Almost-Everywhere Secure Computation with Edge Corruptions*

Nishanth Chandran† Juan Garay‡ Rafail Ostrovsky§

Abstract

We consider secure multi-party computation (MPC) in a setting where the adversary can separately corrupt not only the parties (nodes) but also the communication channels (edges), and can furthermore choose selectively and adaptively which edges or nodes to corrupt. Note that if an adversary corrupts an edge, even if the two nodes that share that edge are honest, the adversary can control the link and thus deliver wrong messages to both players. We consider this question in the information-theoretic setting, and require security against a computationally unbounded adversary.

In a fully connected network the above question is simple (and we also provide an answer that is optimal up to a constant factor). What makes the problem more challenging is to consider the case of sparse networks. Partially connected networks are far more realistic than fully connected networks, which led Garay and Ostrovsky [Eurocrypt'08] to formulate the notion of (unconditional) *almost everywhere (a.e.) secure computation* in the node-corruption model, i.e., a model in which not all pairs of nodes are connected by secure channels and the adversary can corrupt some of the nodes (but not the edges). In such a setting, MPC amongst all honest nodes cannot be guaranteed due to the possible poor connectivity of some honest nodes with other honest nodes, and hence some of them must be "given up" and left out of the computation. The number of such nodes is a function of the underlying communication graph and the adversarial set of nodes.

In this work we introduce the notion of *almost-everywhere secure computation with edge corruptions*, which is exactly the same problem as described above, except that we additionally allow the adversary to completely control some of the communication channels between two correct nodes—i.e., to "corrupt" edges in the network. While it is easy to see that an a.e. secure computation protocol for the original node-corruption model is also an a.e. secure computation protocol tolerating edge corruptions (albeit for a reduced fraction of edge corruptions with respect to the bound for node corruptions), no polynomial-time protocol is known in the case where a **constant fraction** of the edges can be corrupted (i.e., the maximum that can be tolerated) and the degree of the network is sub-linear.

We make progress on this front, by constructing graphs of degree $O(n^\epsilon)$ (for arbitrary constant $0 < \epsilon < 1$) on which we can run a.e. secure computation protocols tolerating a constant fraction of adversarial edges. The number of given-up nodes in our construction is μn (for some constant $0 < \mu < 1$ that depends on the fraction of corrupted edges), which is also asymptotically optimal.

*A version of this paper, entitled "Edge Fault Tolerance on Sparse Networks," is scheduled to appear in the proceedings of the 39th *International Colloquium on Automata, Languages and Programming* (ICALP 2012); this version presents a more cryptographically oriented treatment, and also includes proofs and additional details.

†Microsoft Research, Redmond; Email: nish@microsoft.com. Part of this work was done while the author was at UCLA.

‡AT&T Labs – Research; Email: garay@research.att.com.

§Department of Computer Science and Department of Mathematics, UCLA, 3732D Boelter Hall, Los Angeles CA 90095-1596, U.S.; Email: rafail@cs.ucla.edu. Supported in part by NSF grants 0830803, 09165174, 1065276, 1118126 and 1136174, US-Israel BSF grant 2008411, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

Key words: Almost-everywhere secure computation, bounded-degree network, secure message transmission, Byzantine agreement.

1 Introduction

Secure multi-party computation (MPC) [27, 16, 3, 7] is perhaps the most popular paradigm in the area of cryptographic protocols. It requires that n parties compute some function f of their inputs without revealing any additional information, even when some of them may behave arbitrarily. More specifically, n parties $P_1, \cdots, P_n$, each holding input x_i, must run a protocol such that at the end of the protocol, all "honest" (i.e., not misbehaving) parties obtain $f(x_1, \cdots, x_n)$, while any set of colluding malicious parties $\mathcal{T}_{\text{nodes}} \subset [n]$ learn nothing more than what can be learnt from $f(x_1, \cdots, x_n)$ and the x_i values for all $P_i \in \mathcal{T}_{\text{nodes}}$.

Traditionally, MPC protocols assume that any two of the n parties share a private and authenticated channel which they can use for communication. In the case of information-theoretic MPC, the channel between two honest nodes is assumed to be a secure physical link which the adversary cannot eavesdrop nor tamper with; in the computational case, this channel is provided via a public-key infrastructure.

However, in many settings, the assumption of incorruptible pair-wise secure chanels might be unrealistic and overoptimistic. Think for example, of the communication subsystem of a networked computer (e.g., network interface controller card) being infected by malicious software designed to disrupt or alter operation. This would affect the communication between honest parties. Or worse, of a scenario where the secret keys shared by two parties in the system is compromised, yet the parties themselves are honest. In light of this, one first question we can ask is whether we can obtain MPC protocols when both parties as well as communication channels are corrupted.

Naturally, we must assume some bound on the number of parties as well as the number of communication channels that the adversary can corrupt. First, observe that we cannot hope to obtain a protocol that is secure against an adversary that corrupts at least $\frac{n^2}{3} + n$ of the n^2 communication channels[1] (a constant fraction of the edges in the network), even if all the parties are honest. This is because, such an adversary can effectively "simulate" the corruption of $\frac{n}{3} + 1$ out of the n parties (in the standard setting where only parties can be corrupted). and in this case MPC is known to be impossible [3]. (In this work, we focus on information-theoretic MPC.) On the positive side, it is quite easy to construct protocols that asymptotically match these bounds of edge corruptions; i.e., protocols tolerating corruption of a constant fraction of communication channels as well as a constant fraction of parties (for constants $< \frac{1}{3}$)[2].

However, all the protocols discussed so far assume that any two parties share a secure channel to begin with! (These channels may later on be corrupted, but the connectivity of every party is still n.) For protocols that are executed over large networks such as the Internet, in which nodes are typically connected by a communication graph of small degree, this assumption is unreasonable. Thus, we turn our attention to the problem of constructing MPC protocols (secure against corruption of both parties and communication channels) in which parties (or nodes) have low connectivity.

1.1 Almost-everywhere secure computation

Before we describe our model in more detail, we remark that the setting of constructing protocols on networks that are not fully connected is not new. Obtaining protocols on networks that are of

[1] For ease of exposition, we will assume that the total number of communication channels between the n parties is n^2 (i.e., we assume a "self-channel").

[2] Note that such a protocol will, unavoidably,"leave out" certain honest parties from the computation. We will discuss this in more detail later on.

low degree (in the presence of node corruptions alone), was first considered for the task of Byzantine agreement [20, 19] in the seminal work of Dwork, Peleg, Pippenger, and Upfal [12]. More formally, in the Dwork *et al.* formulation, the n parties (or nodes) are connected by a communication network G. Nodes that are connected by an edge in G share a reliable and authentic channel, but other nodes must communicate via paths in the graphs that may not be available to them (due to the adversarial "corruption" of some of the nodes).

Naturally, in such a setting, one may not be able to guarantee agreement amongst all honest parties; for example, one cannot hope to be able to communicate at all with an honest party whose neighbors are all adversarial. Given this fact—and ubiquitously—Dwork *et al.* termed the new problem *almost-everywhere (a.e.) agreement*, wherein the number of such abandoned nodes (which henceforth will be called "doomed") introduces another parameter of interest, in addition to the degree of the communication graph (which we wish to minimize), and the number of adversarial nodes that can be tolerated (which we wish to maximize) in reaching agreement.

Indeed, in [12], Dwork *et al.* provide a.e. agreement protocols for various classes of low-degree graphs and bounds on the number of adversarial nodes as well as abandoned nodes. For example, they construct a graph of constant degree and show an agreement protocol on this graph tolerating a $\frac{\alpha}{\log n}$ fraction of corrupted nodes (for constant $0 < \alpha < 1$), guaranteeing agreement amongst $(1 - \alpha - \mu)n$ of the honest nodes (for constant $0 < \mu < 1$). In another construction, they give a graph of degree $\mathcal{O}(n^{\epsilon})$ (for constant $0 < \epsilon < 1$) and show an agreement protocol on this graph tolerating a constant α $(0 < \alpha < 1)$ fraction of corrupted nodes, and again guaranteeing agreement amongst $(1-\alpha-\mu)n$ nodes. In a subsequent and remarkable result, Upfal [23] constructed a constant-degree graph and showed the existence of an a.e. agreement protocol on this graph tolerating a constant fraction of corrupted nodes, while giving up a constant fraction of the honest nodes. Unfortunately, the protocol of [23] runs in exponential time (in n).

