upon receiving (sign-request, sid, m) from S, check if $sid = (S, sid')$ for some sid'. If not, then ignore the request. Send (sign-request, sid, m) to $\mathcal{A}$. Upon receiving (signature, sid, m, σ) from $\mathcal{A}$, verify that no entry $(m, \sigma, 0)$ is recorded. If it is, then output an error message to S and halt. Else, output (signature, sid, m, σ) to S, and record the entry $(m, \sigma, 1)$.
Signature Verification: Upon receiving a value (verify, sid, m, σ) from some player P, hand (verify, sid, m, σ) to $\mathcal{A}$. Upon receiving (verification, sid, m, φ) from $\mathcal{A}$, do:

1. If $(m, \sigma, 1)$ is recorded then set $b = 1$.
2. Else, if the signer is not corrupted, and no entry $(m, \sigma', 1)$ for any σ' is recorded, then set $b = 0$ and record the entry $(m, \sigma, 0)$.
3. Else, if there is an entry (m, σ, b') recorded, then set $b = b'$.
4. Else, set $b = \varphi$, and record the entry (m, σ', φ).

Output (verification, sid, m, b) to P.

Figure 9: Ideal digital signature functionality, dubbed as “signing oracle”, for player S.

On the face of it, a corrupt P can possibly query (verify) on many triples (sid, m, σ) and then $\mathcal{A}$ forces $\mathcal{F}_{\mathrm{CERT}}$ to record $(m, \sigma, 0)$, preventing the subsequent use of these parameters by the signer S. But actually, as explained in [17, p 10], the (verify) requests are ignored if they do not come with a *sid* which was used by the signer S in the first place.

Notice also that $\mathcal{A}$ may block the delivery of the signature, by never answering (signature, sid, m, σ). Thus, the advancement of the BA protocol is stalled. This is a formal problem since, on the other hand, the Environment $\mathcal{Z}$ allows players to take an infinite number of steps, so the execution is still considered as infinite and thus the Termination requirement of Definition 1 should apply. This problem could be easily fixed by specifying $\mathcal{F}_{\text{CERT}}$ to issue a signature generated in a prescribed distribution, in case $\mathcal{A}$ would take too much time to respond. The mechanism in UC would follow the same fetch-and-delay mechanism as the one of [41] for synchronous $\mathcal{F}_{\text{AUTH}}$, recalled above. Notice that this fix is enough, thanks to the synchronous UC clock of [41] recalled above. Namely, whatever finite time it takes to $\mathcal{F}^S_{\text{CERT}}$ to deliver its output, the clock waits until S receives it and finishes its computations, before ticking the next round.

Notice that, without such a clock, so under asynchrony, nothing would prevent $\mathcal{F}^S_{\text{CERT}}$ from taking more delay than a number of messages delays. This artifact of the UC model is observed in [15], which point some failures in security proofs due to it. For this reason, they propose a UC mechanism which forces functionalities such as $\mathcal{F}_{\text{CERT}}$, i.e., modeling local computations, to deliver their output in priority before other functionalities.

B.5 Probabilistic inequalities

Proposition 17 (Markov bound). *Let X be a non-negative random variable. Then for any $a > 0$,*

$$\mathbb{P}[X \geqslant a] \leqslant \frac{\mathrm{E}[X]}{a} .$$

Proposition 18 (Chernoff bounds). *Consider $X_1, \dots, X_m$ Bernoulli variables, each of expected value p , i.e., $\mathbb{P}[X_i = 1] = p$ and $\mathbb{P}[X_i = 0] = 1 - p$ $\forall i \in [m]$. So $\mu := E\left[\sum_i X_i\right] = pm$. Then for every $0 \leq \delta < 1$ we have:*

$$\mathbb{P}\Big[\sum_i X_i \geq (1+\delta)\mu\Big] \leq e^{-\delta^2\mu/3} \tag{14}$$

$$\mathbb{P}\Big[\sum_i X_i \leq (1-\delta)\mu\Big] \leq e^{-\delta^2\mu/2} \tag{15}$$

In our applications $p := \frac{\lambda}{n}$. Equation (14) will be applied to $m := (1-\epsilon)n/2$ and $1+\delta = 1/(1-\epsilon)$, while Equation (15) will be applied to $m := (1+\epsilon)n/2$ and $1-\delta = 1/(1+\epsilon)$

