Adaptive Security, Erasures, and Network Assumptions in Communication-Local MPC

Nishanth Chandran[1], Juan Garay[2], Ankit Kumar Misra[3], Rafail Ostrovsky[3⋆], and Vassilis Zikas[4]

[1] Microsoft Research, India.
nichandr@microsoft.com
[2] Texas A&M University
garay@tamu.edu
[3] University of California Los Angeles
{ankitkmisra,rafail}@cs.ucla.edu
[4] Purdue University
vzikas@cs.purdue.edu

Abstract. The problem of reliable/secure all-to-all communication over low-degree networks has been essential for *communication-local* (CL) n-party MPC (i.e., MPC protocols where every party directly communicates only with a few, typically polylogarithmic in n, parties) and more recently for communication over *ad hoc* networks, which are used in blockchain protocols. However, a limited number of adaptively secure solutions exist, and they all make relatively strong assumptions on the ability of parties to act in some specific manner before the adversary can corrupt them. Two such assumptions were made in the work of Chandran *et al.* [ITCS '15]—parties can (a) *multisend* messages to several receivers simultaneously; and (b) *securely erase* the message and the identities of the receivers, before the adversary gets a chance to corrupt the sender (even if a receiver is corrupted).

A natural question to ask is: Are these assumptions necessary for adaptively secure CL MPC? In this paper, we characterize the feasibility landscape for all-to-all reliable message transmission (RMT) under these two assumptions, and use this characterization to obtain (asymptotically) tight feasibility results for CL MPC.

- First, we prove a strong impossibility result for a broad class of RMT protocols, termed here *store-and-forward* protocols, which includes all known communication protocols for CL MPC from standard cryptographic assumptions. Concretely, we show that no such protocol with a certain expansion rate can tolerate a constant fraction of parties being corrupted.
- Next, under the assumption of only a PKI, we show that assuming secure erasures, we can obtain an RMT protocol between all pairs of parties with polylogarithmic locality (even without assuming multisend) for the honest majority setting. We complement this result by showing a negative result for the setting of dishonest majority.
- Finally, and somewhat surprisingly, under stronger assumptions (i.e., trapdoor permutations with a reverse domain sampler, and compact and malicious circuit-private FHE), we construct a polylogarithmic-locality all-to-one RMT protocol, which is adaptively secure and tolerates any constant fraction of corruptions, without assuming either secure erasures or multisend. This last result uses a novel combination of adaptively secure (e.g., non-committing) encryption and (static) FHE to bypass the impossibility of compact adaptively secure FHE by Katz *et al.* [PKC'13], which we believe may be of independent interest. Intriguingly, even such assumptions do not allow reducing all-to-all RMT to all-to-one RMT (a reduction which is trivial in the non-CL setting). Still, we can implement what we call *sublinear output-set* RMT (SOS-RMT for short). We show how SOS-RMT can be used for SOS-MPC under the known bounds for feasibility of MPC in the standard (i.e., non-CL) setting assuming, in addition to SOS-RMT, an anonymous PKI.

[⋆] This research was supported in part by NSF grants CNS-2246355, CCF-2220450, US-Israel BSF grant 2022370, and by Sunday Group.

1 Introduction

1.1 Communication Locality and Adaptive Security

Secure multi-party computation (MPC) [Yao82,GMW87a,BGW88a,CCD88] allows a set of n parties to securely compute a function on their joint private data. Initial work on MPC focused on feasibility, and it was followed by a series of works on improving round and communication complexity. Envisioning the potential need to deploy MPC on massive networks, novel works on *scalable* MPC (e.g., [DI06,IPS08,DIK10,GIOZ17]) have investigated settings and techniques that allowed for protocols with communication complexity that grows (asymptotically) slower than the size of the player set. Boyle, Goldwasser, and Tessaro [BGT13a] put forth a different metric that is very relevant for the design of massive-scale MPC, namely, *communication locality* (CL). The CL of a party in a protocol is the number of parties that this party sends/receives messages to/from, via a direct point-to-point channel, through the execution of the protocol; as such, the CL of a protocol is the maximum CL of any party. Motivated by the potential application of MPC to privately executing sublinear algorithms in a distributed manner, [BGT13a] proposed a solution which achieves MPC with a sublinear (i.e., polylogarithmic in n) CL tolerating a (sub-optimal) number of $t < (1/3 - \epsilon)n$ actively corrupted parties, for $\epsilon > 0$.

The original solution in [BGT13a] only considered static corruptions and relied on the existence of a public-key infrastructure (PKI), a common reference string (CRS), semantically secure public-key encryption and existentially unforgeable signatures. Chandran *et al.* [CCG$^+$15] improved on the above result to tolerate an asymptotically *optimal* number of $t < (1/2 - \epsilon)n$ *adaptive* active corruptions, for an arbitrary small constant ϵ. Their construction relied on the same assumptions except for the CRS, which was replaced by a *hidden (random) graph*: A suitable random graph on n vertices (with sublinear degree) that is sampled by a trusted entity and where each party is given its neighborhood in this graph. However, parties do not know the other parties' neighborhoods, and, most importantly, the *adversary* does not know the (honest) neighbors of honest parties. As shown in [CCG$^+$15], this random graph can be realized via a standard symmetric-key infrastructure (SKI)—wherein every two parties share a (secret) symmetric-key encryption key. The emulation is simple: Every two nodes locally use their symmetric key as a seed to a PRF to derive sufficiently long pseudorandomness that can be utilized to decide (locally and independently) whether or not the parties should have an edge between them in any given round. In fact, we mention that, as noted in [CCG$^+$14], the above hidden graph (or, equivalently, SKI) assumption can be replaced by standard number-theoretic cryptographic assumptions (such as DDH and its resulting PKI), allowing non-interactive key exchange (NIKE, for short—cf. [FHKP13]). The simple idea is that the PKI can be used to non-interactively establish the SKI required in [CCG$^+$15], which can then be used to derive the hidden graph.

The core challenge associated with such sublinear CL protocols is propagating information and connecting any two parties using a sparse (i.e., sublinear degree) communication graph. In such a context, one needs to route messages through the induced (incomplete) communication graph so that the adversary cannot block (even indirect) communication between any two honest nodes, thus disconnecting the graph. Indeed, since any party can only directly communicate with a sublinear number of neighbors, the only way for it to reach all parties in the network is by means of a *gossiping* protocol. In [BGT13a], gossiping was done via a routing protocol based on hierarchical routing and sorting networks that cleverly knit the paths to ensure each message travels over sufficiently many paths, making it impossible for the adversary to block it.

The above gossiping protocol works for a static adversary corrupting $t < (1/3-\epsilon)n$ parties. However, when one considers stronger adversaries, with an asymptotically optimal (for MPC) corruption threshold—i.e., $t < (1/2 - \epsilon)n$—and, most importantly, adaptive adversaries, the problem becomes even more challenging, as message routing through the incomplete graph turns into a "cat-and-mouse" game—more formally, a *graph discovery* game—with the adversary using an initial set of corrupted parties to try to discover possible message routes and block them. In [CCG$^+$15], properties of a hidden Erdős-Rényi graph were used along with a clever use of edges in a disposable manner (where every edge was used only once) in order to win the above graph discovery game, and devise a sublinear-locality communication protocol for the problem of *reliable message transmission* (RMT) between any two honest nodes, which allowed every honest party to reach every other honest party. The protocol from [CCG$^+$15] tolerates an arbitrary constant fraction of the parties being adaptively (and actively) corrupted. RMT protocols can then be used to trivially construct secure message transmission (SMT) protocols—informally, these are protocols which emulate a secure, i.e., private and authenticated, channel between a sender and a receiver—in the model assuming a PKI, which can then be used to build communication local MPC protocols[5].

1.2 Erasures and Network Assumptions in CL Protocols

The protocol in [CCG^{+}15] relied on the aforementioned setup and hardness assumptions, namely, a PKI, an SKI, and the existence of enhanced one-way permutations—the latter being a common minimal assumption for MPC. In addition, an assumption was made in [CCG^{+}15] which is not essential for MPC, but, as we show in this paper, turns out to be *necessary* for sublinear communication when natural gossiping protocols are used. In more detail, the assumption of *secure erasures* [CFGN96a]—namely, that honest parties can erase whichever part of their state they wish, in a way that if they are corrupted later on, the adversary cannot recover the erased information—is only needed for a subset of adaptively secure MPC protocols without the sublinear CL restriction [GS12]. The construction from [CCG^{+}15], however, assumes not just erasures, but actually two levels of strengthening of the assumption: First, it also assumes an *atomic multisend* capability [HZ10], which in a nutshell ensures that if a player p attempts to send a message to a subset $\mathcal{Q}$ of the player set in some given round, then either all honest parties will receive the message or none of them will[6]. However, even assuming secure erasures in addition to such a rushing adversary proves not to be sufficient for the protocol in [CCG^{+}15]. The reason is that when a message is sent to a polylogarithmic (in n) number of parties, then one of these parties may be corrupted, in which case the adversary can corrupt the sender and learn who the other receivers were, *before* the sender has had a chance to erase their identities, and corrupt them too, thereby completely neutralizing the sender. In fact, the inability of the adversary to mount such an attack is essential in [CCG^{+}15]'s security proof. In order to exclude this attack, [CCG^{+}15] also assumes that multisend and erase can jointly be done as an atomic operation—i.e., p can send his message so that it is received by all the parties in $\mathcal{Q}$ and erase their identitites before the adversary is able to corrupt him.

1.3 Our Results

The above state of affairs leaves open several questions regarding the minimal assumptions required for sublinear locality in all-to-all communication (and therefore also in MPC) in the adaptive security setting. In particular, it leaves open the question of the necessity and sufficiency of secure erasures, atomic multisend, and their atomic combination as mentioned above. In this paper we provide a characterization of this landscape, as depicted in Table 1. The impossibility results in Table 1 are for adaptively secure all-to-all reliable message transmission, i.e., the task of allowing every party p_i to send a (potentially different) message to every party p_j in a reliable, i.e., authenticated manner—where p_j becomes aware that the message was sent by p_i—so that the adversary cannot block or alter the message exchanges between any pair of honest parties. As noted earlier, this serves as a building block for SMT and CL MPC protocols. These results apply to a broad class of protocols which we call *store-and-forward* (SF) protocols, which, intuitively, allow intermediate parties to only store and forward previously received messages, and (for the non-erasure case) under an *expansion-rate* assumption, which mandates that messages originating from any neighbors of a sender will reach a large (polylogarithmic size) set relatively fast (i.e., before they reach their respective receiver) (see Definitions 1 and 2). This includes several natural message propagation and gossiping protocols, and in particular all those used in the CL literature (see Section 2 for a discussion).

The positive results in Table 1 are for all-to-one RMT, i.e., there is one receiver who everyone wishes to send messages to. As we shall show, assuming erasures, the feasibility results can be extended to all-to-all

[5] As a side note, an interesting side effect of the recent popularity of blockchain protocols is that, as they also rely on gossiping for communication, results on the feasibility of sublinear-communication protocols provide insights on basic feasibility questions in the blockchain context as well (see related work for further details).

[6] As shown in [HZ10,KMTZ13], this property is impossible to obtain from simple point-to-point communication in the standard adaptive and rushing adversary setting [Can00].

RMT. However, intriguingly, in the non-erasure setting, the protocol can only be extended to allow for a sublinear (polylogarithmic) set of receivers. Even so, the assumption of an additional setup, namely, an *anonymous* PKI, allows us to rescue the situation: We show that we can implement a new notion of MPC, which we term *sublinear output-set* MPC (SOS-MPC for short). Likewise, we use the term SOS-RMT to refer to the formerly obtained notion of RMT with a sublinear set of receivers. Intuitively, in an anonymous PKI setting, parties have access to a PKI but do not know which public-key corresponds to which party. Such a setup is common in YOSO-style MPC protocols which have become highly relevant in the blockchain literature [GHK$^+$21]. Our newly defined SOS-MPC is similar to standard MPC (i.e., the inputs of all parties are accounted for in the computation) but only a (random) subset of the parties of sublinear size receives the output from the computation. We note that SOS-MPC is sufficient for the motivating applications of CL MPC, namely, secure computation of sublinear algorithms, where the output is by definition far smaller than the input (cf. [BGT13b]). This leaves open the question of feasibility or impossibility of (all-to-all RMT and) standard MPC in this setting.

A-MS	Erasures	A-MSE	
✓	✓	✓	[CCG$^+$15]: Assuming *PKI and one-way functions*, there exists an (SF) sublinear locality RMT protocol tolerating $t < (1-\epsilon)n$ corruptions, for any $\epsilon > 0$.
✓	✗	✗	**Theorem 1:** There exists no SF polylogarithmic locality RMT protocol with expansion rate $(\mathsf{polylog}(n), \frac{k \log n}{c \log\log n})$ tolerating a constant fraction of corruptions, for any $k < 1$, where the degree of the communication graph is $O(\log^c n)$.
✗	✓	✗	**Theorem 2:** Assuming *PKI and one-way functions*, there exists an (SF) polylogarithmic locality RMT protocol tolerating $t < (\frac{1}{2} - \epsilon)n$ corruptions, for any $0 < \epsilon < \frac{1}{2}$. **Theorem 3:** There exists no SF polylogarithmic locality RMT protocol tolerating $t > (\frac{1}{2} + \epsilon)n$ corruptions, for any $0 < \epsilon < \frac{1}{2}$.
✗	✗	✗	**Theorem 4:** Assuming *PKI, trapdoor permutations with a reversed domain sampler* and *compact and malicious circuit-private FHE*, there exists a polylogarithmic locality RMT protocol tolerating $t < (1-\epsilon)n$ corruptions, for any $\epsilon > 0$.

Table 1. A characterization of feasibility of reliable communication (RMT) under the different assumptions: Atomic multisend (A-MS), secure erasures (Erasures), and multi-send and secure erasures as an atomic operation (A-MSE). All negative results are for all-to-all RMT. The positive results are for all-to-one RMT; but all except Theorem 4 are extended to all-to-all RMT, whereas Theorem 4 is extended to SOS-RMT. Note that SF stands for store-and-forward as defined in Definition 1, and expansion rate is as defined in Definition 2.

Our positive results are of two flavors. Assuming secure erasures under standard assumptions (i.e., one-way functions), we provide SF protocols for RMT tolerating the (asymptotically) optimal number of corruptions, as implied by the impossibility result of Theorem 3. This leaves open the following important question:

> *Is it possible to construct a (non-SF) RMT protocol in the no-erasures setting that is adaptively secure against a constant fraction of corruptions?*

We answer this question in the affirmative, albeit using strong cryptographic assumptions. Concretely, assuming trapdoor permutations with a reversed domain sampler and the existence of a malicious, compact, and circuit-private fully-homomorphic encryption (FHE) scheme [OPP14], we can construct a protocol, which is not SF, and thus circumvents our impossibility result, allowing for RMT tolerating any constant fraction of corruptions. Its construction relies on a novel combination of adaptively secure (e.g., non-committing) encryption [CFGN96a] and (statically) secure FHE to obtain a homomorphic encryption scheme that, although not fully homomorphic, allows to compute a class of circuits sufficient for our RMT, while providing adaptive security (including deniability), a property which is known to be impossible for general FHE [KTZ13].

Regarding our MPC feasibility results, using techniques from [CCG$^+$15], we can "lift" all the above all-to-all RMT feasibility results on adaptive all-to-all communication to adaptively secure MPC, under the assumption of enhanced trapdoor permutations, or any other assumption that would allow for corruption-optimal adaptively secure MPC over a complete point-to-point network. Similarly, we can lift the SOS-RMT results to SOS-MPC.