Garay and Ostrovsky [15] were the first to consider the problem of unconditional MPC in the context of partially connected networks with adversarial nodes, and obtained results with bounds similar to those in [12]. More recently, Chandran, Garay, and Ostrovsky [6] constructed a graph of degree $\mathcal{O}(\log^k n)$ (for constant $k > 1$) and show an agreement, as well as an MPC protocol on this graph tolerating a constant fraction of corrupted nodes, while giving up only $\mathcal{O}(\frac{n}{\log n})$ of the honest nodes.

1.2 Almost-everywhere secure computation with edge corruptions

All existing work on a.e. agreement and a.e. secure computation mentioned above considers the case where only nodes may be corrupted and misbehave [3]. Furthermore, all existing work, construct *specific* graphs (of low degree) on which one can obtain an a.e. secure computation protocol. In an ideal scenario, we would like to construct a.e. computation protocols on arbitrary adversarially chosen communication networks. Unfortunately, this is impossible in general[4] . However, we can take a step in this direction by allowing the adversary to corrupt edges in the network that we design. That is, we will still construct a specific graph (on which we will obtain a.e. computation protocols); however, we will allow the adversary to "modify" this graph by corrupting a constant fraction of edges in it, thereby taking down (and even actively corrupting) certain channels in the communication network.

In more detail, in this work we will endow the adversary with additional powers which allow him, in addition to corrupting nodes, to corrupt some of the *edges* in the network—i.e., we consider *a.e. agreement and computation with edge corruptions*. When he does (corrupt an edge), he is able to

[3] In the closely related problem of *secure message transmission* [11] (and its variants), however, it is assumed that the "wires" (abstraction of paths in a network) connecting sender and receiver might be corrupted and misbehave.

[4] The adversary could simply design networks where several nodes have extremely poor connectivity and hence corrupting a few edges could create several disconnected components in the network of small size.

completely control the communication channel between the two honest nodes, from simply preventing them to communicate, to injecting arbitrary messages that the receiving end will accept as valid. As in the node-only corruption case, in this case also some of the honest nodes in the network must be abandoned. In this work, we ask the following question:

> *Can we obtain a.e. secure computation protocols on networks of low (i.e., sublinear) degree when a constant fraction of nodes as well as communication channels are corrupted?*

Which we answer in the affirmative. To put things in perspective, observe that an a.e. agreement or a.e. secure computation protocol for node corruptions can be readily transformed into a corresponding a.e. protocol also tolerating edge corruptions, albeit for a reduced fraction of edge corruptions. More specifically, let d be the maximum degree of any node in a graph $\mathcal{G}$ on n nodes that admits an a.e. agreement (a.e. computation) protocol Π amongst $p < n$ nodes, in the presence of x corrupt nodes. Then, it is easy to see that $\mathcal{G}$ admits an a.e. agreement (a.e. computation) protocol Π' amongst p nodes in the presence of x corrupt edges[5]. However, this means that the graph will only admit an agreement/computation protocol for an $\frac{x}{nd}$ fraction of corrupted edges, as opposed to an $\frac{x}{n}$ fraction of corrupted nodes in the former case. Therefore, the result that we get for the case of edge corruptions using this naïve method is asymptotically weaker than in the case of node corruptions (except when d is a constant). Indeed, by applying this method, none of the existing protocols for a.e. agreement/computation against node corruptions give us a solution tolerating a constant fraction of edge corruptions. This is depicted in Table 1, where we outline the results one would obtain by applying this approach to the a.e. agreement/computation protocols for node corruptions of [12, 23, 18, 15, 6], and compare them with the results we obtain in this work. In all the results listed in the table, $0 < \alpha < 1$ is a constant, $0 < \epsilon < 1$ can be any arbitrary constant, and $k > 1$ is a constant.

Reference	Graph degree	Frac. of corrupt edges	Graph/Protocol	Running time
[12, 15]	$\mathcal{O}(n^{\epsilon})$	$\frac{\alpha}{n^{\epsilon}}$; adaptive	Explicit/Deterministic	Polynomial
[12, 15]	$\mathcal{O}(1)$	$\frac{\alpha}{\log n}$; adaptive	Explicit/Deterministic	Polynomial
[23]	$\mathcal{O}(1)$	α; adaptive	Explicit/Deterministic	Exponential
[18]	$\mathcal{O}(\log^k n)$	$\frac{\alpha}{\log^k n}$; static	Explicit/Randomized	Polynomial
[6]	$\mathcal{O}(\log^k n)$	$\frac{\alpha}{\log^k n}$; adaptive	Explicit/Deterministic	Polynomial
[This work]	$\mathcal{O}(n^{\epsilon})$	α; adaptive	Randomized/Deterministic	Polynomial

Table 1: *A.e. secure computation against edge corruptions from a.e. secure computation against node corruptions.*

Note that all the previous results (except for the result obtained as a corollary to [23], in which the protocol's running time is exponential) cannot handle the case where we have a *constant* fraction of corrupted edges. Here we are precisely interested in this case. Specifically, we construct the first protocols for a.e. agreement and a.e. secure computation on graphs with sub-linear degree that can tolerate a constant fraction of edge corruptions. We remark that while the above graph constructions are deterministic, we construct our graph probabilistically, and our result holds with high probability. However, a graph satisfying the conditions required for our protocols to be successful can be sampled with probability $1-\mathsf{neg}(n)$, where $\mathsf{neg}(n)$ denotes a function that is negligible in n, and furthermore, one can also efficiently check if the graph thus sampled satisfies the necessary conditions for our protocols.

[5] To simulate an adversary that corrupts an edge (u, v), simply corrupt either node u or v.

1.3 Overview of our results

We show that for every constant ϵ, $0 < \epsilon < 1$, there exists a graph, call it $\mathcal{G}_{\text{main}}$, on n nodes, with maximum degree $d_m = \mathcal{O}(n^\epsilon)$, and such that it admits a.e. agreement and a.e. secure computation protocols that guarantee agreement/computation amongst $\gamma_m n$ honest nodes (for some constant $0 < \gamma_m < 1$), even in the presence of an α_m fraction of corrupted edges (i.e., at most $\frac{\alpha_m n d_m}{2}$ corrupted edges), for some constant $0 < \alpha_m < 1$. Our protocols work against an adversary that is *adaptive* (i.e., the adversary can decide which edges to corrupt on the fly during the protocol after observing messages of honest parties) and *rushing* (i.e., in every round, the adversary can decide on its messages after seeing the messages from the honest parties).

First, note that the problem of a.e. secure computation can be reduced to the problem of obtaining a public and authentic channel between any two pair of nodes in the set of privileged nodes (i.e., the set of honest nodes from which the set of doomed nodes have been excluded), due to techniques developed in [15]; and the problem of a.e. agreement trivially reduces to the problem of obtaining such a public channel. Hence, our main focus will be on constructing a protocol for this task. We now outline the high-level ideas behind our construction:

1. The first step in our construction is to build a graph with higher degree, $\mathcal{O}(\sqrt{n}\log n)$, on which we can have obtain a public channel (between any two nodes in a set of size $\mathcal{O}(n)$) tolerating a constant fraction of corrupted edges.
   - To do this, we first observe a property of a graph that is sufficient for such a construction (besides, obviously, every node having degree $\mathcal{O}(\sqrt{n}\log n)$), namely, that any two nodes in the graph have $\mathcal{O}(\log^2 n)$ number of paths of length 2 between them.
   - Second, we observe that the Erdős-Renyi random graph $G(n, \frac{\log n}{\sqrt{n}})$ satisfies the above two properties with high probability. That is, a graph $\mathcal{G}$ on n nodes satisfying the above two properties can be easily sampled by putting an edge (u, v) in $\mathcal{G}$, independently, with probability $p = \frac{\log n}{\sqrt{n}}$.
   - Once we have a graph satisfying these properties, the construction is fairly straightforward: to obtain a public channel between any two nodes, say, u and v, u simply sends the message to all nodes in the network via all the paths of length 2, and all the nodes then send the message to v, again via all their paths of length 2. One can then show that if v takes a simple majority of the received values, then a constant fraction of the nodes can build a public channel even in the presence of a constant fraction of corrupted edges.
2. Next, we show how to construct a graph, $\mathcal{G}'$, recursively from $\mathcal{G} \leftarrow G(n, \frac{\log n}{\sqrt{n}})$ above such that the new graph is of size n^2 and its degree at most twice that of $\mathcal{G}$, and yet we can have a public channel on $\mathcal{G}'$ (between every pair of privileged nodes) tolerating a constant fraction of corrupted edges.
   - We construct $\mathcal{G}'$ by taking n "copies" of $\mathcal{G}$ to form n "clouds," and then connecting the clouds using another copy of $\mathcal{G}$. We connect two clouds by connecting the i^{th} node in one cloud with the i^{th} node in the other.
   - Now our hope is to be able to simulate the communication between two nodes u and v in the following way: u will send the message to all nodes in its cloud (call this cloud C_u). Cloud C_u will then send the message to cloud C_v (the cloud which v is a part of). Finally, v will somehow receive the message from cloud C_v.
   - The problem with this approach is that we need to have a protocol that will allow two clouds to communicate reliably. But clouds themselves are comprised of nodes, some of which might be corrupted or doomed; hence, the transmission from cloud C_u to cloud C_v might end up being unreliable. To get over this problem, we make use of a specific type of agreement protocol

known as *differential agreement* [13], which, informally and whenever possible, allows parties to agree on the majority value of the honest parties' inputs. Careful application of this protocol allows us to perform a type of "error-correction" of the message when it is being transferred from one cloud to another.