C Extra details for Theorem 3

C.1 BA with uninstantiated self-sortition: genericBA

To obtain Theorem 3, we introduce a general setup mechanism which we are going to illustrate on the following protocol, which we call genericBA. It is obtained from [2, §5.2] as follows. We simplified it by downgrading eligibility-to-send-a-given-message, into eligibility to speak in a given round, and also by removing the termination mechanism. Moreover, we generalized it by leaving it operate from an ideal functionality for eligibility to speak, as long as it has the interface $\mathcal{F}_{\text{eligib}}$-interface, which we specified in Figure 2. All messages are signed. We assume a key-evolving signature scheme as in [37, 27] (we refer to the formalism of [27, Figure 1], and their proof in §B that it is implemented by the standard definition). When being instructed to *conditionally multicast* a given round-r message, a player P does the following. It queries $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(r)$. If returned 1, then we say that it is *eligible to speak in round* r. If so, then it multicasts the message with its signature, then updates its signing key to round-$(r+1)$, and finally erases its old (round-r) signing key. Hence, even if it gets corrupt in the same round r, the adversary cannot anymore make P issue signed round-r messages.

On receiving a round-r message from some player P, a player Q processes it if and only if $\mathcal{F}_{\text{eligib}}.\mathsf{verify}(P, r)$ returns 1.

The protocol runs in iterations $v = 1, 2, \ldots$. The first iteration consists of the two rounds $r = 1, 2$, while, higher iterations $v \geqslant 2$ consist of four rounds $r = 2 + 4(v-1) + j$, $j \in \{1, 2, 3, 4\}$. To ease the presentation, for each iteration $v \geqslant 2$ we call status,propose,vote, and commit the round numbers corresponding to $j = 1, 2, 3, 4$, while the first two round numbers $r = 1, 2$ are dubbed vote and commit.

A collection of λ (signed and $\mathcal{F}_{\text{eligib}}$-eligible) iteration-$v$ vote messages for the same value x from distinct players is said to be a view -v certificate for x. A certificate from a higher iteration is said to be a higher certificate. Below is the protocol for an iteration. The protocol for the first iteration v =1 skips the first two rounds (status and propose).

1. *Status.* Every player conditionally multicasts a status message of the form (status, r, x, c) containing the highest certified value x it has seen so far as well as the corresponding certificate c.
2. *Propose.* Every player P conditionally multicasts a propose message of the form (propose, r, x, c) where x is a value with a highest certificate known to P, denoted c. Ties between two highest ranked values are broken arbitrarily. To unify the presentation, we say that a value without any certificate has an iteration-0 certificate and it is treated as the lowest ranked certificate.
3. *Vote.* In the first iteration $v = 1$, a player conditionally multicasts $(\mathsf{vote}, v = 1, x)$ where x is its input value.
   For all iterations $v \geqslant 2$, if a (signed and $\mathcal{F}_{\text{eligib}}$-eligible) $(\mathsf{propose}, v, x, c)$ message has been received with a certificate c for x, and if the player has not observed a strictly higher certificate for a conflicting value $x' \neq x$, it conditionally multicasts an iteration-v vote message for x, of the form (vote, v, x), attached with the above iteration-v propose message.
   //Importantly, even if the player has observed a certificate for a conflicting value $x' \neq x$ from the same iteration as v, it will still vote for x.

4. *Commit.* If a player has received $\lambda/2$ iteration-v ($\mathcal{F}_{\text{eligib}}$-eligible and signed) votes for the same x from distinct players (which form an iteration-v certificate for x) and no iteration-v vote for a conflicting value $x' \neq x$, it multicasts an iteration-v commit message for x of the form (commit, v, x) with the certificate c attached.

$*$ output - *without termination.* (This step is not part of the iteration and can be executed at any time.) If a player has received $\lambda/2$ commit messages for the same x from the same iteration from distinct player, it outputs x. This last message will make all other honest player *conditionally multicast* the same terminate message, output x and terminate in the next round.

C.2 Proof of Properties 16

[Round complexity to output.*]* In the proof of their [2, Corollary 1] it is used that, if an iteration v satisfies both conditions [leader] and [committees], then all players output by the end of v. This shows that all players output by the end of V^{luck}. Notice that they call "good" an iteration are soon as it satisfies [leader], thus conflicting with our terminology.

[Consistency.] In the proof of [2, Thm 5] it is shown that consistency holds if both conditions (i) and (ii) hold.

[Strong unanimity.] In the proof of [2, Thm 6] it is shown that strong unanimity (which is called "validity") holds if both conditions (i) and (ii) hold.

Notice that all probabilities of success stated in [2, §5.3] are implicitely exponentiated by the (constant) expected number of rounds before all players output.