1.4 Related Work

Although introduced as a notion explicitly for standard MPC by Boyle, Goldwasser, and Tessaro [BGT13b], the idea of low communication locality was already implicit in a number of works on *almost-everywhere* secure (communication, Byzantine agreement, and computation) protocols [KSSV06a,KSSV06b,DPPU86,Upf92,CGO10,CGO12,GO08]. Such protocols can operate over incomplete networks (or under-utilize a complete network to achieve low CL as we do here) but "give up" security for a number of parties; such so-called *doomed* parties might loose their input privacy, and/or contribute a false input to or receive a false output from the computation. The goal of such almost-everywhere secure protocols is then to achieve optimal tradeoffs between number of corruptions and number of doomed parties. We refer to [CFG$^+$22] for a recent formal treatment of and detailed literature review of almost-everywhere secure protocols.

As already mentioned, Chandran *et al.* [CCG$^+$15] improved on the resiliency of the protocol from [BGT13b] and brought adaptive security to the model, at the cost, however, of the strong atomic erase-and-multisend assumption, which restricts the ability of an adaptive adversary to attack the protocol, as we discuss in detail in Section 2. The results from [CCG$^+$15] relied on a hidden-graph setup which, by construction, was an expander graph. The follow-up work by Boyle *et al.* [BCP15] provided the first solutions to the problem for PRAM-based MPC, which in addition achieved some load balancing properties. Following that, Boyle *et al.* [BCDH18] investigated the question of whether an expander is in fact needed for sublinear locality MPC, answering it in the negative.

Also related to our goals are works that explicitly target sublinear per-party communication complexity. In this context, Dani *et al.* [DKMS12] presented a statically secure information-theoretic MPC protocol with a per-party communication complexity of $O(\sqrt{n})$ tolerating $t < n/3$ corruptions. King and Saia [KS10] showed how to construct a Byzantine agreement (BA) protocol that is secure against adaptive corruptions, where the communication complexity of every party is $\tilde{\mathcal{O}}(\sqrt{n})$, which leads to a BA protocol with $\tilde{\mathcal{O}}(n)$ communication locality tolerating $t < (\frac{1}{3} - \epsilon)n$ corruptions.

Also related to our work is the work of Matt *et al.* [MNT22], who consider a weakening of the adversary's adaptivity, which they term *delayed adaptive corruption*. Here the adversary who wants to corrupt a party needs to first indicate its intention, say, in round r, but the actual corruption does not take effect until a few rounds later. Despite being useful for making statements about the load balance and delivery guarantees of blockchain-inspired message propagation protocols, in the context of sublinear locality we are considering, this assumption trivializes feasibility questions. Indeed, the latter protocols tend to use parties (network nodes) as "disposable" relays, i.e., once a party successfully relays its message to its neighbors, corrupting it does not buy the adversary anything. This fact, in combination with the delayed adaptive corruption assumption (which would imply that the adversary's ability to corrupt is **slower** than the message propagation), would prevent the adversary from adaptively blocking discovered paths.

2 Model

Notation. Here we present some basic notation used throughout the paper. We denote by $[n]$ the set $[n] = \{1, \ldots, n\}$. $\mathcal{P} = \{p_1, \ldots, p_n\}$ denotes the set of parties participating in the MPC protocol. We will often refer to parties as nodes in a network. We will assume that the adversary is able to corrupt a number $t < n$ of the parties; it will be convenient for our exposition to express corruption in terms of a fraction τ of the total number of parties; hence $t = \tau n$, for $0 < \tau < 1$.

In any directed graph over $\mathcal{P}$, we will use $\rho_{i,j}$ to denote the length of the shortest path from node $i \in \mathcal{P}$ to node $j \in \mathcal{P}$. Further, we will denote by $\mathit{\Gamma}_q(u)$ the set of all nodes (not including u) that are at forward distance $\leq q$ from node $u \in \mathcal{P}$, and we will define $\gamma_q(u) = |\Gamma_q(u)|$. Hence, $\Gamma_1(u)$ denotes the set of all outgoing neighbors of u. Analogously, we will denote by $\Gamma_q^{\text{in}}(u)$ the set of all nodes $v \in \mathcal{P}$ such that $u \in \Gamma_q(v)$. Hence, $\Gamma_1^{\text{in}}(u)$ denotes the set of all incoming neighbors of u.

2.1 Adversarial Model

Next, we turn to defining the communication and adversary model for adaptively secure computation with communication locality. We note that most works in this area leave several of the model assumptions implicit or unspecified. Instead, since our goal is to provide a complete feasibility landscape given such assumptions, we need to take a more rigorous and detailed approach to the specification of the model.

Consistently with the classical MPC literature, we assume that parties are connected with each other via a complete network of secure (i.e., authenticated and private) point-to-point channels [GMW87b,BGW88b]. However, since each party can "talk" to only a sublinear (i.e., polylogarithmic in n) number of other parties, no party will be using all its point-to-point channels. The communication is synchronous, which means all parties advance in a round-based manner, where whenever the round switches everyone is informed, and messages sent in any round r are guaranteed to be delivered by the beginning of the following round $r + 1$—unless the sender gets corrupted during r and no *multi-send* capability is assumed (see below). We will consider an adaptive and rushing adversary who might actively corrupt parties during the protocol execution.

The combination of sublinear communication locality and synchrony with such an adversary brings up a number of modeling challenges, described below.

Localized notification. In classical synchronous point-to-point networks, the adversary is always informed when a party p_i sends a message to another party p_j via their direct point-to-point channel. Notifying the adversary about such a transmission implicitly captures the assumption that the adversary has a **global** view of the entire network, including runtime-observable events, such as messages being transmitted. This gives him the ability to induce a worst-case (arbitrary/adversarial) scheduling of messages that makes for stronger security statements.

However, when we shift to settings with vastly large sets of parties—these are the settings where sublinear-locality protocols become relevant—the assumption that the adversary has a complete view of all the events that occur in the network might be too strong. In fact, it is impossible to achieve adaptive security with sublinear locality in this worst-case setting. Indeed, in low-locality settings[7], a party might only communicate with a small number of its neighbors. Hence, if the adversary is able to detect that an honest party p attempts to send a message to another honest party, then he can simply corrupt the receiver and block this transmission path; by performing this attack on every transmission of p the adversary will be able to isolate the party from the rest of the network, making *all-to-all* communication (and therefore "full" MPC) impossible[8]. Hence, for such settings, it is natural and relevant to limit the visibility to the adversary in message transmission events to only events happening near his neighborhood, i.e., assume that the adversary only observes message transmissions on channels that are incident to a neighborhood where he is present—e.g., channels in the immediate neighborhoods of parties currently under his control. In our model, we make this assumption—which is necessary for sublinear-locality communication and MPC, and therefore implicit in all relevant works—explicit by assuming that the adversary is only able to observe a transmission when the sender or the receiver of that transmission is corrupted.

Adaptive adversarial scheduling. The above localized notification assumption is natural in large networks (and necessary for adaptive corruption), but it does create a challenge with allowing the adversary to perform worst-case scheduling: Since the adversary does not have a full view of which honest-to-honest channels are used in each round (recall that only a small, sublinear per party, number is utilized), how can he

[7] Here "low" specifically means asymptotically smaller than the adversary's corruption budget.

[8] Here, we use "full" MPC to refer to the standard MPC formulation where no party is left out; this is in contrast to "almost-everywhere" MPC [GO08], where some of the parties are not given any correctness and privacy guarantees.

induce a worst-case scheduling? In fact, although the above localized-notification assumption was implicit in [CCG+15], this delicate issue was not addressed.

To address it, we look back at the classical way of defining scheduling for an adaptive and rushing adversary in [Can00]: Every round is split in "mini-rounds", where in each mini-round the adversary can have one party p_i send his message to another party p_j. If in the protocol every party communicates with all other parties in each round (a common protocol structure in feasibility results) then this means that each round has n^2 mini-rounds. The adaptive adversary is able to corrupt parties between any two mini-rounds. Note that, as explicitly discussed in [Can00], this means that a worst-case adversary would first deliver all messages sent to corrupted receivers and then schedule the remaining messages in an adaptive manner. To allow for worst-case scheduling in our model, we rely on the exact same idea: In each round r, the adversary operates in n^2 mini-rounds, where each such mini-round corresponds to a unique (ordered) pair of parties (p_i, p_j), allowing p_i to send its rth round message to p_j. The only difference here is that if and while both p_i and p_j are honest, the adversary does not learn whether or not a message was sent on the (p_i, p_j) channel in that round. (Of course, if either p_i or p_j gets corrupted down the line, then the adversary will find out at the point of corruption whether a message was exchanged in round r, unless the corrupted party has had a chance to erase before becoming corrupted—see discussion about erasures below.)

Trusted setup assumptions. We assume classical correlated randomness setups: The parties have access to a public-key infrastructure (PKI), which they can use for digital signatures and public-key encryption. In addition, the parties are given an appropriately sampled hidden random graph setup with polylogarithmic degree [CCG+15]. As discussed in [CCG+15], and already mentioned earlier, under standard hardness assumptions (i.e., existence of pseudo-random generators) this graph setup can be replaced by a different correlated randomness setup, namely, a secret key infrastructure (SKI); alternatively, if the PKI allows for non-interactive key exchange (NIKE), then any other setup assumption is not needed since an SKI can be created by pairwise invocation of the NIKE protocol—since this is a non-interactive process, it does not result in an increase in communication locality.

Secure erasures. Secure erasures are common (and, in fact, necessary in SF protocols) for adaptively secure MPC and, as we prove here, for point-to-point communication over a sublinear-degree network graph. The assumption is that (honest) parties are able to erase any part of their internal state (including parts of their setup and/or randomness) so that if they get corrupted later on, the adversary does not have access to the erased information. We note that in a model where such erasures are possible, such actions take place responding to protocol instructions, and therefore the adversary corrupting a party is allowed to learn that an erasure was performed by the party in the past.

2.2 Atomicity Assumptions

Atomicity of actions. One of the most important parameters of any adaptive adversary setting, which is often left as implicit, is the question of atomicity of operations for the honest party. A block of operations is considered atomic if, once an honest party starts performing them, it is allowed to complete them before the adversary gets a chance to act (e.g., perform additional corruptions). Clearly, the higher the number of operations that are bundled in an atomic block, the harder the job of the adversary becomes. One of the standard uses of atomicity in the distributed computing and cryptographic protocols literature is the so-called *atomic multi-send*, where if in a given round a party is supposed to send a message to multiple parties, it is allowed to do so without any in-between adversarial interference. One can view this assumption as a way to restrict the rushing ability of the adaptive adversary. As such, in the recent literature the (setting of an) adaptive adversary over a network of standard (point-to-point) channels (i.e., with non-atomic multisend) is at times referred to as *strongly rushing* [ACD+19,WXDS20] and the term "rushing" is used to refer to the setting where the adaptive adversary operates over a network with atomic-multisend channels.

In this work we use the term "rushing" consistently with [Can00,Can01,HZ10,GKKZ11] to characterize the adversary (i.e., its ability to schedule the delivery honest parties' messages) rather than the setting of network and adversary. Hence, atomic multisend is a network/protocol-related atomicity assumption: When the protocol instructs a party p to deliver a message to several parties in some set $\mathcal{Q}$ *in the same round* (or even a vector $(m_{p_1}, \ldots, m_{p_{\mathcal{Q}}})$ of messages, where each m_{p_i} is to be sent to $p_i \in \mathcal{Q}$), then the party can send all those messages as an atomic operation, meaning that these messages will be delivered to their intended recipients at the beginning of the following round; in particular, the adversary cannot corrupt p in the middle of the transmission and enforce that some $p_i \in \mathcal{Q}$ receives the message m_{p_i} from p while some other $p_j \in \mathcal{Q}$ does not.

Atomicity assumptions used in this work. The way different feasibility assumptions are bound together in an atomic operation has an impact on the feasibility of reliable communication (and therefore MPC) that one is able to prove. As our goal is to investigate the different relevant assumptions for sublinear- (polylogarithmic-) locality MPC, we next include a detailed discussion on the ways these assumptions can be bound (and have been bound in the prior literature):

- *No erasures/no (atomic) multisend (NE-NAMS):* This is the worst-case network and erasures model (fourth row of Table 1): (1) No honest party is allowed to erase its internal state and the adversary, upon corrupting a party, learns that party's entire prior and current state, and (2) the parties send their messages one by one as discussed in the adaptive scheduling paragraph above.
- *No erasures/(atomic) multisend (NE-AMS):* This corresponds to the second row of Table 1): (1) (Lack of) erasures are as above, and (2) the parties can atomically multisend their messages.
- *Erasures/no (atomic) multisend (E-NAMS):* This corresponds to the third row of Table 1. We assume erasures but no atomic multisend. Hence: (1) In each mini-round where a party p_i is allowed to "speak" (we say the party is *activated*); i.e., in each miniround corresponding to a pair (p_i, p_j) for some p_j, the party p_i can first erase and then send, but it cannot erase the message it is about to send (including the identity of the receiver) until the next mini-round when the party is activated), and (2) messages are sent in a one-by-one manner (in mini-rounds as above), where between any two mini-rounds the adversary can act.
- *Erasures/(atomic) multisend (E-AMS):* (1) Erasures are as in the previous case, and (2) when a party is activated for sending, it is allowed to erase and then send to a set $\mathcal{Q}$ of parties (but as above, it cannot erase this set or the messages it sends until the next time it is activated). Note that this case allows an adversary who has corrupted a party in $\mathcal{Q}$ to learn the message, corrupt the sender, learn the identities of all other parties in $\mathcal{Q}$ (since the sender is not given a chance to erase before) and corrupt all of these parties thereby blocking this message.
- *Atomic erasures* **and** *multisend (AE-AMS):* This is the strongest of the atomicity assumpions (first row of Table 1) and corresponds to the model considered in past works on sublinear locality MPC with adaptive corruptions (e.g., [CCG+15,BCDH18]). In this case, whenever a party is activated for sending a message, it is allowed to send to a set $\mathcal{Q}$ of parties (where all are guaranteed to receive their messages at the beginning of the following round) *and* perform erasures after sending is complete and before the adversary has a chance to corrupt this party. This, in particular, means that even if the adversary controls one of the parties in $\mathcal{Q}$, he is still unable to learn who the other parties in $\mathcal{Q}$ are even by corrupting the sender (as before corruption the sender is able to erase their identities).

3 Technical Overview

Impossibility of store-and-forward without erasures. Our first technical contribution (see Section 4) is an impossibility result for all-to-all (in fact, even one-to-all) store-and-forward RMT with a high expansion rate, if we do not assume erasures. In particular, we show this impossibility for expansion rate $(\log^z(n), \frac{k \log n}{c \log \log n})$, for all $z > 1$ and $k < 1$, where the degree of the communication graph is $O(\log^c n)$. Intuitively, our definition of (L, ℓ) expansion rate (see Definition 2) captures the constraint that, in an honest execution of the protocol, when the sender's message reaches parties up to a distance ℓ from himself, his message also reaches at least L parties through each of his neighbors. To our knowledge, this property can be shown to hold for all CL MPC protocols in the current literature. The result holds independently of whether or not we assume atomic multisend. The proof utilizes a combination of graph-theoretic and protocol results and can be summarized as follows: The first step in the proof shows that, due to the polylogarithmic locality assumption, there must exist a pair (p_s, p_r) of sender and receiver that are far enough from one another, concretely, in distance greater than $\frac{k \log n}{c \log\log n}$ (Lemma 2). Looking ahead, this will be the pair the adversary will try to disconnect. The expansion rate assumption will then ensure that for each forward neighbor p of p_s in the RMT to p_r, the graph rooted at p that is created by coloring the communication graph as the message of p_s (intended for p_r) passes though the intermediate nodes, will have sufficiently many colored nodes before it reaches p_r, so that with high probability one of them can be adversarial. Once this happens, the adversary will be able to follow the thread backwards all the way to this neighbor p and then corrupt p and all parties that p has sent the message to, thereby eliminating p as a possible relayer. Since each such p can be eliminated with high probability, and p_s had at most polylogarithmically many neighbors, the above adversary will be able to capture all these neighbors (and the colored graphs rooted at them) before the message reaches p_r (and without corrupting p_s). This adversary can drop all the message passing from captured nodes, thereby disconnecting p_s and p_r and violating the security of the RMT. The details of this proof are given in Section 4.