- Combining the above techniques leads us to our main result, the construction of a public channel between any two nodes in a set of size $\mathcal{O}(n)$, on graphs of degree $\mathcal{O}(n^\epsilon)$ (for all constants $0 < \epsilon < 1$), tolerating a constant fraction of corrupted edges, while giving up μn honest nodes (for a constant $0 < \mu < 1$).

Our new public channel construction directly gives us an a.e. agreement protocol tolerating both node and edge corruptions (with the required parameters), and by applying our protocol to the construction in [15], we also obtain an a.e. MPC protocol tolerating both node and edge corruptions for graphs of degree $\mathcal{O}(n^\epsilon)$ and same parameters as above.

1.4 Related work

As mentioned above, the problem we consider is most closely related to the problem of almost-everywhere agreement tolerating node corruptions [12, 5, 23, 6]. To our knowledge, the problem of agreement or computation on networks with faulty (either Byzantine or not) edges has been only considered before either in the setting of random faults or where there are limited edge faults in a complete network. We now present a brief overview.

The work of Diks and Pelc [9] considers the problem of obtaining an agreement protocol (amongst all nodes) in a complete network when every edge in the network can fail independently with constant probability p_f (this in turn means that the expected degree of every node is linear (in n)). This work also deals only with non-Byzantine faults. The work of Pelc [21] also considers only probabilistic edge faults (where the failure probability is a constant less than 0.29), but considers Byzantine faulty edges. The work of Chlebus *et al.* [8] considers the round complexity of agreement protocols in the presence of probabilistic, non-Byzantine edge faults (where the failure probability is a constant). Shanbhogue and Yung [22] study the necessary and sufficient conditions for asynchronous agreement protocols amongst n nodes on a complete graph in the presence of Byzantine edges (and show a tight upper and lower bound of $\lfloor \frac{n-2}{2} \rfloor$ on the total number of corrupted edges). Similarly, the work of Yan *et al.* [26, 24] also considers malicious faulty links in complete networks with the aim of constructing protocols for optimal-round Byzantine agreement, while Wang and Yan [25] additionally consider fault detection in a similar setting. Berman *et al.* [4] consider agreement on protocols with random Byzantine edge faults (with a probability of failure $< \frac{1}{2}$) and show a broadcasting algorithm that works for n nodes in time $\mathcal{O}(\log n)$, with probability of correctness $1 - \frac{1}{n}$.

The work of Barak *et al.* [1] considered the problem of secure computation in the computational setting without authentication. In this setting, parties wish to run secure computation protocols when *no* two of them share an authenticated channel, and, furthermore, the adversary controls the delivery of messages. In this setting, the adversary can trivially partition the parties into various sets of parties, with each set running its own "secure" computation protocol. Barak *et al.* show that, essentially, this is all that an adversary can do. We remark that our setting is different – while some pairs of parties share authenticated channels in our setting (namely those that are connected by an edge in the communication network), we wish to include a much larger fraction of honest nodes in the computation protocol (by making use of other nodes in the network to perform secure message transmission).

The work of Gordon *et al.* [17] considered the problem of broadcast with a partially compromised public-key infrastructure. Their work is limited to networks that are initially fully connected, and are furthermore, in the computational setting. Zikas *et al.* [28] considered a model of secure computation, where an adversary, can in addition to corrupting parties, can also block outgoing or incoming messages

of honest parties. Their setting differs from ours in the following two ways. Their work permits the adversary to only block messages of honest parties, whereas, in our setting, when an adversary takes over a channel, he completely controls all messages on it. Secondly, their setting allows the adversary to block only either *all* incoming or outgoing messages of a party, or none, whereas, we allow the adversary to selectively corrupt the edges incident on any node.

Dolev [10] and Dolev *et al.* [11] showed that if there are t faulty processors in a network, then every pair of processors can communicate reliably if and only if every node has connectivity at least $2t + 1$. This work also assumes that whenever a connection is present between two nodes, the connection is reliable and not under the control of an adversary (unlike in our work). In order to overcome the barrier on the maximum number of adversaries that can be corrupted, Beimel and Franklin [2], considered a model in which some pairs of processors (other than the pairs that are already connected by reliable channels), are also given authentication keys. The reliable channels defines a natural "communication graph" and the pairs of parties sharing authentication keys define a natural "authentication graph". Beimel and Franklin showed that every pair of processors can communicate reliably if and only if all nodes in the communication graph have connectivity at least $t+1$ and the union of the two graphs has connectivity at least $2t + 1$.

2 Model, Definitions and Building Blocks

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph with n nodes (i.e., $|\mathcal{V}| = n$). The nodes of the graph represent the processors (parties) participating in the protocol, while the edges represent the communication links connecting them. We assume a synchronous network and that the protocol communication is divided into rounds. In every round, all parties can send a message on all of their communication links (i.e., on all edges incident on the corresponding node); these messages are delivered before the next round.

An adversary $\mathcal{A}$ can "corrupt" a set of nodes (as in taking over them and completely control their behavior), $\mathcal{T}_{\text{nodes}} \subset \mathcal{V}$, as well as a set of edges, $\mathcal{T}_{\text{edges}} \subset \mathcal{E}$, in the network such that $|\mathcal{T}_{\text{nodes}}| \leq t_n$ and $|\mathcal{T}_{\text{edges}}| \leq t_e$. $\mathcal{A}$ has unbounded computational power and can corrupt both nodes and edges *adaptively* (that is, the adversary can decide which nodes and edges to corrupt on the fly during the course of the protocol, after observing the messages from honest parties). Furthermore, $\mathcal{A}$ is *rushing*, meaning that it can decide the messages to be sent by adversarial parties (or on adversarial edges) in a particular round after observing the messages sent by honest parties in the same round.

Almost-everywhere agreement. The problem of *almost-everywhere agreement* ("a.e. agreement" for short) was introduced by Dwork *et al.* [12] in the (traditional) context of node corruptions. A.e. agreement "gives up" some of the non-faulty nodes in the network from reaching agreement, which is unavoidable due to their poor connectivity with other non-faulty nodes. We refer to the given-up nodes as *doomed* nodes, and to the honest nodes for which we guarantee agreement as *privileged* nodes. Let the set of doomed nodes be denoted by $\mathcal{X}$ and the set of privileged nodes by $\mathcal{P}$; note that the sets $\mathcal{P}$ and $\mathcal{X}$ are a function of the set of corrupted nodes ($\mathcal{T}_{\text{nodes}}$) and the underlying graph. Let $|\mathcal{X}| = x$ and $|\mathcal{P}| = p$. Clearly, we have $p + x + t = n$. We now present a definition of a.e. agreement.

Definition 1 *A protocol for parties $\{P_1, P_2, \cdots, P_n\}$, each holding initial value v_i, is an* almost-everywhere agreement *protocol for node corruptions if the following conditions hold for any adversary $\mathcal{A}$ that corrupts a set of nodes $\mathcal{T}_{\text{nodes}}$ with $|\mathcal{T}_{\text{nodes}}| \leq t$:*

- Agreement: *All nodes in $\mathcal{P}$ output the same value.*
- Validity: *If for all nodes in $\mathcal{P}$, $v_i = v$, then all nodes in $\mathcal{P}$ output v.*

The difference with respect to standard Byzantine agreement [20, 19] is that in the latter the two conditions above are enforced on *all* honest nodes, as opposed to only the nodes in $\mathcal{P}$. For brevity, we keep the same names.

In the context of a.e. agreement, one would like the graph $\mathcal{G}$ to have as small a degree as possible (in relation to the size of the graph and to the number of corrupted parties), while tolerating a high value for t_n (a constant fraction of n is the best possible), while minimizing x.