C.3 Adding termination to genericBA

Now, in addition, there is one type of message which players may be instructed to conditionally multicast at anytime, called terminate. To unify the presentation, we say that it is a "round-$\perp$ message". In turn, $\mathcal{F}_{\text{eligib}}$ is updated to allow $\perp$ round numbers as input.

$*$ output - *with termination.* (This step is not part of the iteration and can be executed at any time.) If a player has received $\lambda/2$ commit messages for the same x from the same iteration from distinct players, it *conditionally multicasts* a termination message of the form $(\text{terminate}, x)$ with the $\lambda/2$ commit messages attached. The player then outputs x and terminates. This last message will make all other honest player *conditionally multicast* the same terminate message, output x and terminate in the next round.

Since players can terminate, the upper-bound V^{luck} on the round complexity is not good anymore for some executions. Indeed, it could be the case that half of the honest players terminate before V^{luck} happens, then from this point there will not be enough honest players querying $\mathcal{F}_{\text{eligib}}$ to become eligible, hence V^{luck} may well never happen. For this reason we now bound the round complexity by $V := \min(V^{\text{term}}, V^{\text{luck}})$, where V^{term} is the first iteration from which enough players have terminated. Precisely, we define a parameter $\epsilon^{\text{term}} < \epsilon$ such that, when a threshold fraction of players $\epsilon^{\text{term}} n2$ have terminated, hence, queried $\mathcal{F}_{\text{eligib}}.\mathsf{speak\text{-}request}(\perp)$, then with overwhelming probability at least one of them was eligible, hence has made all other players terminate. In [1, §5.3] it is implicitely set $\epsilon^{\text{term}} = \epsilon/2$. Moreover, for all rounds before V^{term}, ϵ must be set large enough such that, despite up to $< \epsilon^{\text{term}} n$ honest players having terminated, the remaining $(\epsilon - \epsilon^{\text{term}})n$ are numerous enough to guarantee the conditions [committees.], (i) and (ii) of Properties 16 with overwhelming probability.

C.4 Reminder of the idealized VRF of [27]

$\mathcal{F}_{\text{VRF}}$

$\mathcal{F}_{\text{VRF}}$ interacts with all players $P \in \mathcal{P}$ and the adversary $\mathcal{A}$. Session identifiers (*sid*) are omitted.

Key generation Upon receiving (keyGen) from a player P, hand (keyGen, P) to $\mathcal{A}$. Upon receiving (verificationKey,P, vk) from $\mathcal{A}$, if P is honest, verify that no pair of the form $(\cdot, vk)$ is already stored, store the pair (P, vk) and return (verificationKey, vk) to P. Initialize the table $T(vk, \cdot)$ to empty.

Malicious key generation Upon receiving (keyGen, vk) from $\mathcal{A}$, ignore if vk is already stored. Initialize the table $T(vk, \cdot)$ to empty and record the pair $(\mathcal{A}, vk)$.

VRF evaluation Upon receiving a message (eval, m) from P, verify that some pair $(P,\ vk)$ is recorded. If not, then ignore the request. Then, if the value $T(vk, m)$ is undefined, pick a random value $y \xleftarrow{\$} \{0,1\}^{\kappa}$ and set $T(vk, m) = (y, \varnothing)$. Then output (eval, y) to P, where y is such that $T(vk, m) = (y, S)$ for some S.

VRF evaluation and proof Upon receiving (evalProve, m) from a player P, ignore if no pair (P, vk) is recorded. Else, send (evalProve, P, m) to $\mathcal{A}$. Upon receiving (evalProve, m, π) from $\mathcal{A}$, if value $T(vk, m)$ is undefined, verify that π is unique, pick a random value $y \xleftarrow{\$} \{0,1\}^{\kappa}$ and set $T(vk, m) = (y, \{\pi\})$. Else, if $T(vk, m) = (y, S)$, set $T(vk, m) = (y, S \cup \{\pi\})$. In any case, output (eval, y, π) to P.

Malicious VRF evaluation Upon receiving (eval, vk, m, S)$*$ from $\mathcal{A}$ for some vk, do the following. First, if $\big\{(\mathcal{A}, vk)$ or (P, vk) for P corrupt$\big\}$ is recorded and $T(vk, m)$ is undefined, then choose a random value $y \xleftarrow{\$} \{0,1\}^{\kappa}$ and set $T(vk, m) = (y, S)$ and output (eval, y) to $\mathcal{A}$. Else, if $T(vk, m) = (y, S')$ for some $S' \neq \varnothing$, union S to S' and output (eval, y) to $\mathcal{A}$, else ignore the request.