We remark that, although the above SF subclass might appear somewhat simple, it captures the structure of several natural message-propagation and gossiping protocols—in particular those used in the context of blockchains, as well as in so-called *store-and-forward* (switching) networks (cf. [BAB+01]). In fact, as demonstrated in Lemma 1, the class of SF RMT protocols with expansion-rate parameter that fall within our impossibility range includes all known message-propagation protocols in the sublinear locality MPC realm. We stress that our impossibility is not intended as a way to tightly characterize the feasibility landscape of CL RMT in the non-erasure setting, but rather to abstract the core of the above protocols that makes them inadequate against adaptive adversaries in this setting. Notwithstanding, we are not aware of any technique that yields a protocol in the non-erasure setting, as even common approaches for anonymous communication, e.g., onion-routing-based protocols, seem insufficient in this highly adversarial setting (as discussed in Section 1.4).

The power of secure erasures for CL protocols. As discussed above, the above impossibility (for SF RMT) protocols renders existing communication protocols in the CL MPC (and blockchain-via-gossip) literature insecure when erasures are not assumed. Continuing our exploration of the landscape, we turn to the question of how far can the secure-erasures assumption take us in terms of feasibility of RMT. Recall that [CCG+15] proved that if erasures and multisend can be performed/bundled in an atomic operation, then any adversary corrupting any constant fraction of the parties can be tolerated in RMT. Here we ask what happens if we unbounded these two assumptions, i.e., assume either only erasures, or erasures and multi-send as separate operations. And we shoot for the strongest possible results: (1) Feasibility even without atomic multisend and (2) Impossibility even with atomic multisend.

For the first (feasibility) result, our starting point is the RMT protocol from [CCG+15], which at a high level operates as follows: The sender sends his signed inputs to his hidden graph Round-1 neighbors,[9] and whenever a party receives a message in some round rnd, he relays it (in the next round, $rnd + 1$) to his $(rnd + 1)$-round hidden-graph (forward) neighbors[10].

The above protocol does not work here, since the communicating parties can be cut off by the attack described in Section 1.2; namely, a corrupted node p who receives a message from an honest neighbor q can corrupt q and all of q's neighbors before any chance of erasure. To make the above protocol secure when (just) erasures are assumed, we make the following modification: Every round is assigned to exactly one party in $\mathcal{P}$, who, if he has a message to send, sends this message to exactly one of his hidden graph (forward) neighbors at a time, and then (in the next activation/round) erases both the fact that he sent it and the relevant (used) edge from his hidden graph setup (i.e., the edge pointing to the neighbor he just contacted). Importantly, unlike [CCG+15], where as soon as a party relays a message he does not need to do any more relaying, we require every party that has received a message to keep relaying it to its hidden graph(s) neighbors. By

[9] Since we will be running at most polylogarithmic-round protocols, we can assume wlog that the hidden graph is a multi-graph consisting of polylogarithmic, independent copies of polylogarithmic-degree hidden graph setup.

[10] Recall that the hidden graph is directed.

forwarding messages to neighbors one at a time and erasing in between, the adversary is prevented from corrupting the neighbors of honest relayers who get corrupted, enabling a message to propagate into the network even after a past relayer is compromised. Note that to allow for worst-case attacks, one needs to devise a careful structure that gives the adversary sufficient attack-opportunities. We ensure this by forcing on our protocol a structure that makes erasures slow enough, so that the adversary is given enough time to attack (see Section 5.1).

The above protocol is clearly SF, and it even has an expansion rate that matches the parameter of our non-erasure impossibility theorem. Hence, one might be tempted to believe that if the adversary corrupts a constant fraction of the parties, then with good probability he will be able to block the message in a similar way as in our no-erasures impossibility proof. However, contrary to the above intuition, we show that any adversary corrupting at most $t < (1/2 - \epsilon)n$ of the parties can be tolerated (with overwhelming probability). The proof follows a careful probabilistic argument: Since, from a Chernoff bound, more than half of the hidden graph neighbors of any party are honest, we can show that the above protocol will create an avalanche effect: With good probability, for several rounds from the start of the above protocol the set of honest parties that has received the message will keep growing and it will become large enough to be guaranteed to include a party who is one hop (in the hidden graph) from the RMT receiver. Once this happens, it is game over for the adversary since this party will relay the message to the RMT receiver in the following round.

The above result establishes (one-to-one) RMT assuming an honest majority and erasures only (Theorem 2) and as a corollary, (one-to-one) RMT assuming honest majority, erasures, and multisend, not necessarily bulked as a single atomic operation (Corollary 1). Furthermore, it can be trivially extended to the all-to-all RMT case (where every party wants to send a message reliably to every other party) by using batching techniques from [CCG$^+$15] (Corollary 2). This settles the feasibility question.

We next turn to impossibility. Here we use an argument that can be seen as mirroring the proof of Theorem 2: We prove that if the majority of the parties can be corrupted (in particular, $t > (1/2 + \epsilon)n$ for any constant ϵ) then in *any* RMT-protocol candidate, the adversary can with noticeable probability make the expansion of the set of parties that learn the sender's input shrink, resulting in the message dying in the network before it reaches the receiver. This yields a strong impossibility of RMT in the erasures model, which as we show holds even if we assume multisend (Theorem 3).

Beyond Store-and-Forward. Next, we turn our attention to the question of whether one can design RMT protocols in the non-erasure model, by devising non-SF protocols, thereby circumventing our impossibility. The answer to this question is far from simple, which further underpins the challenges that CL protocol design poses.

Our key idea is to hide the store-and-forward procedure under the hood of fully homomorphic encryption (FHE). This will hide the message and signature (and in particular, origin and path) for transmitted messages. But as as we shall see, this seemingly simple intuition needs several modification to work.

The protocol structure is similar to the above protocol (which assumed secure erasures), with the main difference being that the sender encrypts his original message and signature with the receiver's HE public key, and this ciphertext is what is diffused through the low locality network. In particular, instead of checking signatures in the clear, every party uses HE to check if any of the received ciphertexts is an encoding of a message signed by the sender. If so, it (the circuit evaluated by HE) outputs a new, rerandomized HE ciphertext that encrypts the sender's message and signature (if several such messages are received then use any of them); otherwise encrypt the all-zero message along with a default signature on it. We denote the above protocol by $\Pi^{\mathsf{RMT}}_{\mathsf{FHE}}$.

In order for the above approach to work we need the HE scheme to satisfy several properties that are common in the fully homomorphic encryption (FHE) literature, namely *compactness* and *malicious circuit-privacy* [Gen09,SV10,vGHV10]. Informally, compactness ensures that the ciphertext size depends only on the plaintext and the security parameter (in particular it does not grow when the $\mathtt{Eval}_{\mathtt{FHE}}$ operation is applied). On the other hand, malicious circuit privacy ensures that the ciphertext that is computed by $\mathtt{Eval}_{\mathtt{FHE}}$ leaks no information about the circuit that was homomorphically evaluated, even when the ciphertext on which $\mathtt{Eval}_{\mathtt{FHE}}$ is computed is maliciously formed. This last property will ensure that applying $\mathtt{Eval}_{\mathtt{FHE}}$ automatically rerandomizes the ciphertext. We note that both of these properties are common in standard FHE schemes (cf. [Gen09,SV10,vGHV10,OPP14]).

A caveat in the above idea is that it is known that adaptively secure FHE which satisfies compactness is impossible [KTZ13]. Nonetheless, as we show below, we can construct a compact adaptively secure homomorphic encryption scheme which allows for the homomorphic evaluation of the specific function needed by our RMT protocol. In particular the specific function we require does not actually modify the contents of the underlying message that was encrypted (but only checks validity of a signature "under-the-hood" and then retains or disregards the underlying message). We stress that existence of such a scheme does not contradict the impossibility result of Katz *et al.* [KTZ13], as the circuits we consider are not in the class considered there.

Next, we describe the idea to circumvent the above impossibility. We consider only the single-pair RMT setting, where one sender u wishes to send a message to one receiver v. In order to ensure that ciphertexts can be simulated without knowledge of the plaintext and, if, later on, the sender and/or receiver is corrupted, the simulator can present randomness matching this ciphertext, we use the following idea: The sender first encrypts the message m it wishes to send with the receiver's public key, using an adaptively secure encryption scheme (e.g., the non-committing encryption of Canetti *et al.* [CFGN96b]). (Looking ahead, this will allow a simulator to equivocate the message if needed.) Next, the sender signs the resulting ciphertext c, and encrypts the resulting pair (c, σ) using the receiver's public key for a (*statically secure*) FHE scheme. For brevity, we will refer to the above operation of encrypting with FHE an authenticated version of the adaptively encrypted plaintext as *adaptively authenticated homomorphic encryption* (aaHE for short). Then the sender propagates the aaHE ciphertext through the hidden communication graph, identically to the original protocol described in Section 5.1.

However, unlike the original protocol, parties that are at distance > 1 from the sender cannot tell if the aaHE ciphertext they receive is encrypting a valid message (i.e., one actually originating from the sender) or if it was generated by the adversary. Looking ahead, this property is the key part where our protocol deviates from the SF protocol structure; yet, parties need to decide what to relay. One solution would be for the intermediaries to propagate all the messages they receive. This, however, would result in an exponentially growing message and could compromise the security of the protocol, as the number of relayed messages would leak information about the position of the relayer in the hidden graph, information that could be used by the adversary.

This is where FHE comes to the rescue. Upon receiving several such ciphertexts, a relayer homomorphically evaluates the circuit which on input all the (plaintexts of the) received ciphertexts and the sender's verification key, checks if any of these plaintexts is of the form (c, σ), where σ is a valid sender's signature on c, and if it finds one it outputs (an FHE ciphertext $\tilde{c}$ encrypting) it. (If more than one such message is found then output the first one encountered in the above search.) This will make sure that every relayer sends out ciphertexts of the same length which encrypt either (c, σ), in case the sender was honest and the relayer is on an honest path between the sender and the receiver, or some arbitrary pair $(*, *)$ (chosen by the adversary) otherwise. In other words, the above scheme ensures that the information transmitted by the above scheme is exactly an aaHE encryption of the message m which our original SF protocol from Section 5.1 would propagate, except that since this information is encrypted, the adversary cannot use it to trace the message/aaHE ciphertext back to the sender.

We stress that the properties of compactness as well as malicious circuit privacy, of FHE play a crucial role here:

- Compactness ensures that all communicated ciphertexts have the same size, and therefore their size conveys no information about the position of the relayer in the hidden communication graph.
- Malicious circuit privacy ensures that when the FHE circuit evaluation algorithm, $\mathtt{Eval}_{\mathtt{FHE}}$, is given input an invalid (i.e., maliciously generated) ciphertext it first projects it to a valid ciphertext of a (potentially adversarially chosen) plaintext x, and then performs the evaluation on this x. Circuit privacy ensures that any two relayed messages have computationally indistinguishable distributions (i.e., that $\mathtt{Eval}_{\mathtt{FHE}}$ rerandomizes the aaHE ciphertext), and hence, they can also not be used to expose relayers' position in the hidden graph. Further, when the sender is honest, it is impossible for the adversary to make a

relayer forward an aaHE ciphertext of (c, σ) of a message $m' \neq m$, as this would correspond to forging a signature σ' on c' (encryption of m') $\neq c$ (encryption of m) except with negligible probability.

However, there is still a way the adversary can obtain information in the above scheme that allows him to potentially link the sender to the transmitted aaHE ciphertext, by observing which parties "speak" in which round. This can easily be mitigated by decoy traffic: every party sends some message in every round, where in the first round, parties other than the actual sender create and send to their neighbors an aaHE ciphertext using dummy strings d and s in place of m and σ respectively, with $|d| = |m|$ and $|s| = |\sigma|$.[11]

The final missing piece in the protocol is to ensure that we can use the above protocol for transmitting multiple messages (as is needed in MPC). It is not hard to see that for this to be possible, we need to exclude replay attacks. Indeed, although the adversary cannot create an aaHE of a new message corresponding to honest senders, he can replay past aaHE of messages created by the sender. The way to mitigate this is, as it is common in the security literature, to use unique publicly agreed identifiers—e.g., round-number and/or message ID—and make sure that the circuit which $\mathtt{Enc}_{\mathtt{FHE}}$ is run on, also takes as input (and checks) the corresponding identifier. Yet, one needs to be careful about where the identifier is placed inside the aaHE. If one encrypts $(m, \mathsf{msg\_ID})$ with aaHE then $\mathtt{Eval}_{\mathtt{FHE}}$ will be unable to check the message ID $\mathsf{msg\_ID}$ under the (adaptively secure) encryption. Therefore, the actual message which is encrypted with FHE will be of the form $((c, \mathsf{msg\_ID}), \sigma)$, where σ is a signature on the pair $(c, \mathsf{msg\_ID})$.

This completes the high-level protocol description. In Theorem 4 we prove that the above protocol is an adaptively secure single-pair RMT protocol with polylogarithmic locality. Intuitively, the fact that the message is transmitted successfully between two honest parties follows from the fact that the protocol view of the adversary in this case is fully simulatable, hence any attack by an adaptive adversary who does not corrupt the sender or receiver can be reduced to the static case proven in [CCG$^+$15]. If, on the other hand, the adversary corrupts the sender or the receiver, then the only challenge for the simulator is to be able to come up with coins that are consistent with the actual input m of the sender. But this is straightforward in aaHE as the FHE ciphertext (i.e., encrypting $((c, \mathsf{msg\_ID}), \sigma)$) is generated by the simulator and hence he can simply reveal the keys used for this encryption. Having these coins, the simulator needs to simply show how to open c to the message m. But for that, he can use the adaptive security of the underlying encryption scheme. The detailed description of the protocol now follows, followed by Theorem 4. The proof of the theorem follows the above arguments and can be found in the supplementary material.

From one-to-one CL RMT to many-to-many CL RMT. The FHE-based protocol described above is for one-to-one RMT. One might might be tempted to assume that having such a protocol gives us also all-to-all RMT in this model. Indeed, at first thought, it appears that one can achieve this by simply running $\frac{n(n-1)}{2}$ instances of $\Pi^{\mathsf{RMT}}_{\mathsf{FHE}}$ in parallel over the same hidden graph (i.e., a joint state) on separate slots, one for each sender-receiver pair, following the idea of Section 5.1. However, the following major flaw breaks such a protocol.

Let u and v be honest parties in the network, and z be a party corrupted by adaptive adversary $\mathcal{A}$, with u being a sender while v and z act as receivers. Further, suppose the shortest honest path from u to z is shorter than that from u to v; i.e., z receives her message from u before (in an earlier round than) v. Now, adversary $\mathcal{A}$ can decrypt z's message in that same round and learn that it is a valid message originating from u. Moreover, $\mathcal{A}$ can corrupt the neighbor who sent her this valid message and decrypt that neighbor's previously received set of messages for z. In this manner, $\mathcal{A}$ can corrupt everyone who this message has passed through by following the inverse path the message travelled. Clearly, this is identical to the adversarial strategy employed in Section 4 to prove the impossibility of SF RMT in the $\mathsf{NE\text{-}NAMS}$ model, and it is easy to see that the same reasoning breaks the protocol here.