In [12], Dwork *et al.* construct graphs with degree $\mathcal{O}(n^\epsilon)$ (for constant $0 < \epsilon < 1$) tolerating at most $t_n = \alpha_n n$ (for constant $0 < \alpha_n < 1$) corruptions and at the same time guaranteeing agreement amongst $n - \mathcal{O}(t_n)$ nodes in the network, with a number of doomed nodes a constant times t_n; call such a graph $\mathcal{G}_{\text{DPPU}} = (\mathcal{V}_{\text{DPPU}}, \mathcal{E}_{\text{DPPU}})$. The idea behind the a.e. agreement protocol is to simulate a complete graph on the set of privileged nodes. The theorem from [12] is as follows:

Theorem 1 [12] *There exist constants $0 < \alpha, \mu < 1$ independent of n and t_n, an n-vertex $\mathcal{O}(n^\epsilon)$-regular graph $\mathcal{G}_{\text{DPPU}}$ which can be explicitly constructed, and a communication protocol (transmission scheme) $\mathsf{TS}_{\text{DPPU}}$, such that for any set of adversarial nodes $\mathcal{T}_{nodes}$ in $\mathcal{G}_{\text{DPPU}}$ such that $|\mathcal{T}_{nodes}| = t_n = \alpha_n n$, $\mathsf{TS}_{\text{DPPU}}$ guarantees reliable communication between all pairs of nodes in a set of non-faulty nodes $\mathcal{P}$ of size $|\mathcal{P}| \geq n - t_n - \mu n$. The protocol generates a polynomial (in n) number of messages and has polynomial (in n) running time.*

Given the above theorem, Dwork *et al.* observe that one can run any Byzantine agreement protocol designed for a fully connected graph on $\mathcal{G}_{\text{DPPU}}$ by simulating all communication between nodes in the network with the communication protocol $\mathsf{TS}_{\text{DPPU}}$. We refer the reader to [12] for further details.

As a result, let μ, d, and t_n be as defined above and let $\mathsf{BA}(n, t'_n)$ be a Byzantine agreement protocol for a complete network tolerating up to $t'_n = \mu n + t_n$ faulty processors. Then, simulating the protocol $\mathsf{BA}(n, t'_n)$ on the network $\mathcal{G}_{\text{DPPU}}$ using the communication protocol $\mathsf{TS}_{\text{DPPU}}$, guarantees agreement among at least $n - t_n - \mu n$ honest nodes in the presence of up to $t_n = \alpha_n n$ faulty nodes.

Differential agreement. In [13], Fitzi and Garay introduced the problem of *δ-differential agreement*. In the standard Byzantine agreement problem, n parties attempt to reach agreement on some value v (for simplicity, we assume $v \in \{0, 1\}$). Let c_v denote the number of honest parties whose initial value is v, and δ be a non-negative integer. *δ-differential agreement* is defined as follows:

Definition 2 *A protocol for parties $\{P_1, P_2, \cdots, P_n\}$, each holding initial value v_i, is a* δ-differential agreement *protocol if the following conditions hold for any adversary $\mathcal{A}$ that corrupts a set $\mathcal{T}_{\text{nodes}}$ of parties with $|\mathcal{T}_{\text{nodes}}| \leq t_n$:*

- AGREEMENT: *All honest parties output the same value.*
- δ-DIFFERENTIAL VALIDITY: *If the honest parties output v, then $c_v + \delta \geq c_{\bar{v}}$.*

Theorem 2 [13] *In a synchronous, fully connected network, δ-differential agreement is impossible if $n \leq 3t_n$ or $\delta < t_n$. On the other hand, there exists an efficient (i.e., polynomial-time) protocol that achieves t_n-differential agreement for $n > 3t_n$ in $t_n + 1$ rounds.*

We will use $\mathsf{DA}(n, t_n, \delta)$ to denote a δ-differential agreement protocol for a fully connected network tolerating up to t_n faulty processors.

The edge-corruption model. In this work we additionally allow the adversary to corrupt edges on the network graph—the set $\mathcal{T}_{\text{edges}} \subset \mathcal{E}$, $|\mathcal{T}_{\text{edges}}| \leq t_e$. We will bound this quantity, as well as the total number of nodes that the adversary can corrupt, and attempt to construct a network graph $\mathcal{G}$ of small (sublinear) degree on which a significant number of honest nodes can still perform MPC. We now give some definitions and make some remarks about a.e. agreement and a.e. MPC for this setting.

We first observe that since we are working with (asymptotically) regular graphs, obtaining an a.e. (agreement, MPC) protocol in the presence of a constant fraction of corrupted edges will also imply a protocol in the presence of a constant fraction of corrupted edges and a constant fraction of corrupted nodes nodes, as every corrupted node can be "simulated" by corrupting all the edges incident on this node. Thus, we will henceforth consider only adversarial edges and assume that all the nodes are honest.

As in the case of a.e. MPC on sparse networks in the presence of adversarial nodes, a.e. MPC in the presence of adversarial edges also "gives up" certain honest nodes in the network, which, as argued before, is unavoidable due to their poor connectivity with other honest nodes. Let the set of such doomed nodes be denoted by $\mathcal{X}$ and the set of privileged nodes by $\mathcal{P}$. Note that the sets $\mathcal{P}$ and $\mathcal{X}$ are a function of both the set of corrupted edges ($\mathcal{T}_{\text{edges}}$) and the underlying graph. Let $|\mathcal{X}| = x$ and $|\mathcal{P}| = p$; we let the fraction of corrupt edges be α_e. The definition of a.e. agreement with corrupted edges, in particular, now readily follows in the same manner as in Definition 1.

Next, we remark that the problem of a.e. agreement for edge corruptions also reduces to that of constructing a reliable and authentic channel between any two nodes $u, v \in \mathcal{P}$, in particular those which are not directly connected by an edge in $\mathcal{E}$. (With foresight, we will be calling such a channel a *public channel*.) Furthermore, Garay and Ostrovsky showed that, given such a public channel between two nodes u and $v \in \mathcal{P}$, plus some additional paths, most of which (i.e., all but one) might be corrupted, it is possible to construct a (unidirectional) *secure* (i.e., private and reliable) channel between them. The construction is via a protocol known as *secure message transmission by public discussion* (SMT-PD) [15, 14]. In turn, from the protocol for a secure channel, an a.e. MPC protocol amongst the nodes in $\mathcal{P}$, satisfying the same notion of security as in [15], readily follows (see Theorem 7 in Appendix A).

The definition of a secure channel between two nodes, which are not necessarily directly connected, and in the presence of adversarial edges, essentially paraphrases the more general definition of secure message transmission (see, e.g., [14]), where the channel is parameterized by a privacy error (κ_1) and a reliability error (κ_2).

More formally, an execution of a (unidirectional) secure channel protocol between two nodes $u, v \in \mathcal{V}$, with u as the sender and u as the receiver, is determined by the random coins of the two nodes and the adversary $\mathcal{A}$ controlling the set of edges $\mathcal{T}_{\text{edges}}$, and the message m (without loss of generality, we will assume that $m \in \{0, 1\}$). Let $\mathsf{View}_{\mathcal{A}}$ denote the view of the adversary in the execution of the protocol between u and v; this view includes all messages seen by $\mathcal{A}$ (on edges in the set $\mathcal{T}_{\text{edges}}$), as well as the adversary's random coins. Let $\mathsf{View}_{\mathcal{A}}(b)$ denote the view of the adversary. In each execution, v will output a message m_v. We now describe the security of secure-channel protocol:

Definition 3 *We say that a protocol* Π *between nodes* $u, v \in \mathcal{P}$*, is a* (κ_1, κ_2)-secure channel *protocol if it satisfies:*

- Privacy: $\mathsf{View}_{\mathcal{A}}(0)$ *and* $\mathsf{View}_{\mathcal{A}}(1)$ *are* κ_1*-close.*
- Reliability: *For* $m \in \{0, 1\}$, $\Pr[m_v = m] \geq 1 - \kappa_2$. *(The probability is taken over all players' coins.)*

Similarly, we say a protocol Π between nodes $u, v \in \mathcal{P}$, is a κ*-public channel* protocol between u and v if it satisfies the reliability property (as defined above) with probability at least $1 - \kappa$.

For completeness, we also provide the security definition of a.e. MPC (for node-corruptions) given in [15] in Appendix A.