Verification Upon receiving (verify, m, y, π, vk) from some player P, send (verify, m, y, π, vk') to $\mathcal{A}$. Upon receiving (verification, m, y, π, vk') from $\mathcal{A}$ do:

1. If $vk' = vk$ for some stored $(\cdot, vk)$ and the entry $T(vk, m)$ equals (y, S) with $\pi \in S$, then set $b =$ accept.
2. Else, if $vk' = vk$ for some stored $(\cdot, vk)$, but no entry $T(vk, m)$ of the form $(y, S \ni \pi)$ is stored, then set $b =$ reject.
3. Else, initialize the table $T(vk', \cdot)$ to empty, and set $b =$ reject.

Output (verification, m, y, π, b) to P.

$*$ The π in [27, Fig. 2] obviously seemed to be an S.

Figure 10: VRF, idealized as an ideal functionality, following [27, Fig. 2].

C.5 Proof of implementation of $\mathcal{F}_{\text{eligib}}^{\text{bias}}$ by $\Pi_{\text{eligib}}^{\text{bias}}$

We describe a simulator in Figure 11.

The comments in the description make it clear that the evaluations $y \in [0,1]$ returned by the simulated $\mathcal{F}_{\text{VRF}}$ are compatible with the eligibility bits sampled by $\mathcal{F}_{\text{eligib}}^{\text{bias}}$ (both for simulated corrupt players and for dummy honest players). It remains to show that these evaluations are uniformly independently distributed distributed in $[0,1]$, each conditioned on the corresponding output bit. This fact follows from the way the $y \in [0,1]$ are sampled, since each of them is sampled equal to:

$$y \leftarrow \mathbb{1}_{y \leqslant \mathfrak{p}}.1 + \mathbb{1}_{y > \mathfrak{p}}.0 \ . \tag{18}$$

C.6 Optimizations, and removing the simplifications made in genericBA

In Algorand [37], there is a much more efficient self-sortition of a leader than the one of [1], which we imported in genericBA. The implementation is that, in a propose round r, a player multicasts as soon as its VRF evaluation is below the threshold: $20/n$. The number 20 is from [8] but can be adapted, the idea is that it is larger than the threshold which we specified so far following [1], i.e., $1/n$. Thus 10 honest players in expectation are eligible to propose. Each player considers as the proposer the one with the smallest VRF evaluation (and ignore the other propose messages). So with this refined mechanism, the [leader.] condition fails if the VRF evaluation of a corrupt player is smaller than the smallest one of all honest players. Although the adversary can try q different seeds during the setup, the interesting point is that *it doesn't know the VRF evaluations of honest players corresponding to each seed*. So it could be the case that, over the q different seeds tried, it adopts one σ such that a honest player will turn out to have a lower VRF evaluation in one of the V first iterations, whereas this would not have happened with another seed σ' tried. So intuitively, there is hope to obtain an upper-bound on the probability of failure which is strictly better than the union bound over all q tried seeds. We leave it for future work.

We furthermore believe that there may exist a tighter upper-bound for the round complexity of genericBA than the one given by Properties 16, in particular in the case of binary BA. Concretely, it is possible that players terminate in an iteration with an honest player eligible to propose, despite some corrupt players concurrently multicasting propose. For instance, we could optimize the protocol by specifying that, if in a given iteration $v+1$, players are reported an iteration-v certificate c (likely: from an honest player eligible to speak in the status round), then they consider c as a propose. From there, assuming ties to vote between two conflicting iteration-v certificates are in favour of c, we thus have that all players terminate in $v+1$.

We now turn to instantiations of the VRF. The idealized model of VRF which we used, borrowed from [27] and defined in Appendix C.4, can be instantiated as suggested by [37] and adopted in [53, §4.2.1]. Namely: sign the value to-be-evaluated with a unique signature scheme, then apply a random oracle on the signature. Turning to post-quantum VRFs, the most promising one is the one introduced in [32], which realizes a weaker-but-sufficient primitive. Namely, they allow a maximum number of evaluations fixed from the public key (which is the root of a Merkle tree), and each proof of evaluation has logarithmic size in this maximum number.