Intuitively, the root of the problem is that using FHE under the receiver's key renders messages "unlinkable" for everyone *but* the receiver. In single-pair RMT, there is a single receiver and correctness is trivial if he is corrupted, so $\Pi^{\mathsf{RMT}}_{\mathsf{FHE}}$ suffices. But with multiple receivers, corrupted receivers can evidently cause trouble for honest ones.

[11] WLOG, we assume that valid RMT messages are from a fixed-size domain.

The above discussion, points to a restricted class of all-to-all RMT, which we call *sublinear-output-set RMT* (in short, SOS-RMT) to come to the rescue. SOS RMT allows every party (a sender) to send a (potentially different) message to every receiver *in a subset of $\mathcal{P}$ of size $o(n)$*—in our case this will be of polylogarithmic size. Here is how we proceed towards the design such an SOS RMT:

1. First we observe that the above one-to-one RMT can be trivially turned into an all-to-on RMT, by having honest parties replace their decoy messages with the actual message they want to send to the (single) receiver and adjusting the homomorphic operation to keep (one copy of) all these messages as the ciphertext is diffused through the network. We provide the specification of this operation and the corresponding statement of security in Section 6.2. Note that this protocol uses exactly the same edges of the underlying communication graph as the one-to-one RMT protocol it "piggybacks" on.
2. Having such an all-to-one RMT it is straightforward to turn it to an SOS RMT by using an independent (part of the) hidden-graph setup for each of the receivers. Since there are polylogarithmic receivers and each of these hidden (sub-)graphs has polylogarithmic degree, the resulting protocol will also have polylogarithmic locality (see Corollary 4).

From CL RMT to CL MPC. Last but not least, we show how to turn the above feasibility results on RMT into feasibility for CL multi-party computation (CL MPC). For the erasures case, we can use the same approach as the one used in [CCG$^+$15] for this reduction: Use a constant-round MPC, e.g., [BMR90], where calls to a broadcast channels are replaced by a polylogarithmic-round (in the worst-case) byzantine broadcast protocol. Because the expected constant-round protocol of [KK06] is guaranteed to terminate with overwhelming probability after polylogarithmically many round, we can simply employ this protocol. This will result in polylogarithmically many invocations of the all-to-all RM from Section 5.1, which consumes a polylogarithmic hidden graph setup, thereby yielding a CL MPC protocol in the secure-erasures model (with or without atomic multisend), which is secure under an honest-majority (adaptive) adversary. We refer to Section 7.1 for details.

The more challenging case is the non-erasures setting. Here, we do not have an all-to-all CL RMT, so we cannot hope for standard CL MPC—as the latter would imply the former. Instead, we go for (CL) SOS MPC, which as with SOS RMT, computes a function with inputs from all parties, but only distributes the output to a sublinear (polylogarithmic) set of parties. We believe that the notion of SOS MPC is interesting in is own accord, as it appears to be a best-possible security notion in the CP non-erasure setting. Furthermore this notion is already reasonable for the core application of CL MPC, namely computing sublinear algorithms. Indeed, such algorithms typically have output asymptotically smaller than n. In this case, having the output-set of SOS MPC distribute the otputs to the whole player set (using the complete graph) does not incur a big overhead in communication complexity.

To implement CL SOS MPC, we assume an additional setup, namely, anonymous PKIs for the FHE, NCE, and signature schemes used in our non-erasures RMT protocol: Parties are given public keys but they do not know who has the corresponding secret key. Given this setup and SOS RMT, SOS MPC can be designed as follows: Let C denote a polylogarithmic size subset of the (owners of the secret keys for) the anonymous public keys. (Any subset will do, but for simplicity we can assume that this is the first polylog(n) public key in a lexicographic order). The parties use SOS RMT (where the FHE and NCE encryptions, and the underlying signatures are are with the anonymous PKIs) to share their input to C. Then the parties in C run an MPC over SOS RMT (again with these anonymous keys). However, to avoid leaking their identities through communication pattern, *all* n parties participate in these RMTs, where parties not in C simply send decoy traffic as in our one-to-one version of the non-erasure RMT protocol. The details on this construction and security proof are given in the end of Section 7.1.

4 Impossibility in the NE-NAMS and NE-AMS Models

In this section, we show an impossibility result for a natural class of reliable message transmission (RMT) protocols (and therefore also MPC) which we term store-and-forward protocols, with low communication locality in NE-NAMS and NE-AMS models—i.e., a model where secure erasures are not allowed—tolerating an adaptive adversary corrupting a linear number of parties. As discussed above, this class includes most if not all, gossip-style communication protocols which have been used in the CL MPC and CL communication literature.

Let us first define the class of store-and-forward RMT protocols:

Definition 1 (Store-and-Forward). *A PKI-hybrid RMT protocol with sender p_s and receiver p_r, using parties in $\mathcal{P}$ who communicate over point-to-point channels, is a* store-and-forward *(in short, SF) protocol if it has the following structure:*

- *In the first round of the protocol, p_s sends the message m (that he wishes to transmit to p_r) to any neighbors of his choosing, along with his signature σ_s on m. p_s does not participate in any other transmission.*
- *In every round $j > 1$, any party who has received a pair (m, σ_s), where σ_s is a valid signature (with respect to p_s's verification key) on m, may forward the pair (m, σ_s) to any neighbors of his choosing.*

We remark that the above definition allows parties to selectively decide (using their current state) when and to whom they forward the pair (m, σ_s). As such it also captures protocols that use delays to hide communication patterns. Furthermore, the assumption that in the first round, p_s sends all his protocol messages does not pose a restriction with respect to such scheduling—i.e., on when p_s sends his message to each neighbor—as such delays can be trivially simulated by p_s telling his neighbors (in the first round) to apply the intended delay. We also point out that the above single-pair RMT can be trivially extended to all-to-all RMT, by allowing each relayer p to forward vectors of pairs $((m_{i_1}, \sigma_{i_1}), \ldots, (m_{i_\ell}, \sigma_{i_\ell}))$, where each (m_{i_j}, σ_{i_j}) is a (message, p_{i_j}-signature)-pair that p heard in a previous round.

Our impossibility result assumes a restriction on the above class of SF protocols, which relates the maximum length of a path traversed by the sender's message to the size of the set of parties that have seen the message. To define this, we introduce the following graph theoretic notation.

Notation. We will denote by $G_{s,r}$ the *labeled* graph with vertices $V = \mathcal{P}$ which corresponds to a protocol's execution in the following manner: an edge (w_1, w_2) is added to $G_{s,r}$ when $w_1 \in V$ sends the message (m, σ_s) to $w_2 \in V$ through their point-to-point channel, in the RMT protocol between sender p_s and receiver p_r. The label l_{w_1,w_2} of each such edge (w_1, w_2) is defined as the round of the RMT protocol in which this edge was added to $G_{s,r}$. Further, we denote by $G_{s,r}^{\mathsf{rnd}}$ the subgraph of $G_{s,r}$ that only contains edges (w_1, w_2) having labels $l_{w_1,w_2} \leq \mathsf{rnd}$.

Definition 2 (Expansion rate of SF RMT protocols). *We say that an SF RMT protocol has expansion rate (L, ℓ), where $L \in [n]$ and $\ell \in \mathbb{N}$, if the following property holds at every round* rnd *in the protocol execution: If the maximum size of a path in $G_{s,r}^{\mathsf{rnd}}$ from p_s to a sink (i.e., a node which has out-degree 0 in $G_{s,r}^{\mathsf{rnd}}$) is ℓ, then for any (forward) neighbor p of p_s in $G_{s,r}^{\mathsf{rnd}}$, the number of nodes in the subgraph of $G_{s,r}^{\mathsf{rnd}}$ (with in-degree at least 1) rooted at p is at least L.*

The above definition can easily be extended to all-to-all SF RMT protocols (resp. one-to-all), by requiring that for every pair (p_s, p_r) (resp. for sender p_s and all receivers p_r) the expansion rate of the transmission is as in the above definition.

Looking ahead, we will prove our impossibility result for SF RMT protocols with expansion rate $(\log^z(n), \frac{k \log n}{c \log \log n})$ for all $k < 1$ and $z > 1$, where the degree of the communication graph is $O(\log^c n)$. To get a better intuition of how the expansion rate affects the security of RMT protocols against adaptive adversaries, it is worth looking at the simpler case with expansion rate $(d^{\xi-1}, \xi)$, where $d = O(\log^c n)$ is the degree of the underlying communication graph and ξ is a constant. We note that this seemingly simple case corresponds to (the first of ξ rounds of) the "vanilla" SF strategy which, to our knowledge, is employed by all CL MPC protocols in the literature.

Lemma 1. *Assuming no erasures, there exists no polylogarithmic-locality SF one-to-all RMT protocol with sender p_s that has expansion rate $((\log^c n)^2, 3))$ for some constant $c > 1$, and tolerates an adaptive adversary corrupting a constant fraction of the parties.*

Note that this is the case with $\xi = 3$. We next give a sketch of the proof of the above lemma. In fact, the above is just a special case of the general theorem (Theorem 1) proved later in this section. For that reason, we keep the proof at an informal level, to allow the reader to grasp the main ideas, and refer to the remainder of this section for formal claims (that even cover the more general case).

Proof (sketch). Assume the adversary corrupts $t = \tau n$ parties for some $\tau = O(1)$, and let π be a one-to-all SF RMT protocol with sender p_s and expansion rate as in the lemma. A crucial step in the proof is to argue that there exists a receiver $p_r \in \mathcal{P}$ such that the shortest path in $G_{s,r}$ from p_s to p_r is of length at least 3. This follows from the sublinear locality assumption of the communication graph. (In fact, in Lemma 2, we will argue that there is a receiver at distance greater than $q = \frac{k \log n}{c \log \log n} >> 3$.) In the following, we focus on the graph $G_{s,r}$ corresponding to the sender p_s and this far-away receiver p_r.

Let rnd be the round of π in which the (p_s, p_r)-RMT message (m, σ_s) first travels 3 hops away from p_s. As defined in Section 2, let $\mathit{\Gamma}_q(u)$ denote the set of nodes at forward distance $\leq q$ from any node u in $G_{s,r}^{\mathsf{rnd}}$. The expansion rate assumption implies that by the time (m, σ_s) travels 3 hops in the network, the number of nodes that have received it through each forward neighbor of p_s will be at least $\log^{2c} n$. As we show in (the general case) Lemma 3, with overwhelming probability, if the adversary corrupts at random a constant (say $\tau/4$) fraction of the parties, then for every (forward) neighbor p of p_s, there will be an adversarial node in the sub-graph of $G_{s,r}^{\mathsf{rnd}}$ rooted at p. In this case, the adversary realizes that p is an immediate neighbor of p_s and can (1) corrupt p, (2) corrupt all forward neighbors of p, and their forward neighbors too (i.e., corrupt all nodes in $\mathit{\Gamma}_2(p)$)—the adversary can do this as (a) there are no erasures which means that each party's state has his neighbor's identities, and (b) the adversary still has a linear corruption budget $3\tau/4$ unspent and there are at most polylogarithmic nodes in $\mathit{\Gamma}_2(p)$—, and (3) crash all corrupted parties, thereby blocking any transmission of (m, σ_s) that might have used p as a relay, as it has only traveled for 3 hops. Since the adversary has an overwhelming probability of succeeding in the above attack at each p_s-neighbor p, and there are at most $\mathsf{polylog}(n)$ such neighbors, the probability of this adversary corrupting all p_s-neighbors and preventing them from transmitting (m, σ_s) is noticeable. This contradicts the assumed security of π. □

The case of expansion rate $(\log^z n, \frac{k \log n}{c \log \log n})$**.** We assume that the adversary is able to corrupt a constant fraction τ of the nodes; hence $t = \tau n$. Our impossibility result holds for any constant $\tau > 0$. Our proof relies on a series of lemmas on the (polylogarithmic-degree) communication graph and how the adversary attacks any RMT protocol π over such a graph. Concretely, towards our impossibility result, we prove the following:

1. First, we will show (Lemma 2) that there exists a sender p_s and a receiver p_r, such that the length of the shortest path between p_s and p_r (in the communication graph $G_{s,r}$ of π) is strictly greater than q, where $q = \frac{k \log n}{c \log \log n}$, for any $k < 1$ and where the communication locality of the RMT protocol is $O(\log^c n)$. In other words, let rnd be the round of π in which the (p_s, p_r)-RMT message first reaches a distance q from p_s; then, p_r is not connected to any node in $G_{s,r}^{\mathsf{rnd}}$.

2. The remainder of our proof strategy is as follows: Consider an execution of RMT with sender p_s and receiver p_r as above; the communication graph after rnd rounds is $G_{s,r}^{\mathsf{rnd}}$. The goal of the adversary (and what we will prove he can achieve) is to corrupt each of the neighbors of p_s in this graph, and also corrupt everyone to whom they have (directly or indirectly) conveyed information on p_s's message, *before* this information reaches the receiver p_r. Thus, our proof focuses on each of the neighbors of s individually, and shows that the above is achieved with overwhelming probability. Using the fact that the total number of neighbors of p_s is $O(\log^c n)$, and by the choice of q, we can then prove that the probability of an adversary successfully attacking all of p_s's neighbors and cutting p_s off before (information on) his intended message reaches p_r is noticeable (Lemma 3). This forms a successful attack on the RMT protocol π, as it disconnects p_s and p_r, completing the proof.

The details of the proof now follow.

Lemma 2. *For any given SF RMT protocol π with communication locality $O(\log^c n)$, let $q = \frac{k \log n}{c \log \log n}$ for any $k < 1$. Consider a sender p_s. There exists a receiver p_r such that p_r is not connected to any node in $G_{s,r}^{\mathsf{rnd}}$, where* rnd *denotes the round of π in which the (p_s, p_r)-RMT message first reaches distance q from p_s.*

Proof. To show this, consider the union of all $G_{s,r}$ for all $p_r \in \mathcal{P}$; call this graph G_s. We know that the degree of every node in G_s is at most $d = O(\log^c n)$ (from the communication locality of the protocol). Now, in this graph, the number of nodes that are at distance $\leq q$ from p_s can be at most $d^q = O(n^k) < n$. This means there exists a node p_r that is at distance $> q$ from p_s in G_s, and thus in $G_{s,r}$. Now, consider the subgraph $G_{s,r}^{\mathsf{rnd}}$ for this p_r, where rnd is as defined in the lemma. Clearly, $G_{s,r}^{\mathsf{rnd}}$ only contains edges up to a distance q from p_s; thus, p_r is not connected to any node in $G_{s,r}^{\mathsf{rnd}}$. Hence the lemma follows. □

The most interesting step which captures the essence of our proof is Lemma 3 below. The intuition of the proof is that because the distance between p_s and p_r is larger than q, each p_s-neighbor in $G_{s,r}$ will have distance of at least q to p_r in $G_{s,r}$. Thus, any message originating from such a p_s-neighbor needs at least q hops to reach the receiver. Now consider the subgraph of $G_{s,r}$ which grows from a forward neighbor p of p_s only, i.e., the graph consisting of p, his neighbours, his neighbors' neighbors and so on. To prevent p_s from communicating with p_r via p, the adversary will first corrupt nodes at random with the hope of corrupting at least one node in this subgraph. More specifically, let us consider an adversary that initially corrupts a $\beta < \frac{\tau}{4}$ fraction of random nodes in the whole graph $G_{s,r}$, and show that such an adversary leaves both p_s and p_r initially uncorrupted—and looking ahead, will avoid corrupting p_s and p_r after this initial step. The first observation is that the message sent by p_s and relayed by p will, with overwhelming probability, hit some party in this initially corrupted set before it hits the receiver p_r. Once this happens, the adversary corrupts everyone who this message has passed through by following the inverse path the message travelled. This is feasible because nodes cannot erase any information, and in particular where messages came from and where they were relayed. If the adversary is able to do this for every p_s-neighbor, then he successfully cuts p_s off from p_r. To complete the proof we need to argue that this strategy can be launched within the adversary's corruption budget. Intuitively, this is the case, because in each step the total set of parties who have received information about the sender's message grows by a polylogarithmic factor. Hence, by the above choice of q we are guaranteed that the size of the set remains sublinear. Since the adversary has only spent a fraction of his linear budget in his initial corruption choice, he still has sufficiently many corruptions to perform the above attack. The formal statement and proof follow.