Finally, we remark that one can define the notion of *a.e.* differential agreement (for edge corruptions) in the same manner as a.e. agreement by replacing the set of honest parties with the set of privileged parties in Definition 2 (i.e., by treating doomed parties also as adversarial). Furthermore, note that one can also obtain an a.e. differential agreement protocol (for edge corruptions) from the construction of a public channel between any two nodes $u, v \in \mathcal{P}$: simply, execute a standard differential agreement protocol and replace every communication between nodes with an execution of the public channel protocol. We will use $\mathsf{AE\text{-}DA}(n, t_n, \delta)$ to denote an a.e. δ-differential agreement protocol for a partially connected network where the number of privileged parties is $n - t_n$.

3 Remote and Secure Channels on Low-Degree Networks

In this section we construct a graph in which the maximum degree of any node is low, and yet, there exists a set of nodes (of size a constant times the total number of nodes), such that any two nodes in this set can communicate with each other privately and reliably, even a constant fraction of the edges in the graph are corrupted. More specifically, our goal is to construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on n nodes with maximum degree d, and a protocol for a secure channel, $\mathsf{SC}_{u,v}(m)$, with the following properties. Let the set of edges that are corrupted by an adversary be denoted by $\mathcal{T}_{\text{edges}} \subset \mathcal{E}, |\mathcal{T}_{\text{edges}}| \leq \alpha nd$. We shall show that there exists a set of nodes $\mathcal{P} \subseteq \mathcal{V}$, such that $|\mathcal{P}| \geq \gamma n$, and any two nodes $u, v \in \mathcal{P}$ can communicate using $\mathsf{SC}_{u,v}(m)$. As mentioned earlier, our main focus will be on building a public channel $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ that can be used by any two nodes $u, v \in \mathcal{P}$ to communicate authentically, but publicly, as this will be sufficient to obtain a protocol for secure channel using the techniques from Garay and Ostrovsky [15]. Our graph will have maximum degree $\mathcal{O}(n^{\epsilon})$, for arbitrary constants $0 < \epsilon < 1$, such that $|\mathcal{P}| \geq \gamma n$, for constant $0 < \gamma < 1$.

We begin this section by constructing such a public channel scheme on a graph of larger degree, $\mathcal{O}(\sqrt{n}\log n)$, and then show how to use that construction to obtain a scheme on a graph of maximum degree $\mathcal{O}(n^{\epsilon})$. Finally, we show how to obtain a secure channel between any two nodes in $\mathcal{P}$.

3.1 Public channels on $\mathcal{O}(\sqrt{n}\log n)$-degree graphs

We now show how to construct a graph of maximum degree $\mathcal{O}(\sqrt{n}\log n)$, and then present a protocol for a public channel between any two nodes $u, v \in \mathcal{P}$, tolerating a constant fraction of corrupted edges. For simplicity, we will assume that all messages in our protocols are binary. We remark that this restriction can be easily removed.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph on n nodes, d_v the degree of vertex $v \in \mathcal{V}$, and $\mathsf{Paths}_2(u, v)$ the set of all paths between any two vertices $u, v \in \mathcal{V}$ of length exactly 2. Let $\mathcal{G}$ satisfy the following two properties:

1. $\frac{\sqrt{n}\log n}{2} \leq d_v \leq 2\sqrt{n}\log n$ for all $v \in \mathcal{V}$; and
2. $|\mathsf{Paths}_2(u, v)| \geq \frac{\log^2 n}{2}$ for all $u, v \in \mathcal{V}$.

We will construct our public channel on any graph $\mathcal{G}$ satisfying the above properties. We first observe that such a graph is easy to construct probabilistically. Consider the Erdős-Renyi random graph $G(n, p)$, with $p = \frac{\log n}{\sqrt{n}}$; that is, construct the graph $\mathcal{G}$ such that there is an edge between every pair of nodes u and v, independently with probability $p = \frac{\log n}{\sqrt{n}}$ (for simplicity, we allow self-edges). Then, except with negligible (in n) probability, $G(n, p)$ satisfies the conditions that we require of graph $\mathcal{G}$. (For completeness, we provide the proof of this in Appendix B.) For brevity, sometimes we will denote this process by $\mathcal{G} \leftarrow G(n, p)$. We now present two lemmas for graph $\mathcal{G}$ satisfying the two properties above.

Lemma 1 *In graph $\mathcal{G}$, no edge participates in more than $4\sqrt{n}\log n$ paths of length exactly 2 ($\mathsf{Paths}_2(u,v)$) between any two vertices $u, v \in \mathcal{V}$.*

PROOF. Since paths in any $\mathsf{Paths}_2(u,v)$ are of length exactly 2, an edge (w,t) can participate only in paths of the form $\{x,y,z\}$ where either $(x,y) = (w,t)$ or $(y,z) = (w,t)$. From graph $\mathcal{G}$'s property 1 above, every vertex has degree $\leq 2\sqrt{n}\log n$. Hence, edge (w,t) participates in no more than $4\sqrt{n}\log n$ paths. □

Let $0 < \alpha_e, \alpha_n < 1$ be constants denoting the fraction of corrupt edges and corrupt nodes in the graph, respectively. Note that if we are able to design a protocol that can tolerate $\alpha_e\sqrt{n}(n-1)\log n + 2\alpha_n\sqrt{n}(n-1)\log n$ edge corruptions, then we will automatically get a protocol that can tolerate an α_e fraction of corrupt edges and an α_n fraction of corrupt nodes. Hence, let $\alpha = \alpha_e + 2\alpha_n$; we will construct a protocol that can tolerate an α fraction of corrupt edges (and no corrupt nodes). The next lemma bounds the number of nodes in $\mathcal{G}$ with poor connectivity.

Lemma 2 *Let $\mathcal{Y}_u$ denote the set of nodes v such that the fraction of paths in $\mathsf{Paths}_2(u,v)$ with no corrupt edges is $\leq \frac{1}{2}$. We say that a node $u \in \mathcal{V}$ is* doomed *if $|\mathcal{Y}_u| \geq \frac{n}{4}$. Then, in graph $\mathcal{G}$, at most $64\alpha n$ nodes are doomed.*

PROOF. Consider a particular node u. In order to make u doomed, an adversary must corrupt at least $\frac{\log^2 n}{4} \cdot \frac{n}{4}$ paths. Recall (Lemma 1) that every edge can participate in at most $4\sqrt{n}\log n$ paths. Since at most $\alpha\sqrt{n}(n-1)\log n$ edges can be corrupt, this contributes to at most $4\alpha n(n-1)\log^2 n$ corrupt paths. Hence, the total number of nodes that can be doomed is at most $64\alpha(n-1) < 64\alpha n$. □

The set of privileged nodes $\mathcal{P}$ in $\mathcal{G}$ will simply be the nodes that are not doomed. By Lemma 2 above, we have that $|\mathcal{P}| \geq (1 - 64\alpha)n = \gamma n$ (for some constant $0 < \gamma < 1$). We now present the construction of a public channel between any two nodes $u, v \in \mathcal{P}$:

$\underline{\mathsf{PC}^{\mathcal{G}}_{u,v}(m)}$

1. For every node $w \in \mathcal{V}$, u sends m over all paths in $\mathsf{Paths}_2(u,w)$.
2. Every node $w \in \mathcal{V}$, upon receiving m over the different paths, takes the majority of the values received, and sends this value to v over all paths in $\mathsf{Paths}_2(w,v)$.
3. For every w, v takes the majority value of all messages received over $\mathsf{Paths}_2(w,v)$ as the message received from w. Then, v takes the majority (over all w) of the received values as the value sent by u.

We now show that if nodes u and v are not doomed, then the protocol described above is a 0-public channel protocol.