We now explain how to possibly remove the use of secure memory erasures, which we assumed in genericBA. To this end, let us recall the technique of [1, §5.2], which applies only of the space of values is small, e.g., typically binary BA. In [2, §5.2] $\mathcal{F}_{\text{eligib}}$ checked eligibility to send every specific message content m, instead of just the round number of the message, as we did. Hence in [2, §5.2], even if a player P gets corrupt after eligibly multicasting a round-r message m, the adversary may not be able to make P eligibly multicast a round-r conflicting m'. So this removed the need for P to securely erase its round-r signing key from its memory. Our mechanism for implementing $\mathcal{F}_{\text{eligib}}$ is obviously compatible with this refinement. In turn, not to degrade too much the probabilities with the union bound over all message contents m, the BA of [2, §5.2] applies to only a small number of possible values (in their case: binary). That way, the number of possible message contents m is limited (in their case: two). It seems to us that secure memory erasure is regarded as a realistic, given the number of areas based on it (forward secrecy, e.g., in TLS and Signal, and proactive security [14]).

We will explain in Appendix C.3 how to re-incorporate the termination mechanism of the BA of [1, §5.2], at the cost of a larger ϵ as in [1, §5.3]. Notice that termination is unecessary in a chain-of-BA-instances regime, i.e., a blockchain ([37, 26, 27]).

It should be clear from the statement of Properties 16 that our mechanism also applies to bootstrap the setup of other baseline BAs than [1, §5.2], e.g., those [37, 26, 27] in which players output an old-enough prefix of their observed chain of proposed blocks. Notice that those alternative BAs offer a trade-off: by increasing the round complexity, they enable to reduce the corruption threshold gap: ϵ, since they allow conflicting chains to be produced.

Simulator $\mathcal{S}$ for $\Pi^{\text{bias}}_{\text{eligib}}$

$\mathcal{S}$ initializes simulated honest players, simulated corrupt players, and internal copies of a bulletin-board and of $\mathcal{F}_{\text{VRF}}$. In particular, in addition to its interfaces for the simulated players, the simulated $\mathcal{F}_{\text{VRF}}$ offers its adversary interface to the environment $\mathcal{Z}$. The simulated bulletin-board follows its intended behavior. The simulated $\mathcal{F}_{\text{VRF}}$ processes all $\mathcal{F}_{\text{VRF}}$.verify requests following its intended behavior.

Setup. - $\mathcal{S}$ makes simulated honest players follow the protocol, i.e., query $\mathcal{F}_{\text{VRF}}$.keyGen, then publish on the bulletin-board the keys received. It makes the simulated $\mathcal{F}_{\text{VRF}}$ process the keyGen requests as specified, in particular, it forwards them to $\mathcal{A}$ then delivers the keys received from $\mathcal{A}$ to the simulated honest players.

- **eval(vk, m, π) during setup.** Upon receiving such a request on behalf of $\mathcal{F}_{\text{VRF}}$ during the setup, so which comes from a corrupt player or $\mathcal{A}$ directly, check if m is of the form:

$$m = (\sigma', r) \text{ s.t. } \sigma' = (vk'_1, \ldots, vk'_n) \text{ and } \exists i, vk = vk'_i \tag{16}$$

and furthermore if it comes from a corrupt P_j, then: check if $j = i$. If not, then $\mathcal{F}_{\text{VRF}}$ processes the request following its intended behavior. Else, i.e., if all checks pass:
//now, $\mathcal{S}$ must craft a VRF output value which is compatible with the output bit which the environment will observe upon instructing a dummy honest player to check if P_i is eligible to speak in round r.
send (re-seed, σ') to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ then send speak-request(r) to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ on behalf of the simulated corrupt P_i. Upon receiving the output, which we denote $coin[\sigma', P_i, r]$: if it is equal to 0, then set the evaluation as $y \xleftarrow{\$} \mathcal{U}([\mathfrak{p}, 1])$, i.e., equal to a uniform sample in $[\mathfrak{p}, 1]$, else, set it as $y \xleftarrow{\$} \mathcal{U}([0, \mathfrak{p}])$.

- Just before $t = 0$: denote $\sigma \leftarrow (vk_1, \ldots, vk_n)$ the list of keys published on the bulletin-board. Send (re-seed, σ) to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ one last time.

Real honest request to speak. Upon being leaked by $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ that one real dummy honest player P_j was returned an output bit: $coin[\sigma, P_j, r]$ to its request to speak in some round r. //now, $\mathcal{S}$ must set a VRF evaluation which is compatible with the output bit: $coin[\sigma, P_j, r]$ which the environment observed output by the dummy honest P_j.
Make the simulated honest P_j request $\mathcal{F}_{\text{VRF}}$.evalProve$((\sigma, r))$. Set the evaluation $T(vk_j, (\sigma, r)) \leftarrow y$ as follows: if $coin[\sigma, P_i, r] = 0$, then set $y \xleftarrow{\$} \mathcal{U}([\mathfrak{p}, 1])$, else, set it as $y \xleftarrow{\$} \mathcal{U}([0, \mathfrak{p}])$. Finally, make the simulated honest P_j multicast (r, y, π), where π is the VRF proof received on behalf of the adversary from the environment.