Lemma 3. *For any given SF RMT protocol π with communication locality $O(\log^c n)$, let $q = \frac{k \log n}{c \log\log n}$ for any $k < 1$. Further, let π have expansion rate $(\log^z n, q)$ for any $z > 1$. Consider a sender p_s and a receiver p_r, and an adversary $\mathcal{A}$ who corrupts each node in $\mathcal{P}$ at random with constant probability $\beta < \frac{\tau}{4}$. Then, with noticeable probability $\frac{1}{\mathsf{poly}(n)}$, $\mathcal{A}$ (i) does not corrupt more than $\frac{\tau n}{2}$ nodes in total; (ii) does not corrupt nodes p_s and p_r; but (iii) corrupts at least one node in $\{p\} \bigcup \Gamma_{(q-1)}(p)$ for every $p \in \Gamma_1(p_s)$.*

Proof. From the Chernoff bound, the probability q_1 with which $\mathcal{A}$ corrupts more than $\frac{\tau n}{2}$ nodes in G is $\leq e^{-\frac{\tau n}{12}}$. Next, the probability that p_s and p_r are not corrupted, q_2, is $(1 - \frac{\tau}{4})^2$. Now consider $\Gamma_{q-1}(p)$ for an arbitrary $p \in \Gamma_1(p_s)$. From the expansion rate assumption and Definition 2, we have $|\{p\} \bigcup \Gamma_{q-1}(p)| \geq \log^z n$.

Now, let q_3 be the probability that at least one node in $\{p\} \bigcup \Gamma_{q-1}(p)$ is corrupted for all $p \in \Gamma_1(p_s)$. Since the adversary corrupts nodes at random with probability β, the probability that no node is corrupted in $\{p\} \bigcup \Gamma_{(q-1)}(p)$ for a single p is $\tilde{q} = (1-\beta)^{\log^z n}$ (which is negligibly small), and hence $q_3 = (1-\tilde{q})^{\gamma_1(p_s)}$ is the probability that for all nodes $p \in \Gamma_1(p_s)$, at least one node in $\{p\} \bigcup \Gamma_{q-1}(p)$ is corrupted. Now, $\gamma_1(p_s) = O(\log^c n)$, due to communication locality. Therefore, with probability $(1-q_1)q_2 q_3 = \frac{1}{\mathsf{poly}(n)}$, we have that $\mathcal{A}$ does not corrupt more than $\frac{\tau n}{2}$ nodes in total, does not corrupt nodes p_s and p_r, but corrupts at least one node in $\{p\} \bigcup \Gamma_{q-1}(p)$ for every $p \in \Gamma_1(p_s)$. □

We now combine Lemmas 2 and 3 to obtain our impossibility result, in the following theorem.

Theorem 1. *In NE-NAMS and NE-AMS models (i.e. models not assuming erasures), there exists no SF protocol, with $(\log^z n, q)$ expansion-rate, for all-to-all RMT with polylogarithmic (i.e., $O(\log^c n)$) communication locality tolerating an adaptive adversary corrupting a linear number $t = \tau n$ (for any constant τ) of parties, where $q = \frac{k \log n}{c \log\log n}$, for any $k < 1$ and $z > 1$. The statement holds even assuming an arbitrary correlated randomness setup, atomic multisend, and any cryptographic hardness assumptions.*

Proof. Towards a contradiction, suppose there does exist such an SF RMT protocol with an expansion rate $(\log^z n, q)$; call this protocol π. From Lemma 2, we know there exists a sender $p_s \in \mathcal{P}$ and a receiver $p_r \in \mathcal{P}$, such that $\rho_{p_s,p_r} > q$ in $G_{s,r}$. We show that there exists an adversarial strategy that can disconnect p_s and p_r before they have exchanged any messages with each other, thereby violating the security of π.

The strategy is as follows: Let rnd be the round of π in which the (p_s, p_r)-RMT message first reaches distance q from p_s. At this point, $\mathcal{A}$ corrupts each node in $\mathcal{P}$ at random with constant probability $\beta < \frac{\tau}{4}$. For every $p \in \mathit{\Gamma}_1(p_s)$, let p^* be the node in $\mathit{\Gamma}_{(q-1)}(p)$ that is corrupted (such a node exists with noticeable probability by Lemma 3). Since no erasures are allowed, when $\mathcal{A}$ corrupts p^*, he also learns which nodes sent messages to p^* at any prior point of time in the protocol. Now, define S_{p^*} to be the set of nodes that sent messages to p^* at any time up to this point (pertaining to the RMT protocol between p_s and p_r). Next, $\mathcal{A}$ corrupts all nodes in S_{p^*}. Iteratively, consider all nodes $x \in S_{p^*}$ and for all such nodes x, consider the set S_x. The adversary can corrupt all nodes in S_x as well (since there are no erasures, the adversary learns the sets S_x for all x). Iteratively, the adversary corrupts all such nodes and finally also corrupts p. Now, $\mathcal{A}$ can move forward through the communication graph, and eventually corrupt all the nodes in $\mathit{\Gamma}_{q-1}(p)$, before the message can be transmitted outside of $\mathit{\Gamma}_q(p_s)$ by the honest parties.

What is left to be shown is the bound on the number of corrupted parties. The total nodes corrupted using this strategy is at most (using Lemma 3) $\frac{\tau n}{2} + \gamma_q(p_s)$. Now, this quantity is $\leq \tau n$; since $\gamma_q(p_s) \leq d^q = O(n^k) < \frac{\tau n}{2}$, where $d = O(\log^c n)$ is the degree of the communication graph, and $q = \frac{k \log n}{c \log \log n}$ for $k < 1$. Moreover, observe that all nodes in $\mathit{\Gamma}_q(p_s)$ who possessed p_s's message have been corrupted, and p_s has not communicated any message to any node outside of $\mathit{\Gamma}_q(p_s)$. We know from Lemma 2 that $p_r \notin \mathit{\Gamma}_q(p_s)$. Furthermore, since p_s has already communicated with all nodes in $\mathit{\Gamma}_1(p_s)$, it cannot communicate with any new neighbors. Hence, all messages that p_s sends henceforth are to corrupted nodes. Thus, $\mathcal{A}$ has successfully cut off transmission from p_s to p_r, violating the security of RMT protocol π.

Note that none of the arguments in the above proof rely on the absence of an atomic multisend capability; as such, this impossibility holds even if we assume atomic multisend. □

5 Positive Results in the E-NAMS/E-AMS Models

In this section, we assume that parties can erase their state. In our positive result, we do not assume that parties have an atomic multisend operation available to them, and the operations of sending a message and erasing state are not atomically bound either. This corresponds to the E-NAMS model. We will first show an all-to-all RMT protocol in this model with polylogarithmic locality tolerating $t < (\frac{1}{2} - \epsilon)n$ corruptions (Section 5.1); this automatically also implies a protocol in the stronger E-AMS model. To complement this result, we also show that tolerating $t > (\frac{1}{2} + \epsilon)n$ corruptions is impossible in the E-NAMS model (Section 5.2).

5.1 Polylogarithmic Locality RMT in the E-NAMS Model

Our protocol is a standard RMT protocol that allows a sender p_s to reliably transmit a message to a remote recipient p_r over a polylogarithmic degree graph. The all-to-all RMT is then obtained by having each pair use this RMT simultaneously. In addition to a PKI (for digital signatures) our protocol uses a hidden graph setup as in [CCG+15] as follows: the setup picks a directed random Erdős-Rényi graph $G(n, p) = (V, E)$, where $V = \mathcal{P}$ is the vertex set and E is the set of edges in G, and for every $i, j \in V$, $\Pr[(i, j) \in E] = p$. This graph is given to the parties such that every party learns its incoming and outgoing edges in G (and nothing else). From Section 2, recall that for any node $u \in V$, we denote its set of outgoing neighbors by $\mathit{\Gamma}_1(u)$. The set of nodes at distance $\leq i$ on a forward (directed) path starting from a node u are denoted by $\mathit{\Gamma}_i(u)$.

5.1.1 Single-pair RMT protocol in the E-NAMS model. We now describe our RMT protocol from honest sender p_s to honest receiver p_r, where $p_s, p_r \in V$, denoted by $\mathit{\Pi}_{RMT}$. Our protocol assumes a PKI, a hidden graph setup, existentially unforgeable digital signatures (equivalently, one-way functions), worst-case secure erasures (as discussed in Section 2), and no atomic multisend. As a corollary of our statement, at the end of the section we prove that the protocol with a minor tweak works even if we assume atomic multisend instead of just point-to-point communication. (The latter corollary is not that surprising, as atomic multisend restricts the adversary's power, but needs to be nonetheless done with care to make sure the adversary cannot abuse it to discover more of the hidden graph.)

Protocol structure. To make the protocol and proof simplest, we will (implicitly) induce the following structure: The protocol advances in blocks of two sequential (mini-)rounds, where in the first of these two minirounds a specific sender gets a chance to send a message to a specific receiver, and in the second one that sender gets a chance to erase his state (e.g., the information of the previous receiver). These miniround-blocks are advanced in a round-robin fashion: the first n blocks of such minirounds are with sender p_1 and receiver each party p_j in the party set; the next n blocks of minirounds are for sender p_2 and receiver each party p_j in the party set, and so on; after n such sets of n blocks, the $(n+1)$st set of n blocks is again with sender p_1, etc. Thus, a sequence of $2n^2$ minirounds (where all parties have had a chance to send all the messages they have for any other parties) constitutes a round in the protocol. We induce the above structure as it makes the influence of an adaptive adversary clean, no matter what model one is used to. Recall that we consider an adversary $\mathcal{A}$ who can adaptively corrupt up to a τ fraction of nodes in the network. Hence, $\mathcal{A}$ is allowed to order the blocks of minirounds within a single round (worst-case adaptive scheduling), and he can corrupt any party in between any two minirounds. Observe that this protocol structure does not increase the CL of the protocol, as a party will utilize its associated minirounds if and only if it has something to send to the corresponding receiver.

More concretely, the protocol starts at round 0, and we call the entire block below a round in the protocol.

- For $\mathsf{spkr} = 1$ to n, do the following:
  - If Party p_{spkr} has a message to send to p_0 it will do so.
  - Party p_{spkr} is given a chance to erase.
  - If Party p_{spkr} has a message to send to p_1 it will do so.
  - Party p_{spkr} is given a chance to erase.

    $\vdots$
  - If Party p_{spkr} has a message to send to p_{n-1} it will do so.
  - Party p_{spkr} is given a chance to erase.

The protocol then proceeds to the next round, completing a total of R rounds. The protocol Π_{RMT} itself is defined as follows.

RMT protocol between p_s and p_r. Our protocol proceeds for a total of $R = \log^{\tilde{c}} n$ rounds, for some constant $\tilde{c} > 1$ (where rounds are defined as above). All the verification keys of all nodes (denoted vk_w for each party $w \in \mathcal{P}$, with corresponding signing key sk_w) are known to all parties.

1. First, p_s signs (m, p_r) with sk_{p_s}. Denote the signed message (which comprises of the (m, p_r) as well as the signature on it) by μ_m. Party p_s also initializes ctr_{p_s} to the index of a random neighbor in $\Gamma_1(p_s)$.
2. Now, at every round $0 \le j \le R$, every node w does the following:
   - w checks if he possesses a single valid message μ_m - i.e., a message of the form (m, p_r) that has been signed by p_s. If so, then w sends μ_m to $\Gamma_1(w)[\mathsf{ctr}_w]$ and sets $\mathsf{ctr}_w = \mathsf{ctr}_w + 1$. (This constitutes a mini-round, and is immediately followed by another mini-round in which w is given a chance to erase the information of the node in $\Gamma_1(w)[\mathsf{ctr}_w]$ he sent to in the previous miniround.) Node w repeats the above over d many neighbors in $\Gamma_1(w)$ by iteratively incrementing ctr_w, where $d = O(\log^c n)$ (for some $c > 1$) is the communication locality. Otherwise, if he possesses no valid message μ_m, then he does nothing.
   - w disregards all messages from $w^* \notin \Gamma_1^{\text{in}}(w)$.

We now prove that the above protocol is an RMT between p_s and p_r. Define set, GOOD_j, $0 \le j \le R$ to be the set of nodes, who at the beginning of round j of the protocol are a) honest and b) are in possession of the message μ_m. Nodes can be corrupted by an adversary adaptively at any point of time (note that nodes can get corrupted even within a round as defined above) e.g. after a node sends μ_m to an adversarial node, it can get corrupted. As long as some node in GOOD_j for some j sends μ_m to p_r, RMT is achieved between p_s and p_r. Also, note that until an honest node w sends a message to its next neighbor, $\Gamma_1(w)[\mathsf{ctr}_w]$ is a random node from the adversary's view. Hence, the probability that p_r receives the message μ_m is simply the probability that of all the random neighbors selected by nodes in GOOD_j, $0 \leq j \leq R$, at least one of these nodes is p_r. Let $g_j = |\mathsf{GOOD}_j|$ for all j.

The idea of our proof is that no matter what the adversary does, the set of parties that know the message (and will therefore forward it in the next round) grows multiplicatively in each round of the protocol. Hence, in $\log^{\tilde{c}} n$ rounds, the message will reach a large enough honest set, so that one of them will be a neighbor of p_r and will therefore forward the message to p_r. Once this happens, p_r will observe it (since the message carries the sender's signature) which means that the RMT will have succeeded. Details follow.

The first lemma below will be useful in arguing that the good set (of parties that have the message) starts off with a large enough size.

Lemma 4. *Let $W = \{w_1, \cdots, w_k\}$ be a set of nodes selected uniformly at random with replacement. Let $k \geq \log^c n$ for some $c > 1$. Then, for any constant $\delta < 1$ the number of unique nodes in W, i.e., $|W|$ is $\geq (1-\delta)k$, except with probability negligible in n.*

Proof. Consider the ordered list of nodes $w_1, \cdots, w_k$. Now, for any w_i, the probability with which w_i is a repeated node (i.e., there exists some w_j, $j < i$, s.t. $w_j = w_i$) is at most $\frac{k}{n}$. Then, the probability with which w_i is repeated for δk of the indices is bounded by $(\frac{k}{n})^{\delta k}$, which is negligible in n, when $k \geq \log^c n, c > 1$. Hence the lemma follows.

Using the above lemma we can prove that the set GOOD_1 of honest parties that have seen message μ_m at the beginning of round 1 has size polylogarithmic in n.

Lemma 5. *For corruption threshold τ and any $0 < \epsilon < 1 - \tau$, the size of the good set after round 0 is bounded as $g_1 \geq (1 - \tau - \epsilon)d$, except with probability negligible in n.*

Proof. This follows from a simple Chernoff bound argument. Let $\mathsf{ctr}_{p_s} = f$ initially. Then, $D_1^{p_s} = \{\Gamma_1(p_s)[f], \Gamma_1(p_s)[f+1], \ldots, \Gamma_1(p_s)[f+d]\}$ is a set of nodes chosen at random independently of everything else that the adversary $\mathcal{A}$ has seen so far in the network. Hence, the nodes in $D_1^{p_s}$ are each corrupt independently with probability τ. The probability with which the number of honest nodes in $D_1^{p_s}$ is $< (1 - \tau - \epsilon')d$ is the probability with which the number of adversarial nodes in $D_1^{p_s}$ is $\geq (\tau + \epsilon')d$, which by the Chernoff bound is at most $e^{\frac{-\epsilon' d}{3}}$, i.e., negligible in n as $d = \log^c n$, for some $c > 1$, by assumption. Finally, the number of unique nodes in this set will be $\geq (1 - \tau - \epsilon)d$ (by Lemma 4).