Lemma 3 *Let $u, v \in \mathcal{P}$ (i.e., any two nodes in $\mathcal{G}$ that are not doomed), Then, after an execution of $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$, v outputs m with probability 1.*

PROOF. Since u is not doomed, when u sends m to every node w in the first step of the protocol, the number of nodes that will receive the value m correctly is more than $\frac{3n}{4}$ (since more than half of the paths from u to each of these nodes is not corrupted). Consider now one of these nodes w that receives the value correctly from u. Let us discount those nodes w that do not have a majority of uncorrupted paths to v. Since there can be only less than $\frac{n}{4}$ such nodes (as otherwise, v would be doomed), this leaves us with $> \frac{3n}{4} - \frac{n}{4} = \frac{n}{2}$ nodes. Each of these nodes have a majority of uncorrupted paths to v and

hence values sent by these nodes to v in step 2 will be received correctly by v. Hence, v will receive, out of the n values[6], more than $\frac{n}{2}$ values correctly, and hence when v takes the majority of values received in step 3 it will output m correctly. Note also, that our protocol is deterministic, and hence v will furthermore output m correctly with probability 1 when it takes the majority of values received. □

3.2 Public channels and secure channels on $\mathcal{O}(n^\epsilon)$-degree graphs

In this section we present our main technical result: we show how to recursively increase the number of nodes in graph $\mathcal{G}$ from the previous section, while not increasing its degree (asymptotically), and show how to implement a public channel on such graphs. We will do this in two steps. Let $\gamma = (1 - 64\alpha)$. We will first show the following:

Lemma 4 *Let $\mathcal{G}$ be a graph on n nodes with maximum degree d. Furthermore, let $\mathcal{G}$ be such that it admits a public channel protocol, $\mathsf{PC}^{\mathcal{G}}_{u,v}(\cdot)$, between any two nodes u and v from a set of size at least γn nodes even in the presence of αnd corrupt edges. Then, there exists a graph $\mathcal{G}'$ on n^2 nodes of maximum degree $2d$, such that $\mathcal{G}'$ admits a public channel protocol between any two nodes u and v from a set of size at least $\gamma^2 n^2$ nodes even in the presence of $\alpha^2 n^2 d$ corrupt edges.*

Later on, we will show how to apply the $\mathcal{G}'$ construction from $\mathcal{G}$ recursively to obtain the desired result on graphs of degree $\mathcal{O}(n^\epsilon)$.

Construction of $\mathcal{G}'$. We construct $\mathcal{G}'$ as follows. Take n copies of graph $\mathcal{G}$; we will call each copy a *cloud*, and denote them $C_1, \cdots, C_n$. Connect the n clouds using another copy of graph $\mathcal{G}$. We do this by connecting the i^{th} node in cloud C_j to the i^{th} node in cloud C_k by an edge, whenever there is an edge between C_j and C_k in $\mathcal{G}$. We will call such a collection of edges between cloud C_j and cloud C_k as a *cloud-edge*. Note that the maximum degree of any node in $\mathcal{G}'$ is $2d$.

We now describe how a node u in cloud C_j will communicate with a node v in cloud C_k—call this protocol $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$. To do this, we will first describe how two clouds that share a cloud-edge will communicate. Let every node $i \in C_j$ hold a value m_i as input (note that every node need not hold the same value m_i) and assume cloud C_j wishes to communicate with cloud C_k. We describe a protocol such that, assuming a large-enough fraction of nodes in C_j hold the same input value, say m, then at the end of this protocol's execution a large-enough fraction of nodes in cloud C_k will output m. We call this protocol $\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$. Let δ be such that $64\alpha n < \delta < (\gamma - 130\alpha)n$.

$\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$

1. For every node $1 \leq i \leq n$, the i^{th} node in C_j sends m to the i^{th} node in C_k through the edge connecting these two nodes.
2. The nodes in C_k execute a.e. differential agreement protocol $\mathsf{AE\text{-}DA}(n, 64\alpha n, \delta)$ using the value they received from their counterpart node in C_j as input. (Recall that the existence of protocol $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ between privileged nodes in $\mathcal{G}$ guarantees that one can construct an a.e. differential agreement protocol; see Section 4 for more details on constructing an a.e. agreement protocol on $\mathcal{G}$.)
3. Each node takes the output of protocol $\mathsf{AE\text{-}DA}(n, 64\alpha n, \delta)$ as its output in this protocol.

We are now ready to describe $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$:

[6] v receives values from all nodes, including u as well as v itself. This is because, in the first step, u sends m to all nodes (including u and v).

$\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$

1. u sends m to i for all nodes i in cloud C_j using $\mathsf{PC}^{\mathcal{G}}_{u,i}(m)$ from Section 3.1. The i^{th} node in C_j receives message m_i.
2. Clouds C_j and C_k now execute protocol $\mathsf{PC}^{\mathcal{G}}_{C_j,C_k}(m_i)$ over the graph $\mathcal{G}$ connecting the n clouds[7]. Whenever cloud C_w is supposed to send a message to C_z according to the protocol, they use protocol $\mathsf{CloudTransmit}_{C_w,C_z}(\cdot)$ over the cloud-edge connecting C_w and C_z.
3. Node $v \in C_k$ takes its output in the protocol $\mathsf{PC}^{\mathcal{G}}_{C_j,C_k}(m_i)$ as the value sent by u.

We prove the correctness of the transmission scheme above through a series of lemmas. At a high level, our proof goes as follows. We will call a cloud C_j as good if it does not have too many corrupt edges within it (that is, corrupt edges of the form (u, v) with both u and v in C_j); otherwise we will call the cloud, bad (Definition 4). We first show that an adversary cannot create too many bad clouds (Lemma 5). Next, we define what it means for a cloud-edge between two clouds C_j and C_k to be good in Definition 5 (informally, the cloud-edge is good if both C_j and C_k are good clouds and there are sufficient number of edges connecting privileged nodes in C_j and C_k). We then show that the adversary cannot create too many bad cloud-edges (Lemma 6). Next, we show that two good clouds can communicate reliably across a good cloud-edge (Lemma 7). Finally, we show that there exists a large set of clouds such that any two privileged nodes in any two clouds from this set, can communicate reliably (Lemma 8). From this, the proof of Lemma 4 readily follows.

Definition 4 *We say that a cloud C_j, $1 \leq j \leq n$, is* good *if $t_{C_j} \leq \alpha nd$. Otherwise, we say that the cloud is* bad.

Lemma 5 *The number of bad clouds in $\mathcal{G}'$ is at most $\lfloor \frac{t_c}{\alpha nd} \rfloor$.*

Proof. By Lemma 2, in order to make a cloud bad, $\mathcal{A}$ must corrupt more than αnd edges within the cloud. Since the total number of edges that the adversary corrupts within clouds is t_c, the total number of bad clouds is at most $\lfloor \frac{t_c}{\alpha nd} \rfloor$. □

Definition 5 *We say that a cloud-edge between two clouds C_j and C_k is* good *if*

1. *Both C_j and C_k are good, and*
2. *the number of nodes i such that the i^{th} node in C_j as well as the i^{th} node in C_k are both not doomed (in the graph $\mathcal{G}$ connecting the nodes in the respective cloud), while the edge connecting these two nodes is corrupt, is $\leq \alpha n$.*

Otherwise, we say that the cloud-edge is bad.

Lemma 6 *The number of bad cloud-edges is at most $\lfloor \frac{t_{ce}}{\alpha n} \rfloor$.*

Proof. In order to make a cloud-edge between two good clouds C_j and C_k bad, $\mathcal{A}$ must corrupt at least αn edges connecting them. Since the total number of corrupt inter-cloud edges is t_{ce}, the total number of bad cloud-edges is at most $\lfloor \frac{t_{ce}}{\alpha n} \rfloor$. □

Lemma 7 *Let C_j and C_k be two good clouds connected by a good cloud-edge. Further, let all nodes in C_j that are not doomed (in $\mathcal{G}$) hold value $m_i = m$ as input. Then, after executing protocol $\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$, all nodes in C_k that are not doomed (in $\mathcal{G}$) output m.*

[7] We again use m_i as the input argument, since the input values to nodes in C_j might be different.

PROOF. Since C_j is a good cloud, at most αnd edges in cloud C_j are corrupt. Hence, there are at least γn nodes in C_j that are not doomed, and that hold value m as input. When $\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$ is executed, all these nodes will send m across their edges to their counterpart nodes in C_k. Now, since a node that is not doomed in C_j may be connected to a node that is doomed in C_k, and some of the edges connecting two nodes that are not doomed may also be corrupted, the number of nodes that are not doomed in C_k that receive the value m correctly is at least $\gamma n - 64\alpha n - \alpha n = (\gamma - 65\alpha)n$.