Malicious eval(vk, m, π) after setup upon receiving such request from a corrupt player or $\mathcal{A}$ directly, check if m is of the form:

$$m = (\sigma, r) \text{ and } \exists i, vk = vk_i \tag{17}$$

and furthermore if it comes from a corrupt P_j, then: check if $j = i$. If not, then $\mathcal{F}_{\text{VRF}}$ processes the request following its intended behavior. Else, i.e., if all checks pass:
//now, $\mathcal{S}$ must craft a VRF output value which is compatible with the output bit which the environment will observe upon instructing a dummy honest player to check if P_i is eligible to speak in round r.
send speak-request(r) to $\mathcal{F}^{\text{bias}}_{\text{eligib}}$ on behalf of the simulated corrupt P_i. Upon receiving the output, which we denote $coin[\sigma, P_i, r]$: if it is equal to 0, then set the evaluation as $y \xleftarrow{\$} \mathcal{U}([\mathfrak{p}, 1])$, i.e., equal to a uniform sample in $[\mathfrak{p}, 1]$, else, set it as $y \xleftarrow{\$} \mathcal{U}([0, \mathfrak{p}])$.

Figure 11

D Details for the proof of Theorem 4

Let us formalize the bounds on the probabilities of failure in Theorem 4, in order to derive the claimed $\eta \geqslant 1/6$.

For each $b \in \{0,1\}$, let us denote $X_{h,b}$ and $\overline{X_{h,1-b}}$ the events in $W_{h,b}$ that the real execution, resp., the simulated one, satisfies simultaneously consistency and multicast complexity at most f. Then by assumption and an intersection bound, we have:

$$\mathbb{P}(X_{h,b} \cap \overline{X_{h,1-b}}) \geqslant 1 - 2\eta \ . \tag{19}$$

Furthermore, the distribution of the view of p is the same in both $X_{h,b} \cap \overline{X_{h,1-b}}$, $b \in \{0,1\}$. So there is a bit B' that p does not output in both $X_{h,b} \cap \overline{X_{h,1-b}}$, $b \in \{0,1\}$ with probability at least 1/2. Assume without loss of generality that this bit B' is 1. Combined with Equation (19), this yields in particular:

$$\mathbb{P}(X_{h,1} \cap \overline{X_{h,1}} \cap \{\text{p does not output } 1\}) \geqslant 1/2(1 - 2\eta) \ . \tag{20}$$

On the other hand, recall that the view of so far honest players other than p is equally distributed in $W_{c,b}$ and $W_{h,b}$, and recall furthermore that in each $W_{c,b}$ they output b with probability at least $1-\eta$. Hence, they output 1 in each $W_{h,b}$ with probability at least $1-\eta$, in particular, output 1 in $W_{c,1}$ with probability $1-\eta$. Intersecting with Equation (20), we obtain a consistency violation with probability at least $1/2(1-2\eta) - \eta = 1/2 - 2\eta$. By assumption, this quantity must itself be smaller than η. In conclusion, we obtain $\eta \geqslant 1/6$, as claimed.

E The important Lemma 19 for the proof of Theorem 5

Recall that the proof of Theorem 5 relied on the existence of a partition of players: $\mathcal{P} = \mathcal{S}_0 \cup \{h_0\} \cup \mathcal{S}_0'$, such that the player h_0 often sends no message in both worlds W_{AH} and W_{HH}. This is somewhat analogous to the proof of [28, Theorem 1], which was based on existence of a player which sends few messages. However, existence in [28, Theorem 1] is easily proven since they consider a *fixed* world (in which all players are honest). By contrast, the additional difficulty here is that the definitions of both worlds W_{AH} and W_{HH} themselves *depend* on the choice of the partition of players $\mathcal{P} = \mathcal{S}_0 \cup \{h_0\} \cup \mathcal{S}_0'$. Thus, a standalone averaging over players does not prove anymore existence of such a h_0. Instead, we must consider simultaneously many worlds, thus the following notations.

For $\mathcal{I}$ any set of players, we denote $W_{HA}(\mathcal{I})$ the world in which $\mathcal{I}$ is honest and assigned input 1, while the adversary corrupts $\overline{\mathcal{I}} := \mathcal{P} \backslash \mathcal{I}$ and makes them play honestly as if having input 0. For instance, with the previous notations of Section 1.3.2, we have $W_{HA} = W_{HA}(\mathcal{S} \cup \{h\})$.