Next we show that as long as the adversary corrupts a minority of parties, in every round, the size of the good set increases multiplicatively by a constant greater than 1.

Lemma 6. *For any corruption threshold τ and any $0 < \epsilon < 1 - \tau$, the size of the good set after any round $0 < j \leq R$ is bounded as $g_{j+1} \geq 2(1 - \tau - \epsilon)g_j$, except with probability negligible in n.*

Proof. The proof is by induction on j. First, let us analyze the base case. We have $g_1 \geq (1 - \tau - \epsilon)d$ from Lemma 5. Now, in the next round, all nodes w in GOOD_1 will send μ_m to a random neighbor (namely $\Gamma_1(w)[\mathsf{ctr}_w]$). Now, for a node w, $w' = \Gamma_1(w)[\mathsf{ctr}_w]$ is honest with probability $(1 - \tau)$. In this case, we include both w and w' in GOOD_2. If for a node $w \in \mathsf{GOOD}_1$, $w' = \Gamma_1(w)[\mathsf{ctr}_w]$ is malicious, we include neither w nor w' in GOOD_2 (as w' could then also corrupt w). By Lemma 4, the expected size of GOOD_2 will be $2(1 - \tau)g_1$, and applying the Chernoff bound shows that the size of GOOD_2, i.e., g_2 will be $\geq 2(1 - \tau - \epsilon)g_1$ with overwhelming probability.

Now, for the inductive hypothesis, assume that $g_\ell \geq 2(1 - \tau - \epsilon)g_{\ell-1}$. Now, again for every node w in GOOD_ℓ, $w' = \Gamma_1(w)[\mathsf{ctr}_w]$ is honest with probability $(1-\tau)$ and in this case two nodes are added to $\mathsf{GOOD}_{\ell+1}$. Again, by the Chernoff argument, and Lemma 4, the size of $\mathsf{GOOD}_{\ell+1}$, i.e., $g_{\ell+1}$ will be $\geq 2(1 - \tau - \epsilon)g_\ell$.

Finally, we prove that the above constant expansion will ensure that in $\log^{\tilde{c}} n$ rounds (which is how long our protocol runs), the message will arrive at its intended receiver v.

Lemma 7. *For any $j \geq \log^{\tilde{c}} n$, if corruption threshold $\tau < \frac{1}{2} - \epsilon$ for any $0 < \epsilon < \frac{1}{2}$, then probability that $p_r \neq \Gamma_1(w)[\mathsf{ctr}_w]$ for any w in any good set GOOD_j is negligible.*

Proof. This is the probability that p_r was never randomly selected as $\Gamma_1(w)[\mathsf{ctr}_w]$ for any $w \in \mathsf{GOOD}_j$ for any j. Now, since $\tau < \frac{1}{2} - \epsilon$, we have $2(1 - \tau - \epsilon) > 1$ and hence, from Lemma 6, for $j = R = \log^{\tilde{c}} n$, GOOD_R is of size $g_R = \nu n^{\log^{\tilde{c}-1} n}$ for some $0 < \nu < 1$ and $\tilde{c} > 1$. This means then that the probability with which p_r was never randomly selected as $\Gamma_1(w)[\mathsf{ctr}_w]$ for any w in any GOOD_j is bounded by $(1 - \frac{1}{n})^{g_R}$, which is negligible in n.

This completes the proof of the following theorem.

Theorem 2. *Assuming a PKI, a hidden graph setup as above, secure erasures, one-way functions (for existentially unforgeable signatures) and an adaptive adversary corrupting at most $t < (\frac{1}{2} - \epsilon)n$ parties for any $0 < \epsilon < \frac{1}{2}$, the protocol Π_{RMT} realizes reliable message transmission from p_s to p_r. The statement holds in the $\mathsf{E\text{-}NAMS}$ as well as $\mathsf{E\text{-}AMS}$ models.*

Since atomic multisend is a stronger model that the non-atomic multisend setting considered above, the possibility results applies also to this case. Indeed, given atomic multisend one can trivially simulate point-to-point communication between a sender p_s and receiver p_r, by having p_s multisend the vector that includes the intended message to the location corresponding to p_r and 0 to all other parties in its outgoing neighborhood of the hidden graph setup. This proves the following statement about the protocol $\Pi_{RMT}^{(MS)}$ which results from instantiating Π_{RMT} by sending a message via the above invocation of the atomic multisend primitive.

Corollary 1. *Assuming a PKI, a hidden graph setup as above, secure erasures, one-way functions (for existentially unforgeable signatures), atomic multisend, and an adaptive adversary corrupting at most $t < (\frac{1}{2} - \epsilon)n$ parties for any $0 < \epsilon < \frac{1}{2}$, the protocol $\Pi_{RMT}^{(MS)}$ described above realizes reliable message transmission from p_s to p_r in the $\mathsf{E\text{-}AMS}$ model.*

5.1.2 All-pairs (aka all-to-all) RMT in the E-NAMS model. We now describe our protocol for RMT between all pairs of parties, denoted by Π_{a2aRMT}. This will allow every party u to send a message to every other party v in a total of R rounds (where a round is defined as earlier). At a high level, Π_{a2aRMT} works as follows. We will execute a total of $\frac{n(n-1)}{2}$ instances of protocol Π_{RMT} from Section 5.1 in parallel. For every receiver v, every sender u signs (m, v) with sk_u. Denote the signed message (which comprises of the (m, v) as well as its signature) by $\mu_{u,v}$. Every party w will maintain $\frac{n(n-1)}{2}$ slots, each corresponding to one (u, v) pair. Now, at every round $0 \leq j \leq R$, w checks if it possesses any valid message that has been sent by sender u to receiver v (i.e. a message $\mu_{u,v}$ of the form (m, v) that has been signed by u). It places this message in the slot corresponding to the pair (u, v). w then sends (potentially) all $\frac{n(n-1)}{2}$ messages to $\Gamma_1(w)[\mathsf{ctr}_w]$ and sets $\mathsf{ctr}_w = \mathsf{ctr}_w + 1$. It is easy to see that the communication locality of any party does not increase through this process – only the number of messages sent by a party at a time increases from a single message to a collection of $\frac{n(n-1)}{2}$ messages. Applying a union bound over all pairs of senders and receivers, one can obtain the following corollary to Theorem 2.

Corollary 2. *Assuming a PKI, a hidden graph setup as above, secure erasures, one-way functions (for existentially unforgeable signatures) and an adaptive adversary corrupting at most $t < (\frac{1}{2} - \epsilon)n$ parties for any $0 < \epsilon < \frac{1}{2}$, the protocol Π_{a2aRMT} realizes reliable message transmission between all pairs of parties $(u \in \mathcal{P}, v \in \mathcal{P})$. The statement holds in the $\mathsf{E\text{-}NAMS}$ as well as $\mathsf{E\text{-}AMS}$ models.*

5.2 Impossibility of Dishonest Majority in the E-NAMS model

In this section, we shall show that it is impossible to construct SF reliable message transmission (RMT) protocols (and therefore also MPC) with low communication locality, in a model even with erasures, if the corruption threshold is $\frac{1}{2}+\epsilon$ for any constant $\epsilon > 0$. To do so, we shall prove that an adversary can break correctness of any RMT protocol between some pair of honest nodes in any such protocol. The proof idea can be seen as symmetric to the proof of Theorem 2. In particular, in Theorem 2 we showed that if the adversary corrupts $t < (\frac{1}{2}-\epsilon)n$ parties, for any constant $0 < \epsilon < 1/2$, then the set of honest nodes that learns (and forwards) the sender's message grows exponentially fast, and therefore in $\log^{\tilde{c}} n$ rounds it will be large enough to hit a neighbor of the receiver. Here we prove that if $t \geq (\frac{1}{2}+\epsilon)n$ for any constant $0 < \epsilon < 1/2$, then there is a strategy making the above set shrink exponentially fast, which can make the message disappear before reaching the receiver p_r.

Consider an RMT protocol between an honest sender p_s and an honest receiver p_r. We shall show that for any SF RMT protocol from p_s to p_r, an adversary that corrupts $\frac{1}{2}+\epsilon$ parties can prevent the message m from reaching p_r in any polynomial number of rounds. As in Section 5.1, let the RMT protocol from p_s to p_r begin at round 0, and define GOOD_j, $0 \leq j \leq R$ to be the set of nodes, who at round j of the protocol are a) honest and b) are in possession of the message μ_m. Let $g_j = |\mathsf{GOOD}_j|$ for all j.

Adversarial strategy. Our adversarial strategy is as follows: First, corrupt nodes in the graph uniformly at random (i.e., every node is corrupted with probability $\frac{1}{2}+\frac{\epsilon}{4}$). Next, if an adversarial node receives a message (that was a part of the RMT protocol between p_s and p_r) from some node w (other than p_s), then corrupt w. Do not forward any messages.

Lemma 8. *Except with probability negligible in n, the adversary will corrupt at most $\frac{1}{2}+\epsilon$ fraction of nodes.*

Proof. First, from a simple Chernoff bound, the adversary corrupts at most $\frac{1}{2}+\frac{\epsilon}{2}$ when corrupting nodes at random. Next, for every node corrupted this way, the adversary at most corrupts one additional node (the node w which sent the message to it). Hence, at most $\frac{1}{2}+\epsilon$ nodes are corrupted in total.

Next, we show that after initial round, only a small set of honest nodes are in possession of the message.

Lemma 9. *Except with probability negligible in n, the size of the good set after round 0 is bounded as $g_1 \leq (\frac{1}{2}-\epsilon')d$, for some constant $\epsilon' > 0$ and communication locality d.*

Proof. The proof of this Lemma follows from the Chernoff bound in a very similar manner to the proof of Lemma 5.

Finally, we show that after every round in the protocol, the number of honest nodes in possession of the message reduces.

Lemma 10. *Except with probability negligible in n, the size of the good set after any round $j > 0$ is bounded as $g_{j+1} \leq (\frac{1}{2}-\epsilon')g_j$ for some constant $\epsilon' > 0$.*

Proof. The proof of this lemma follows in a similar manner to the proof of Lemma 6. This is because, whenever an honest node sends a message to an adversarial node (which happens with probability $\geq \frac{1}{2}+\epsilon''$), this honest node will be corrupted by the adversary, hence progressively decreasing the size of GOOD_j as j increases.

Putting the above together, it follows from Lemmas 8, 9,and 10, that our adversary corrupts $\frac{1}{2}+\epsilon$ parties and the RMT message from p_s to p_r fails to reach p_r with noticeable probability, which proves the following theorem.

Theorem 3. *In the E-NAMS model (i.e., without atomic multisend), there exists no all-to-all store-and-forward RMT protocol with polylogarithmic CL tolerating an adaptive adversary corrupting $t \geq (\frac{1}{2}+\epsilon)n$ (for any constant $0 < \epsilon < \frac{1}{2}$) parties. The statement holds even assuming an arbitrary correlated randomness setup, secure erasures, and any cryptographic hardness assumption.*

Remark 1. We note that the above argument holds if we do not assume atomic multisend (i.e., in $\mathsf{E\text{-}NAMS}$). Indeed, in the stronger $\mathsf{E\text{-}AMS}$ model, the nodes might be able to do some smart multisend-based relay that prevents the set of parties that know the message from shrinking, or slows down the rate. We leave this interesting question as a future research direction.

6 Polylogarithmic Locality RMT in the $\mathsf{NE\text{-}NAMS}$ Model

In this section, we propose RMT protocols which are not store-and-forward, and can therefore circumvent the impossibility result from Section 4. Our key idea is to remove the ability of the adversary to identify the sender by looking at intermediate messages, with the use of fully homomorphic encryption (FHE) to hide the contents (and in particular, origin and path) of transmitted messages. The resulting protocol for single-pair RMT is described in Section 6.1. However, we subsequently note that the same protocol loses its security when composed in parallel for the purpose of all-pairs RMT. In the following Section 6.2, we show how the protocol can be extended to obtain RMT from all senders to $\mathsf{polylog}(n)$ receivers, which we term *sublinear output-set* RMT (or SOS-RMT) and later use directly to achieve communication-local SOS-MPC in Section 7.2.

6.1 Single-Pair RMT using Fully Homomorphic Encryption

We next provide a description of our (one-to-one) RMT protocol in the non-erasure case under strong cryptographic assumptions. Here we denote the sender by u and the receiver by v.

Single-pair RMT protocol $\Pi_{\mathsf{FHE}}^{\mathsf{RMT}}$ between u and v from FHE and adaptively secure (non-committing) encryption. Similarly to the protocol from Section 5.1, our protocol proceeds for a total of $R = \log^{\tilde{c}} n$ rounds for any constant $\tilde{c} > 1$ (where rounds are as defined in Section 5.1). The protocol assumes setup for the following schemes:

- An existentially unforgeable digital signature scheme $(\mathtt{KeyGen}, \mathtt{Sign}, \mathtt{Verify})$. Denote by vk_u the verification key of the sender u and by sk_u the corresponding signing key.
- A non-committing encryption scheme $(\mathtt{KeyGen}_{\mathtt{NCE}}, \mathtt{Enc}_{\mathtt{NCE}}, \mathtt{Dec}_{\mathtt{NCE}})$. Denote by $\mathsf{pk}_v^{\mathtt{NCE}}$ and $\mathsf{sk}_v^{\mathtt{NCE}}$ the encryption and decryption keys of the receiver.
- A compact and malicious circuit-private FHE scheme $(\mathtt{KeyGen}_{\mathtt{FHE}}, \mathtt{Enc}_{\mathtt{FHE}}, \mathtt{Eval}_{\mathtt{FHE}}, \mathtt{Dec}_{\mathtt{FHE}})$. Denote by $\mathsf{pk}_v^{\mathtt{FHE}}$ and $\mathsf{sk}_v^{\mathtt{FHE}}$ the encryption and decryption keys of the receiver, respectively.

The protocol also assumes that the parties have agreed on unique public message IDs $\mathsf{msg\_ID}$ for the transmitted messages (this will include the protocol ID, the party ID, and the current round). The protocol proceeds as follows:

1. Computation of each party when the protocols starts (to compute the first message they will send):
   - *Code for the sender u:* First, u encrypts m with v's (non-committing) encryption key $\mathsf{pk}_v^{\mathtt{NCE}}$; denote the resulting ciphertext by c. Then, u signs $(c, v, \mathsf{msg\_ID})$ with sk_u; denote the corresponding signature by σ. Finally, u encrypts the pair $((c, v, \mathsf{msg\_ID}), \sigma)$ with v's FHE encryption key $\mathsf{pk}_v^{\mathtt{FHE}}$; denote the resulting (aaHE) ciphertext by $\tilde{c}_u$.
   - *Code for each party $w \neq u$:* Party w computes c as an encryption of the all-zero message of size $|m|$ with v's (non-committing) encryption key $\mathsf{pk}_v^{\mathtt{NCE}}$ and sets σ to the all-zero string of same size as the actual signature of u above. Then, w encrypts $((c, v, \mathsf{msg\_ID}), \sigma)$ with v's FHE encryption key $\mathsf{pk}_v^{\mathtt{FHE}}$; denote the ciphertext by $\tilde{c}_w$.
2. Next, at any round $0 \leq j \leq R$, every node w does the following: Let $C_{w,j} = \{\tilde{y}_1, \ldots, \tilde{y}_q\}$ be the ciphertexts that party w has received in the previous rounds ($C_{w,j} = \{\tilde{c}_w\}$ if no messages have been received yet.)
   - w applies the homomorphic evaluation function $\mathtt{Eval}_{\mathtt{FHE}}$ on input the ciphertexts in $C_{w,j}$, the verification key vk_u of the sender u, and the pre-agreed message ID $\mathsf{msg\_ID}$ to compute the following function: If any $\tilde{y} \in C_{w,j}$ can be parsed as $((c, v, \mathsf{msg\_ID}), \sigma)$, where σ is a valid signature on

$(c, v, \mathsf{msg\_ID})$ according to the sender's verification key vk_u, then output $((c, v, \mathsf{msg\_ID}), \sigma)$. (If there are multiple such $\tilde{y}$, output the one with the smallest c.) Party w denotes the resulting FHE ciphertext by $\tilde{c}_{w,j}$, sends it to $\Gamma_1(w)[\mathsf{ctr}_w]$ and sets $\mathsf{ctr}_w = \mathsf{ctr}_w + 1$.
- w disregards all messages from $w^* \notin \Gamma_1^{\text{in}}(w)$.