Next, in step 2 of protocol $\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$, the nodes in C_k execute protocol $\mathsf{AE\text{-}DA}(n, 64\alpha n, \delta)$. In cloud C_k, we shall consider every doomed node as corrupt and every node that is not doomed as honest. By the property of the a.e. differential agreement protocol (cf. Definition 2), we have that whenever the honest nodes output a value, say, v, we have $c_v + \delta \geq c_{\bar{v}}$, where c_v denotes the number of honest nodes with initial value v (recall that, for simplicity, we are considering the binary case). From the calculation above, the number of nodes that are not doomed with the correct value m is at least $(\gamma - 65\alpha)n$. Hence, the number of nodes with the wrong value ($\bar{m}$) is at most 65αn. Since $65\alpha n + \delta < (\gamma - 65\alpha)n$ by the way δ is picked (in transmission scheme for $\mathcal{G}'$), we have that the nodes that are not doomed will never output $\bar{m}$ in protocol $\mathsf{AE\text{-}DA}(n, 64\alpha n, \delta)$ and hence all the nodes in C_k that are not doomed will output m after protocol $\mathsf{CloudTransmit}_{C_j,C_k}(m_i)$. □

Lemma 8 *There exists a set of clouds $\mathcal{P}_C$, $|\mathcal{P}_C| = \gamma n$, such that for all $C_j, C_k \in \mathcal{P}_C$ and for all nodes $u \in C_j$ and $v \in C_k$ that are not doomed, v always outputs m after protocol $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$.*

PROOF. Let us consider the graph $\mathcal{G}$ that connects the n clouds in graph $\mathcal{G}'$. By Lemma 5 we know that at most $\lfloor \frac{t_c}{\alpha nd} \rfloor$ clouds are bad. Furthermore, we know by Lemma 6 that in this graph $\mathcal{G}$ at most $\lfloor \frac{t_{ce}}{\alpha n} \rfloor$ cloud-edges are bad. Thus, in total, in graph $\mathcal{G}$ connecting the n clouds $\mathcal{A}$ can corrupt $\lfloor \frac{t_c}{\alpha nd} \rfloor$ clouds and $\lfloor \frac{t_{ce}}{\alpha n} \rfloor$ cloud-edges. Now, such an adversary can be perfectly simulated by an adversary that corrupts $\frac{t_c}{\alpha nd} \cdot d + \frac{t_{ce}}{\alpha n}$ cloud-edges in total. Since, the total number of edges that $\mathcal{A}$ is allowed to corrupt in $\mathcal{G}'$ is $t_c + t_{cex}$ and is bounded by $\alpha^2 n^2 d$, we get that this new adversary can corrupt at most αnd cloud-edges.

Now assume, as in Lemma 4, that the graph $\mathcal{G}$ that connects the n clouds, admits a public channel between any two clouds in a set of clouds of size at least γn (since at most αnd cloud-edges are corrupted), provided that any two clouds that are connected by a good cloud-edge can communicate reliably. Let this set of γn clouds be $\mathcal{P}_C$. By now applying Lemma 7 which precisely gives us the guarantee that any two clouds that are connected by a good cloud-edge can communicate reliably, the lemma follows. □

Public channels and secure channels on $\mathcal{O}(n^\epsilon)$-degree graphs. We now arrive at our main result by applying the construction of $\mathcal{G}'$ from $\mathcal{G}$ recursively, a constant number of times, beginning with graph $\mathcal{G} \leftarrow G(n, \sqrt{n}\log n)$. We first show:

Theorem 3 *For all constants $0 < \epsilon < 1$ and sufficiently large n, there exists a graph, call it $\mathcal{G}_{\text{main}}$, on n nodes, of degree at most $d_m = \mathcal{O}(n^\epsilon)$, admitting a 0-public channel protocol between any two nodes in a set of size at least $\gamma_m n$ (for some constant $0 < \gamma_m < 1$) even in the presence of an α_m fraction of edge corruptions (for some constant $0 < \alpha_m < 1$).*

PROOF. Applying Lemma 4 recursively, in conjunction with Theorem 5 (that states that we can obtain a.e. agreement on any graph that admits a public channel between the nodes in the privileged set), we get that there exists a graph $\mathcal{G}''$ on n^k (for constant $k \geq 1$) nodes, with maximum degree at most $d'' = 2^{k-1}\sqrt{n}\log n$ and such that $\mathcal{G}''$ admits a public channel between any two nodes in a set of size at least $\gamma^k n^k$ nodes even in the presence of $\alpha^k n^k d''$ corrupted edges (for constants $0 < \alpha, \gamma < 1$).

Setting $n_m = n^k$, we get that there exists a graph $\mathcal{G}_{main}$ on n_m nodes, with maximum degree at most $d_m = \frac{2^{k-1}}{k} n_m^{\frac{1}{2k}} \log n_m = \mathcal{O}(n^\epsilon)$ (for all constants $k \geq 1$ and appropriately chosen $0 < \epsilon < 1$) admitting a public channel between any two nodes in a set of size at least $\gamma_m n_m$ nodes even in the presence of $\alpha_m n_m d_m$ corruptions (for some constants $0 < \alpha_m, \gamma_n < 1$). □

As mentioned in the beginning of the section, the availability of public channels in $\mathcal{G}_{\text{main}}$ for a constant fraction of nodes, together with the techniques in [15], allow those nodes to communicate securely even in the presence of a constant fraction of corrupted edges.

Theorem 4 *For all constants $0 < \epsilon < 1$ and sufficiently large n, there exists a graph, call it $\mathcal{G}_{\text{main}}$, on n nodes, of degree at most $d_m = \mathcal{O}(n^\epsilon)$, admitting a $(0, \kappa_2)$-secure channel protocol (for negligible κ_2) between any two nodes in a set of size at least $\gamma_m n$ (for some constant $0 < \gamma_m < 1$) even in the presence of an α_m fraction of edge corruptions (for some constant $0 < \alpha_m < 1$).*

The proof of the theorem follows directly from Theorems 3 and 7 (from [15]). The reliability error, κ_2, comes from the SMT-PD (secure message transmission by public discussion) protocol used in [15], and it is unavoidable when a majority of paths between sender and receiver are corrupted (see, e.g., [14]), which is the case for privileged nodes in our low-degree setting.

4 A.E. Agreement and Secure Computation with Edge Corruptions on Low-Degree Networks

In this section, we describe our protocols and derive our conclusions for a.e. agreement and secure computation on $\mathcal{O}(n^\epsilon)$-degree networks, achieving security even in the presence of a constant fraction of edge corruptions.

We start by showing a.e. agreement on graph $\mathcal{G}$, of degree $\mathcal{O}(\sqrt{n} \log n)$ (Section 3.1). The nodes in $\mathcal{G}$ simply execute any standard Byzantine agreement protocol $\mathsf{BA}(n, 64\alpha n)$ (α defined below), executing $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ whenever node u is supposed to send message m to node v according to $\mathsf{BA}(n, 64\alpha n)$. Let Priv be the set of nodes in $\mathcal{V}$ that are not doomed at the end of the execution of the protocol. We then have the following.

Lemma 9 *Let $\alpha < \frac{1}{192}$. Then there exists a set $\mathcal{P} \subset \mathcal{V}$, $|\mathcal{P}| \geq \gamma n$ (for some constant $0 < \gamma < 1$), such that all nodes in $\mathcal{P}$ can reach agreement in the presence of an α fraction of edge corruptions in $\mathcal{G}$. Since the number of doomed nodes is $64\alpha n$, we can obtain agreement when $64\alpha < \frac{1}{3}$.*

Proof. Let $\mathcal{P} = \mathsf{Priv}$. From Lemma 3, we have that any two nodes $u, v \in \mathcal{P}$ can reliably communicate with each other using protocol $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ in the presence of an α fraction of edge corruptions in $\mathcal{G}$. Hence, nodes in $\mathcal{P}$ can reach agreement by executing any standard Byzantine agreement protocol $\mathsf{BA}(n, 64\alpha n)$ (that is, treating the number of doomed nodes as corrupt nodes—recall Lemma 2) and by simply executing $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ whenever u is supposed to send message m to v in BA, and provided that $64\alpha < \frac{1}{3}$ (necessary condition on the number of faulty players for Byzantine agreement in the fully connected setting, yielding $\alpha < \frac{1}{192}$). Hence, we have that $|\mathcal{P}| \geq (1 - 64\alpha)n = \gamma n$. □

Since graph $\mathcal{G}$ is constructed probabilistically, the following theorem establishes the soundness of our construction.