For $\mathcal{S}$ any set of players, we denote $W_{HH}(\mathcal{S})$ the world in which $\mathcal{S}$ is honest with input 1, the remaining players $\overline{\mathcal{S}} := \mathcal{P} \backslash \mathcal{S}$ are also honest with input 0. For instance, with the previous notations of §1.3.2, we have $W_{HH} = W_{HH}(\mathcal{S})$.

Likewise, we denote $\mathbb{P}_{HA(\mathcal{I})}$ and $\mathbb{P}_{HH(\mathcal{S})}$, and $\mathrm{E}_{HA(\mathcal{I})}$ and $\mathrm{E}_{HH(\mathcal{S})}$ the probability laws and expectations in $W_{HA}(\mathcal{I})$ and $W_{HH}(\mathcal{S})$.

Lemma 19. *Let $\eta \geq 0$ be such that, with probability at least $1-\eta$, at most C distinct honest players send messages in the whole execution. Then there exists a player $h_0 \in \mathcal{P}$, along with a subdivision of the set of players: $\mathcal{S}_0 \cup \{h_0\} \cup \mathcal{S}'_0 = \mathcal{P}$ with $|\mathcal{S}_0| = |\mathcal{S}'_0| = f$, such that, denoting*

$$p_{h_0}(\eta, C) := 2\Big(\frac{(1-\eta)C}{f+1} + \eta\Big) \tag{21}$$

then in each *world $W_{HA}(\mathcal{S}_0 \cup \{h_0\})$ and $W_{HH}(\mathcal{S}_0)$ it holds that, with probability at least $1 - p_{h_0}$, h_0 sends* no *message.*

Proof. For every h, we denote $\mathbb{1}_h$ the function equal to 1 when h sends at least one message in the execution and 0 otherwise. For a fixed set $\mathcal{S}$ of cardinality f not containing h, we denote $p_h(\mathcal{S}) := \mathrm{E}_{HA(\mathcal{S}\sqcup\{h\})}\big(\mathbb{1}_h\big) + \mathrm{E}_{HH(\mathcal{S})}\big(\mathbb{1}_h\big)$. Then, to prove the Lemma, it is enough to show existence of a $\mathcal{S}_0$ and h_0, such that $p_{h_0}(\mathcal{S}_0) \leq 2\Big(\frac{(1-\eta)C}{f+1} + \eta\Big)$

To this end, let us upper-bound the following double sum: $Sum := \sum_{|\mathcal{S}|=f} \sum_{h \notin \mathcal{S}} p_h(\mathcal{S})$. We replacing $p_h(\mathcal{S})$ by its expression. To sum the first summand: $\mathrm{E}_{HA(\mathcal{S}\sqcup\{h\})}\big(\mathbb{1}_h\big)$, we make the change of variable $\mathcal{I} := \mathcal{S} \sqcup \{h\}$, i.e., we add h to the summation index set $\mathcal{S}$. We leave unchanged the summation of the other summand $\mathrm{E}_{HH(\mathcal{S})}\big(\mathbb{1}_h\big)$. We deduce:

$$Sum = \sum_{\mathcal{I}} \sum_{h\in\mathcal{I}} \mathrm{E}_{HA(\mathcal{I})}\big(\mathbb{1}_h\big) + \sum_{\mathcal{S}} \sum_{h\notin\mathcal{S}} \mathrm{E}_{HH(\mathcal{S})}\big(\mathbb{1}_h\big)\ . \tag{22}$$

Let us consider the left double-sum. In each fixed $\mathcal{I}$, we are summing, over honest players h, the expectation of h to send at least one message in the execution. By assumption, $\sum_{h\in\mathcal{I}} \mathbb{1}_h \leq C$ with probability at least $1-\eta$ in W_{HA}. On the remaining events, this sum $\sum_{h\in\mathcal{I}} \mathbb{1}_h$ over some $f+1$ honest players cannot exceed $f+1$ by definition. Overall, we deduce this upper bound on the left summand: $\sum_{h\in\mathcal{I}} \mathrm{E}_{HA(\mathcal{I})}\big(\mathbb{1}_h\big) \leqslant (1-\eta)C + \eta(f+1)$.

Let us consider the right double-sum, and repeat the same argument. We obtain the same upper-bound on the right summand: $\sum_{h\notin\mathcal{S}} \mathrm{E}_{HH(\mathcal{S})}\big(\mathbb{1}_h\big) \leqslant (1-\eta)C + \eta(f+1)$.