3. At round R, v uses his FHE decryption key to decrypt each FHE ciphertext in $C_{v,R}$ (i.e., all ciphertexts received in the protocol). If any $\tilde{y} \in C_{v,R}$ can be parsed as $((c, v, \mathsf{msg\_ID}), \sigma)$, where σ is a valid signature on $(c, v, \mathsf{msg\_ID})$ according to the sender's verification key vk_u, then v uses his NCE decryption key $\mathsf{sk}_v^{\mathsf{NCE}}$ to decrypt c and outputs the corresponding message as the one sent by u (if more than one such c exists, then v takes the one corresponding to the smallest message m). Otherwise, v outputs 0 as the message received from u.

Theorem 4. *Assuming a PKI, a hidden graph setup, trapdoor permutations with a reversed domain sampler, and compact and malicious circuit-private FHE [OPP14], protocol $\Pi_{\mathsf{FHE}}^{\mathsf{RMT}}$ securely realizes single-pair RMT, tolerating an adaptive adversary who corrupts $t < \epsilon n$ parties for any constant $0 \leq \epsilon < 1$, in the $\mathsf{NE\text{-}NAMS}$ model.*

6.2 Multi-Sender RMT

While $\Pi_{\mathsf{FHE}}^{\mathsf{RMT}}$ cannot be composed in parallel to achieve all-pairs RMT as discussed in Section 3, we show in this section that simple joint state parallel composition of single-pair RMT is sufficient for all-to-one RMT, and the usage of multiple all-to-one RMT instances over independent hidden graphs can further extend this to SOS-RMT.

We note that each form of parallel composition in this section will require unique message identifiers (denoted by $\mathsf{msg\_ID}$) used in each constituent RMT instance. In more detail, unique message IDs must be used for each of the n instances of single-pair RMT that are composed into an all-to-one RMT, and similarly, distinct message IDs must be used in each of the $\mathsf{polylog}(n)$ all-to-one RMT instances that form SOS-RMT.

6.2.1 All-to-one RMT by parallel composition. We now define our protocol for RMT with a single receiver v, and all other parties acting as senders, denoted by $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$. At a high level, the protocol works as follows. We will execute $n-1$ instances of $\Pi_{\mathsf{FHE}}^{\mathsf{RMT}}$ in parallel over the same hidden graph, with each party maintaining one slot for each (u, v) pair, where u is any node except the receiver v. At the beginning, each sender $u \neq v$ creates an aaHE ciphertext μ_u for his own slot, along with dummy ciphertexts for all other slots. At every round $0 \leq j \leq R$, every party w runs $\mathtt{Eval}_{\mathtt{FHE}}$ on his collection of ciphertexts $C_{w,j}$ for each slot (u, v), before forwarding the results from all $n-1$ slots to a neighbor $\Gamma_1(w)[\mathsf{ctr}_w]$ and incrementing ctr_w. Finally, after round R, the receiver v decrypts the received aaHE ciphertexts from each of the $n-1$ slots using his FHE and NCE decryption keys, to obtain the messages from all $n-1$ senders. Observe that the flaw discussed in Section 6.1 does not apply here, as there is only one receiver. We thus have the following corollary, the proof of which can be found in the supplementary material.

Corollary 3. *Assuming a PKI, a hidden graph setup, trapdoor permutations with a reversed domain sampler, and compact and malicious circuit-private FHE, protocol $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$ securely realizes all-to-one RMT, tolerating an adaptive adversary who corrupts $t < \epsilon n$ parties for any constant $0 \leq \epsilon < 1$, in the $\mathsf{NE\text{-}NAMS}$ model.*

6.2.2 SOS-RMT by multiple hidden graphs. We now show a simple extension of $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$ to achieve RMT with all parties acting as senders, and some $\mathsf{polylog}(n)$ parties acting as receivers. We term this sublinear output-set RMT, or SOS-RMT. We denote our protocol by $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$. The idea is as follows. Instead of a single hidden graph, we provide the parties with $\mathsf{polylog}(n)$ independent hidden graphs as part of the setup. Observe that this can be realized by a single hidden graph setup of a higher-degree $\mathsf{polylog}(n)$, and thus it does not complicate the setup any further. With $\mathsf{polylog}(n)$ independent hidden graphs, $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$ simply requires the parties to perform one instance of all-to-one RMT (using $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$) for each receiver v on a unique hidden graph G_v. Even if some of the receivers are corrupted by the adversary, they clearly cannot block a sender from reaching the honest receivers, due to independence of edges between hidden graphs. The following corollary is immediate.

Corollary 4. *Assuming a PKI, a hidden graph setup, trapdoor permutations with a reversed domain sampler, and compact and malicious circuit-private FHE, protocol* $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$ *securely realizes SOS-RMT, tolerating an adaptive adversary who corrupts* $t < \epsilon n$ *parties for any constant* $0 \leq \epsilon < 1$*, in the* NE-NAMS *model.*

An important point to note is that the overall communication locality of each party remains $\mathsf{polylog}(n)$, since any party communicates with $\mathsf{polylog}(n)$ many neighbors in each of $\mathsf{polylog}(n)$ hidden graphs.

7 Communication-Local MPC

Having established the feasibility of RMT with polylogarithmic CL in all four models, we turn to the question of adaptively secure MPC with polylogarithmic communication locality. In Section 7.1, we show that all-pairs RMT can be used to realize CL MPC, and we outline our impossibility and feasibility results for CL MPC in the NE-AMS, E-NAMS, and E-AMS models. In Section 7.2, we show that our protocol for SOS-RMT in the NE-NAMS model can be used to realize SOS-MPC, and we state our final feasibility result for SOS-MPC with polylogarithmic CL.

7.1 CL MPC in the NE-AMS, E-NAMS, and E-AMS Models

All our negative results on all-to-all RMT trivially apply to MPC, since the former is a special case. Next, we show that a PKI, an appropriate hidden graph setup with polylogarithmic locality, and the standard cryptographic assumptions facilitating adaptively secure MPC in the complete (i.e., non-communication local) setting, are sufficient for MPC, under the combination of the feasibility bounds of all-pairs RMT proven here and the classical $t < n/2$ bound which is necessary and sufficient for MPC in the standard (non-communication local) setting.

The idea for the above is as follows: Execute the MPC protocol where the point-to-point communication is replaced by encrypting the message with the public key of the receiver and sending it using a fresh all-pairs RMT execution (as constructed here). As noticed in [CCG+15] the above seemingly straightforward idea has one caveat: to achieve adaptive security, in each round of the MPC protocol, the used RMT will require a new hidden graph setup which, in the worst case, induces an additive polylogarithmic increase in the CL in every round. To keep the overall communication locality of the MPC polylogarithmic, one needs to be careful that the total number of point-to-point rounds in the MPC protocol we are using is at most polylogarithmic. To this direction, [CCG+15] provided the following solution, which we also adopt here: Invocations to the (typically round-intensive) broadcast channel are replaced by a polylogarithmic-round broadcast protocol which was provided and proved secure in [CCG+15]. This protocol can be used within an adaptively secure constant-round MPC protocol (e.g., [BMR90]) to get an overall polylogarithmic-round MPC protocol.

Theorem 5. *Assuming a PKI, a polylogarithmic-degree hidden graph setup*[12]*, and trapdoor permutations with a reversed domain sampler, the following feasibility and impossibility statements hold for the existence of a* store-and-forward *protocol for securely evaluating any given n-party function against an adaptive t-adversary satisfying the following two conditions with overwhelming probability:*

- **Locality.** *Every party communicates with at most* $\mathcal{O}(\log^{1+\epsilon} n)$ *other parties, for some constant* $\epsilon > 0$.
- **Rounds.** *The protocol terminates after* $\mathcal{O}(\log^{\epsilon'} n)$ *rounds, for some constant* $\epsilon' > 0$.

1. *In the* NE-AMS *and* NE-NAMS *models, i.e. if we do not assume erasures, then no such MPC exists if* $t = O(n)$ *and the protocol has an expansion rate of* $(\log^z n, \frac{k \log n}{(1+\epsilon) \log\log n})$*, for some* $k < 1$ *and* $z > 1$.

[12] Recall that this can be replaced by an SKI or a NIKE scheme assuming the PKI supports it.

2. *In the* E-NAMS *and* E-AMS *models, i.e., if we assume erasures (with or without atomic multisend), then there exists such an MPC protocol if $t < (\frac{1}{2} - \epsilon'')n$ for some constant ϵ''.*

3. *In the* E-NAMS *model, no such MPC exists if $t > (\frac{1}{2} + \epsilon'')n$ for some constant ϵ''.*[13]

7.2 CL Sublinear-Output-Set (SOS) MPC in the NE-NAMS Model

Since we do not have all-pairs RMT in the NE-NAMS model, we must adopt a different approach. Here, we propose our protocol for sublinear output-set MPC (i.e., SOS-MPC), wherein a sublinear (in our case, polylogarithmic) set of parties is able to learn the output. The high-level outline of our protocol is as follows. A committee of some $\mathsf{polylog}(n)$ parties is selected; these are the parties that perform the actual MPC and ultimately learn the output. Next, each party creates $\mathsf{polylog}(n)$ secret shares of his MPC input, and he sends one share to each member of the commitee. Then, the committee members simulate an arbitrary MPC protocol to obtain the output.

The first task here is to select the committee members and keep their identities hidden from the adversary, while still allowing parties to send messages to committee members. We resolve this by introducing an additional assumption: an *anonymous* PKI setup. In particular, we dissociate the identities of parties from their public keys for all three schemes involved, namely the signature scheme, the non-committing encryption scheme, and the FHE scheme. Each party receives its secret keys, while the public keys are made known to everyone without disclosing the party's identity. In addition, the setup selects some $\mathsf{polylog}(n)$ parties uniformly at random to form the committee, and it publishes the set of public keys of committee members.

The next task is to allow all parties in the network to communicate their input shares to the committee members. Since the committee is of size $\mathsf{polylog}(n)$, this is easily achieved by running a single instance of SOS-RMT (using protocol $\Pi^{\mathsf{SOS-RMT}}_{\mathsf{FHE}}$), with each party sending a share of his input to each committee member.

The final task is for the committee members to execute the MPC protocol reliably. In particular, we must allow the committee members to communicate over hidden graphs without disclosing their location in the network. This can be realized by simulating each round of communication in the MPC protocol via a new instance of SOS-RMT (using protocol $\Pi^{\mathsf{SOS-RMT}}_{\mathsf{FHE}}$, with $\mathsf{polylog}(n)$ new hidden graphs), wherein each committee member sends the appropriate message for the MPC protocol to all other committee members, while the rest of the parties just send a dummy all-zeros message to each committee member. At the end of every RMT instance, each committee member simply decrypts only the messages received in slots corresponding to other committee members.

A detailed description of the complete protocol now follows. Note that it only requires executing multiple instances of $\Pi^{\mathsf{SOS-RMT}}_{\mathsf{FHE}}$ sequentially, and thus we immediately get Theorem 6, which states that the protocol achieves adaptively secure SOS-MPC with polylogarithmic communication locality.

Protocol for communication-local SOS-MPC. Our protocol assumes an anonymous PKI setup for (1) an existentially unforgeable digital signature scheme, (2) a non-committing encryption scheme, and (3) a compact and malicious circuit-private FHE scheme.

We also assume that the setup selects a committee $C = \{C_1, \ldots, C_{|C|}\}$ uniformly at random among the n parties (where $|C| = O(\log^c n)$ for some constant $c > 0$), and it publishes the list of committee public keys for all three schemes involved. Let each party p_i's input be x_i, and Π^{MPC} be an arbitrary honest majority MPC protocol. Our SOS-MPC protocol to compute $f(x_1, \ldots, x_n)$ proceeds as follows:

1. Initially, each party p_i computes a $(|C|/2)$-out-of-$|C|$ secret sharing of its input x_i into $|C|$ shares, denoted by $x_i^1, \ldots, x_i^{|C|}$.

2. Next, the parties run one instance of $\Pi^{\mathsf{SOS-RMT}}_{\mathsf{FHE}}$, where each party p_i sends input share x_i^j to committee member C_j $(1 \leq j \leq |C|)$, using the public keys of C_j for non-committing encryption and FHE. Recall

[13] Note that Theorem 3 implies that if we assume erasures as an atomic operation and no atomic multisend, then no MPC as in the above theorem exists if $t > (\frac{1}{2} + \epsilon'')n$ for some constant ϵ''. However, this is anyway implied by the tightness of the condition $t < n/2$ for adaptive security even in the complete (i.e., non-CL) point-to-point channels setting, and is therefore omitted.

that by the anonymous PKI setup, the identity of C_j as a party in the network, say p_k, is kept hidden. Still, $p_k (= C_j)$ is able to receive each share x_i^j $(1 \leq i \leq n)$ as it propagates through the network.

3. Having received shares of x_i from all p_i, the committee members execute Π^{MPC} among themselves to compute the output $f(x_1, \ldots, x_n)$, as follows. Each round of point-to-point communication in Π^{MPC} is replaced by an instance of $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$, where
   - each committee member $C_j \in C$ sends the appropriate messages for Π^{MPC} to all the other committee members in C, and
   - each party $p_i \notin C$ sends a dummy all-zeros message (of the same length as above) to all the committee members in C.

   At the end of every $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$ instance (i.e., every round of Π^{MPC}), each $C_j \in C$ decrypts the messages received from other committee members, and does any local computation prescribed by Π^{MPC}.
4. Finally, every committee member $C_j \in C$ learns the output $f(x_1, \ldots, x_n)$ (after an opening round, if necessary, which can be realized using another instance of $\Pi_{\mathsf{FHE}}^{\mathsf{SOS-RMT}}$).

Note that for an underlying protocol Π^{MPC} using R rounds of communication, our protocol requires $(R+1)\log^{\tilde{c}} n$ rounds of communication, for a constant $\tilde{c} > 1$ (dependent on the communication locality).

Theorem 6. *Assuming an anonymous PKI, a polylogarithmic-degree hidden graph setup, trapdoor permutations with a reversed domain sampler, and compact and malicious circuit-private FHE, there exists a protocol, satisfying the following two constraints with overwhelming probability:*
- **Locality.** *Every party communicates with at most $\mathcal{O}(\log^{1+\epsilon} n)$ other parties, for some constant $\epsilon > 0$, and*
- **Rounds.** *The protocol terminates after $\mathcal{O}(\log^{\epsilon'} n)$ rounds, for some constant $\epsilon' > 0$,*

which securely evaluates any given n-party function against an adaptive t-adversary corrupting up to $t < n/2$ parties in the $\mathsf{NE\text{-}NAMS}$ model, and delivers the output to any $\mathcal{O}(\log^{\epsilon''} n)$ parties, for constant $\epsilon'' > 0$.

References

ACD+19. Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. Communication complexity of byzantine agreement, revisited. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 317–326. ACM, 2019.

BAB+01. Micah Beck, Dorian C. Arnold, Alessandro Bassi, Jack J. Dongarra, Terry Moore, James S. Plank, D. Martin Swany, Sathish S. Vadhiyar, Richard Wolski, Francine Berman, Henri Casanova, and Graziano Obertelli. Logistical computing and internetworking: Middleware for the use of storage in communication. In *3rd Annual International Workshop on Active Middleware Services (AMS 2001), 6 August 2001, San Francisco, CA, USA*, pages 12–21. IEEE Computer Society, 2001.

BCDH18. Elette Boyle, Ran Cohen, Deepesh Data, and Pavel Hubácek. Must the communication graph of MPC protocols be an expander? pages 243–272, 2018.

BCP15. Elette Boyle, Kai-Min Chung, and Rafael Pass. Large-scale secure computation: Multi-party computation for (parallel) RAM programs. pages 742–762, 2015.

BGT13a. Elette Boyle, Shafi Goldwasser, and Stefano Tessaro. Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. pages 356–376, 2013.

BGT13b. Elette Boyle, Shafi Goldwasser, and Stefano Tessaro. Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. In *TCC*, pages 356–376, 2013.

BGW88a. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988.

BGW88b. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). pages 1–10, 1988.

BMR90. Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). pages 503–513, 1990.

Can00. Ran Canetti. Security and composition of multiparty cryptographic protocols. 13(1):143–202, January 2000.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. pages 136–145, 2001.

CCD88. David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988.

CCG$^+$14. Nishanth Chandran, Wutichai Chongchitmate, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky, and Vassilis Zikas. Optimally resilient and adaptively secure multi-party computation with low communication locality. *IACR Cryptol. ePrint Arch.*, page 615, 2014.

CCG$^+$15. Nishanth Chandran, Wutichai Chongchitmate, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky, and Vassilis Zikas. The hidden graph model: Communication locality and optimal resiliency with adaptive faults. pages 153–162, 2015.

CCGZ16. Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Probabilistic termination and composability of cryptographic protocols. pages 240–269, 2016.

CFG$^+$22. Nishanth Chandran, Pouyan Forghani, Juan A. Garay, Rafail Ostrovsky, Rutvik Patel, and Vassilis Zikas. Universally composable almost-everywhere secure computation. In Dana Dachman-Soled, editor, *3rd Conference on Information-Theoretic Cryptography, ITC 2022, July 5-7, 2022, Cambridge, MA, USA*, volume 230 of *LIPIcs*, pages 14:1–14:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

CFGN96a. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 639–648. ACM, 1996.

CFGN96b. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. pages 639–648, 1996.

CGO10. Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Improved fault tolerance and secure computation on sparse networks. In *ICALP (2)*, pages 249–260, 2010.

CGO12. Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Edge fault tolerance on sparse networks. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, pages 452–463, 2012.

DI06. Ivan Damgård and Yuval Ishai. Scalable secure multiparty computation. pages 501–520, 2006.

DIK10. Ivan Damgård, Yuval Ishai, and Mikkel Krøigaard. Perfectly secure multiparty computation and the computational overhead of cryptography. pages 445–465, 2010.

DKMS12. Varsha Dani, Valerie King, Mahnush Movahedi, and Jared Saia. Brief announcement: breaking the $O(nm)$ bit barrier, secure multiparty computation with a static adversary. In *ACM Symposium on Principles of Distributed Computing, PODC '12, Funchal, Madeira, Portugal, July 16-18, 2012*, pages 227–228, 2012.

DPPU86. Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. Fault tolerance in networks of bounded degree (preliminary version). pages 370–379, 1986.

FHKP13. Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. Non-interactive key exchange. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings*, volume 7778 of *Lecture Notes in Computer Science*, pages 254–271. Springer, 2013.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. pages 169–178, 2009.

GHK$^+$21. Craig Gentry, Shai Halevi, Hugo Krawczyk, Bernardo Magri, Jesper Buus Nielsen, Tal Rabin, and Sophia Yakoubov. YOSO: you only speak once - secure MPC with stateless ephemeral roles. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 64–93. Springer, 2021.

GIOZ17. Juan A. Garay, Yuval Ishai, Rafail Ostrovsky, and Vassilis Zikas. The price of low communication in secure multi-party computation. pages 420–446, 2017.

GKKZ11. Juan A. Garay, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. Adaptively secure broadcast, revisited. In Cyril Gavoille and Pierre Fraigniaud, editors, *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011, San Jose, CA, USA, June 6-8, 2011*, pages 179–186. ACM, 2011.

GMW87a. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings of the 19th Annual*

ACM Symposium on Theory of Computing, 1987, New York, New York, USA, pages 218–229. ACM, 1987.

GMW87b. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. pages 218–229, 1987.

GO08. Juan A. Garay and Rafail Ostrovsky. Almost-everywhere secure computation. In *EUROCRYPT*, pages 307–323, 2008.

GS12. Sanjam Garg and Amit Sahai. Adaptively secure multi-party computation with dishonest majority. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, pages 105–123, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

HZ10. Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. pages 466–485, 2010.

IPS08. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. pages 572–591, 2008.

KK06. Jonathan Katz and Chiu-Yuen Koo. On expected constant-round protocols for byzantine agreement. pages 445–462, 2006.

KMTZ13. Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 477–498. Springer, 2013.

KS10. Valerie King and Jared Saia. Breaking the $o(n^2)$ bit barrier: scalable byzantine agreement with an adaptive adversary. In *Proceedings of the 29th Annual ACM Symposium on Principles of Distributed Computing, PODC 2010, Zurich, Switzerland, July 25-28, 2010*, pages 420–429, 2010.

KSSV06a. Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Scalable leader election. In *SODA*, pages 990–999, 2006.

KSSV06b. Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Towards secure and scalable computation in peer-to-peer networks. In *FOCS*, pages 87–98, 2006.

KTZ13. Jonathan Katz, Aishwarya Thiruvengadam, and Hong-Sheng Zhou. Feasibility and infeasibility of adaptively secure fully homomorphic encryption. pages 14–31, 2013.

MNT22. Christian Matt, Jesper Buus Nielsen, and Søren Eller Thomsen. Formalizing delayed adaptive corruptions and the security of flooding networks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 400–430. Springer, 2022.

OPP14. Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. pages 536–553, 2014.

SV10. Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. pages 420–443, 2010.

Upf92. Eli Upfal. Tolerating linear number of faults in networks of bounded degree. In *PODC*, pages 83–89, 1992.

vGHV10. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. pages 24–43, 2010.

WXDS20. Jun Wan, Hanshen Xiao, Srinivas Devadas, and Elaine Shi. Round-efficient byzantine broadcast under strongly adaptive and majority corruptions. pages 412–456, 2020.

Yao82. Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). pages 160–164, 1982.

Supplementary Material

Proof of Theorem 4

Proof (sketch). We will show that the protocol $\Pi^{\mathsf{RMT}}_{\mathsf{FHE}}$ securely realizes the standard RMT functionality which allows the simulator to learn the message, but once he does, he cannot erase it without corrupting the sender (see e.g., [CCGZ16]).

To this direction we provide a simulator $\mathcal{S}$ that that simulates the view of any adversary $\mathcal{A}$ attacking $\Pi^{\mathsf{RMT}}_{\mathsf{FHE}}$.

$\mathcal{S}$ interacts (in a black-box, straight-line manner) with $\mathcal{A}$ and simulates towards $\mathcal{A}$: (1) the signature setup—by generating pairs of signing/verification keys for each party and giving $\mathcal{A}$ all verification keys along with the corrupted parties' signing keys; (2) the non-committing encryption setup—by creating encryption key and giving it to $\mathcal{A}$, and a corresponding decryption key so that if $\mathcal{A}$ corrupts the sender or the receiver, then $\mathcal{S}$ uses the non-commiting property to give $\mathcal{A}$ coins that decrypt all ciphertexts $\mathcal{A}$ has seen so far to the messages preferred by $\mathcal{S}$; (3) the FHE setup—by creating encryption keys and giving them to the adversary, and corresponding decryption keys which are given to $\mathcal{A}$ if he corrupts their intended owner. $\mathcal{S}$ also samples the hidden graph setup as in the real protocol, i.e., as an Erdős-Rényi graph.

For simulating the protocol execution, while neither the sender or receiver is corrupted, $\mathcal{S}$ simulates the code of all parties as follows. For simulating the first message, $\mathcal{S}$ generates each $\tilde{c}_w$ similarly to how the protocol parties would (using the simulated encryption keys), with the only difference that the signature σ encrypted under $\tilde{c}_u$ is computed with u's actual (simulated) signing key. From that point on, $\mathcal{S}$ can easily simulate all parties (by actually following the protocol) as he has control (and knowledge) of the whole (simulated) FHE setup. If $\mathcal{A}$ requests to corrupt any of the parties $w \notin \{u, v\}$, then $\mathcal{S}$ simply hands him the simulated state of that party.

If $\mathcal{A}$ requests to corrupt the sender, then the simulator learns the actual message transmitted by the sender, and uses the non-committing property to give $\mathcal{A}$ coins that encrypt this message to the ciphertext transmitted in the simulation. $\mathcal{S}$ also hands the adversary the sender's signing key. Subsequently, the simulator keeps track of which (if any) of the underlying simulated aaHE plaintexts would be delivered to the receiver in the simulation, and updates the input to the functionality with the smallest such message (or 0 if no such message is received).

If $\mathcal{A}$ requests to corrupt the receiver, then the simulator learns the actual output to the receiver from the functionality, and uses the non-committing property to give $\mathcal{A}$ a decryption key that opens u's transmitted ciphertext to this actual output. $\mathcal{A}$ then learns u's message whenever it is delivered by the simulation to the corrupted receiver.

We next argue that the view of $\mathcal{A}$ in the above experiment is indistinguishable from his view in the protocol. Indeed, the compactness and circuit-privacy property of the FHE ensures that all aaHE ciphertexts that the adversary sees in the protocol are indinsinguishable from one-another and also from the simulated ones. Since the distribution of the simulated hidden graph is identical to the protocol, the adversary cannot distinguish between the two executions, assuming the receiver is honest. If the receiver too is corrupted, then the views are trivially identical, as all ciphertexts gets decrypted to the actual messages in the protocol.

To complete the proof we need to argue that the output of the simulation is also indistinguishable from the output of the protocol. The hardest case here is when both the sender and the receiver are honest through the protocol/simulation. In this case, the simulated execution (ideal world) will always output the sender's message to the receiver. We next argue that this will also be the case in the real protocol. To this direction we show that any such $\mathcal{A}$ will always leave an honest path (i.e., a path consisting of only honest nodes) in the hidden graph of length at most R from the sender to the receiver. This combined with the malicious security of the FHE scheme implies that unless the adversary can forge the sender's signature—this is the only way to change the underlying aaHE ciphertext across an honest path—the message of the sender will be delivered through this path (and will be unique as the sender would never sign a different message with the same $\mathsf{msg\_ID}$).

To prove the above existence of an honest u-v path, assume towards contradiction that such a path does not exists. Then the above simulator can be used to turn $\mathcal{A}$ into a adversary strategy $\mathcal{A}'$ that disconnects u

from v while only performing hidden graph discovery before the protocol starts. This, however, contradicts the results from [CCG+15] which prove that any such $\mathcal{A}'$ corrupting (and removing) a constant fraction of parties has negligible probability of making the distance between u and v larger that $R = \log n$.

Given the existence of such an honest path, it is straight-forward to verify the correctness of the simulation also when the sender gets corrupted (for a corrupted receiver, correctness is trivial). Indeed, in this case, since $\mathcal{A}$ observes indistinguishable views in the real and simulated setting, the distributions of the output message that $\mathcal{A}$'s strategy induce (i.e., the way that the message changes when $\mathcal{A}$ injects additional and potentially contradicting signatures) must also be indistinguishable in the real and ideal setting.

Proof of Corollary 3

Proof (sketch). We will show that the protocol $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$ securely realizes all-to-one RMT, with $n-1$ senders and a single receiver v.

To this direction, we provide a simulator $\mathcal{S}'$, very similar to simulator $\mathcal{S}$ described in the proof of Theorem 4 above, that simulates the view of any adversary $\mathcal{A}$ attacking $\Pi_{\mathsf{FHE}}^{\mathsf{a21RMT}}$. $\mathcal{S}'$ interacts (in a black-box, straight-line manner) with $\mathcal{A}$ and simulates towards $\mathcal{A}$ the exact same setup as $\mathcal{S}$ in the preceding proof; i.e., signatures, non-committing encryption, FHE, and a hidden graph setup.

To simulate the protocol execution, $\mathcal{S}'$ simulates the code of all the parties as follows. To begin with, $\mathcal{S}'$ creates $n-1$ slots at each party of the form (u, v), where v is the receiver and u is any party except v. Then, for each such slot (u, v), in which u acts as the sender, $\mathcal{S}'$ generates each party's aaHE ciphertext $\tilde{c}_w$ exactly as it was in the protocol (using the simulated encryption keys), except u's own ciphertext $\tilde{c}_u$, which has its signature σ computed with u's actual (simulated) signing key. Subsequently, $\mathcal{S}'$ can easily simulate all $n-1$ slots for all the parties by following the protocol.

If $\mathcal{A}$ requests to corrupt any party $u \neq v$, then $\mathcal{S}'$ learns u's actual input to the all-to-one RMT functionality, and it gives $\mathcal{A}$ this input, along with the signing key of u, and coins that encrypt u's actual input to the simulated ciphertext $\tilde{c}_u$ in slot (u, v), using the non-committing property. $\mathcal{S}'$ also gives $\mathcal{A}$ the simulated state of u in all $n-1$ slots. Then, $\mathcal{S}'$ keeps track of which (if any) simulated aaHE plaintexts from u reach v in slot (u, v) of the simulation, and updates u's input to the functionality with the smallest one (or 0, if no such plaintext reaches v).

If $\mathcal{A}$ requests to corrupt the receiver v, then $\mathcal{S}'$ hands $\mathcal{A}$ the simulated state of v in all $n-1$ slots, along with the simulated decryption key of the FHE scheme. Further, $\mathcal{S}'$ also learns the actual outputs obtained by the receiver v from all $n-1$ senders in the ideal functionality, and it uses the non-committing property to compute a decryption key that opens the ciphertexts transmitted in all $n-1$ slots to the actual outputs obtained by the receiver. $\mathcal{A}$ then learns each sender's message whenever it is delivered by the simulation to the corrupted receiver in the corresponding slot.

Exactly as in the preceding proof, the compactness and circuit-privacy of the FHE scheme along with random sampling of the hidden graph ensures that the adversary's view in the above experiment is indistinguishable from his view in the protocol, in all $n-1$ slots, assuming the receiver is not corrupted. If the receiver too is corrupted, then the views are trivially identical, as all ciphertexts gets decrypted to the actual messages in the protocol.

We next argue the indistinguishability of outputs between the simulation and the protocol. Consider the case where $\mathcal{A}$ corrupts a subset T of the parties ($v \notin T$). Then, the ideal functionality will always output to receiver v the inputs of all honest parties $u \notin T$. We argue that this must also be the case in the real protocol. Observe that $\mathcal{A}$'s view, restricted to the slot (u, v), which is used by sender u, is exactly identical to that of the adversary in the simulation described in the preceding proof for single-pair RMT. Applying exactly the same reasoning for each $u \notin T$, $\mathcal{A}$ leaves an honest u-v path in the hidden graph of length $\leq R$, and along with the malicious security of FHE and existential unforgeability of the signature scheme, this guarantees that every honest sender u's message is delivered to v in the slot (u, v) with overwhelming probability. The correctness of the simulation in the case of a corrupted receiver is trivial.