Upper-bounding Equation (22) using the upper bounds just obtained, we obtain two sums, over summation indices: $\mathcal{I}$ and $\mathcal{S}$, which both vary in a set of cardinality $\binom{n}{f+1} = \binom{n}{f}$. Overall, we deduce the upper-bound $Sum \leq \binom{n}{f} 2((1-\eta)C+\eta(f+1))$. But coming back to the definition of Sum, it consists of $\binom{n}{f}(f+1)$ summands (since it is summed over $(\mathcal{S}, h \notin \mathcal{S})$) which are all *non-negative*. From this we deduce existence of one index, which we denote as $(\mathcal{S}_0, h_0)$, such that the corresponding summand $p_{h_0}(\mathcal{S}_0)$ is lower than or equal to the claimed $p_{h_0}(\eta, C)$.

F Proof of Theorem 10: impossibility of partially synchronous randomized consensus for $f \geq n/3$

We show the result for $n = 3$ players: P_0, P_1, η of which at most $f = 1$ is corrupt. The case of general n follows from the well-known reduction technique of [42, §2]. We show the result for VBA, then explain how to adapt the proof to BA. We consider the classical validity predicate which returns accept on a value σ if and only σ is a valid signature (on any message) of some predefined external entity called E. For simplicity we consider idealized digital signatures, as recalled in Appendix B.4. Concretely, we will consider a scenario (called real below) where the honest player P_0 saw only a signature σ_0 on 0, so is unable to forge any other valid value. In this same scenario, honest player P_1 saw only a signature σ_1 on 1, so is unable to forge any other valid value

Let us formalize the assumption: there exists a VBA and a fixed probability η such that for all adversaries and input assignment,

$$\mathbb{P}\Big[\text{Consistency, External validity and Termination}\Big] \geqslant 1 - \eta \,. \tag{23}$$

We now use the same reduction as in Section 2.3. Namely, for any fixed world, up to replacing η by any arbitrarily close value $\eta - \mu$, we can consider that Equation (23) is strenghtened with: *[all players* output *within R rounds]*, where R depends on μ (and which takes poly(κ)-bounded values). For ease of notation we will call R an "essential upper-bound on the round complexity" in this given world.

We consider three worlds: $W_0 \leftrightarrow$ real $\leftrightarrow W_1$, where the $\leftrightarrow$ denotes an indistinguishability between the views of some players.

- World W_0: GST $= 0$, P_0 and P_2 are honest and are assigned input σ_0, P_1 is corrupt and forever silent.

- World W_1: GST $= 0$, P_1 and P_2 are honest and are assigned input σ_1, P_0 is corrupt and forever silent.

- World real: GST $= \max(R_{(0)}, R_{(1)}) + 1$, where $R_{(0)}$ and $R_{(1)}$ denote essential upper-bounds on the round complexities in the W_0 and W_1 worlds. P_0 and P_1 are honest with inputs σ_0 and σ_1, while P_2 is corrupt. All messages sent between P_0 and P_1 are delayed until GST $+ 1$. P_2 runs two threads in parallel denoted $P_2^{(0)}$ and $P_2^{(1)}$. For each $b \in \{0,1\}$, $P_2^{(b)}$ follows honestly the protocol as if starting with input σ_b, but ignores P_{1-b}. A way to formalize this is that messages from P_{1-b} to P_2 are delivered only to the thread $P_2^{(1-b)}$, while messages in the outgoing mailbox from $P_2^{(b)}$ to P_{1-b} are destroyed by the adversary instead of being sent.

Indistinguishability between W_0 and real. The view of P_0 in real until GST is distributed as in W_0. Thus, P_0 ouputs a valid value in real before GST with probability $\geqslant 1 - \eta$. This output can only be σ_0, since P_0 did not see any other valid value.

Indistinguishability between W_0 and real. The view of P_1 in real until GST is distributed as in W_1. Thus P_1 ouputs a valid value in real before GST with probability $\geqslant 1 - \eta$. This output can only be σ_1, since P_1 did not see any other valid value.

In conclusion, the probability of a consistency violation in real is $\geqslant 1 - 2\eta$, which must be smaller than η by assumption, hence $\eta \geqslant 1/3$ as claimed.

The proof carries unchanged over BA. The only difference lies in the argumentation. Namely, P_0 now outputs σ_0 in W_0 by *strong unanimity*, not anymore by unforgeability of any other valid value than σ_0. Likewise, P_1 now outputs σ_1 in W_1 by strong unanimity.