Theorem 5 *Except with negligible probability, graph $\mathcal{G} \leftarrow G(n, \frac{\log n}{\sqrt{n}})$ is a graph of maximum degree $d = 2\sqrt{n} \log n$ such that γn nodes in $\mathcal{V}$ can reach agreement even in the presence of an α fraction of edge corruptions (for constant $0 < \alpha < 1$).*

The proof of the theorem follows from Propositions 1 and 2, and Lemma 9. As a corollary, we get:

Corollary 1 *Except with negligible probability, graph $\mathcal{G} \leftarrow G(n, \frac{\log n}{\sqrt{n}})$ is a graph of maximum degree $d = 2\sqrt{n}\log n$ such that γn nodes in $\mathcal{V}$ can reach agreement even in the presence of an α_e fraction of edge corruptions and an α_n fraction of node corruptions (with $\alpha_e + 2\alpha_n \leq \alpha$).*

We now turn to our target degree: $\mathcal{O}(n^\epsilon)$, for constant $0 < \epsilon < 1$. We first describe an agreement protocol on graph $\mathcal{G}'$ (Section 3.2), with n^2 nodes and degree same as that of $\mathcal{G}$. Applying this recursively, we will obtain our agreement protocol on $\mathcal{G}_{\text{main}}$. To reach agreement on $\mathcal{G}'$, the n^2 nodes execute any standard agreement protocol $\mathsf{BA}(n^2, (1-\gamma^2)n^2)$ for a complete graph, replacing the sending of a message between any two nodes, say, u and v, by an invocation to protocol $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$. We now show that almost all nodes in $\mathcal{G}'$ can reach agreement.

Lemma 10 *There exists a set $\mathcal{P} \subset \mathcal{V}$, $|\mathcal{P}| \geq \gamma^2 n^2$ (for some constant $0 < \gamma < 1$), such that all nodes in $\mathcal{P}$ can reach agreement in the presence of an α^2 fraction of edge corruptions in $\mathcal{G}'$.*

Proof. We put a vertex u in $\mathcal{P}$ if the following conditions hold:

1. $u \in C_j$ such that $C_j \in \mathcal{P}_C$ (from Lemma 8), and
2. u is not doomed in graph $\mathcal{G}$ which connects C_j.

By Lemma 8, since at most αnd cloud-edges are bad, we get that $|\mathcal{P}_C| \geq \gamma n$. Furthermore, since all clouds in $\mathcal{P}_C$ are good, we have (from Definition 4) that for any given cloud in $\mathcal{P}_C$ at most αnd edges are corrupted, and therefore at least γn nodes in each cloud in $\mathcal{P}_C$ are not doomed. Hence, $|\mathcal{P}| \geq \gamma^2 n^2$.

Also by Lemma 8, any two nodes in $\mathcal{P}$, say, u and v, can communicate reliably using protocol $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$ in the presence of an α^2 fraction of edge corruptions in $\mathcal{G}'$. Hence, nodes in $\mathcal{P}$ can reach agreement by executing any standard Byzantine agreement protocol $\mathsf{BA}(n^2, (1-\gamma^2)n^2)$ for complete networks, and simply executing $\mathsf{PC}^{\mathcal{G}'}_{u,v}(m)$ whenever u is supposed to send message m to v in BA. □

Applying this protocol recursively (similarly to what is done in the proof of Theorem 3), we get our a.e. agreement protocol on $\mathcal{G}_{\text{main}}$, which is a graph on n nodes with maximum degree $\mathcal{O}(n^\epsilon)$ (for constant $0 < \epsilon < 1$). We now state our main conclusion, whose proof follows from Theorems 3 and 7.

Theorem 6 *For all sufficiently large n and all constant $0 < \epsilon < 1$, there exists a graph $\mathcal{G}_{main} = (\mathcal{V}, \mathcal{E})$ with maximum degree $\mathcal{O}(n^\epsilon)$, and a set of nodes $\mathcal{P} \subseteq \mathcal{V}$, with $|\mathcal{P}| \geq \mu n$ (for constant $0 < \mu < 1$) such that the nodes in $\mathcal{P}$ can execute a secure multi-party computation protocol (satisfying the security definition of* [15]*), even in the presence of of an α fraction of edge corruptions in $\mathcal{G}_{main}$ (for some constant $0 < \alpha < 1$).*

5 Summary and Open Problems

In this work, we considered the problem of a.e. secure computation in the presence of edge corruptions, and in particular focussed on the case when a constant fraction of the edges in the network can be corrupted. We presented graphs of degree $\mathcal{O}(n^\epsilon)$ (for arbitrary constant $0 < \epsilon < 1$) on which such protocols for a.e. secure computation are possible. Several natural questions remain.

While it is easy to show that one can obtain an a.e. secure computation protocol (amongst a constant fraction of the honest nodes) in the presence of a constant fraction of corrupted edges when the network is fully connected, whether one can obtain such a protocol when less than a $\frac{1}{3}^{\text{rd}}$ fraction of the edges are corrupted remains an interesting open problem[8].

[8] It can be easily seen that this fraction of corrupted edges is the best that one could hope for.

Another interesting question is whether we can obtain *polynomial-time* protocols for a.e. secure computation when a constant fraction of the edges are corrupted on constant-degree networks. Together, the results of Upfal [23] and Garay and Ostrovsky [15] show the existence of such a protocol. The running time of the parties, however, is exponential (in n).

References

[1] Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin. Secure computation without authentication. In *CRYPTO'05, 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, pages 361–377, 2005.

[2] Amos Beimel and Matthew Franklin. Reliable communication over partially authenticated networks. *Theoretical Computer Science*, 220(1):185 – 210, 1999.

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

[4] Piotr Berman, Krzysztof Diks, and Andrzej Pelc. Reliable broadcasting in logarithmic time with byzantine link failures. In *Journal of Algorithms 22 (1997)*, pages 199–211, 1997.

[5] Piotr Berman and Juan A. Garay. Fast consensus in networks of bounded degree (extended abstract). In *WDAG, Distributed Algorithms,* 4^{th} *International Workshop, Bari, Italy, September* 24-26*,* 1990, pages 321–333, 1990.

[6] Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Improved fault tolerance and secure computation on sparse networks. In *ICALP (2), 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, pages 249–260, 2010.

[7] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (abstract). In *STOC, Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing,* 2-4 *May* 1988*, Chicago, Illinois, USA*, pages 11–19, 1988.

[8] Bogdan Chlebus, Krzysztof Diks, and Andrzej Pelc. Reliable broadcasting in hypercubes with random link and node failures. In *Combinatorics, Probability and Computing 5 (1996)*, 1996.

[9] Krzysztof Diks and Andrzej Pelc. Reliable gossip schemes with random link failures. In *Proc. 28th Annual Allerton Conference on Communication, Control and Computing, Allerton, Illinois, October 1990*, pages 978–987, 1990.

[10] Danny Dolev. The byzantine generals strike again. *Journal of Algorithms*, 3(1):14 – 30, 1982.

[11] Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. In *FOCS,* 31^{st} *Annual Symposium on Foundations of Computer Science,* 22-24 *October* 1990*, St. Louis, Missouri, USA*, pages 36–45, 1990.

[12] Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. Fault tolerance in networks of bounded degree (preliminary version). In *STOC, Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing,* 28-30 *May* 1986*, Berkeley, California, USA*, pages 370–379, 1986.

[13] Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential consensus. In *PODC*, pages 211–220, 2003.

[14] Juan A. Garay, Clint Givens, and Rafail Ostrovsky. Secure message transmission by public discussion: A brief survey. In *Coding and Cryptology - Third International Workshop, IWCC 2011, Qingdao, China, May 30-June 3, 2011. Proceedings*, pages 126–141, 2011.

[15] Juan A. Garay and Rafail Ostrovsky. Almost-everywhere secure computation. In *EUROCRYPT,* 27^{th} *Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April* 13-17*,* 2008, pages 307–323, 2008.

[16] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC, Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing, 25-27 May 1987, New York City, NY, USA*, pages 218–229, 1987.

[17] S. Dov Gordon, Jonathan Katz, Ranjit Kumaresan, and Arkady Yerukhimovich. Authenticated broadcast with a partially compromised public-key infrastructure. In *Stabilization, Safety, and Security of Distributed Systems - 12th International Symposium, SSS 2010, New York, NY, USA, September 20-22, 2010. Proceedings*, pages 144–158, 2010.

[18] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Towards secure and scalable computation in peer-to-peer networks. In *FOCS,* 47^{th} *Annual IEEE Symposium on Foundations of Computer Science,* 21-24 *October* 2006*, Berkeley, California, USA*, pages 87–98, 2006.

[19] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[20] Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27:228–234, 1980.

[21] Andrzej Pelc. Reliable communication in networks with byzantine link failures. In *Networks 22 (1992)*, pages 441–459, 1992.

[22] Vasant Shanbhogue and Moti Yung. Distributed computing in asynchronous networks with byzantine edges. In *COCOON'96, Computing and Combinatorics, Second Annual International Conference, Hong Kong, June 17-19, 1996, Proceedings*, pages 352–360, 1996.

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

[24] Shu-Chin Wang, Yeh-Hao Chin, and Kuo-Qin Yan. Byzantine agreement in a generalized connected network. *IEEE Trans. Parallel Distrib. Syst.*, 6(4):420–427, 1995.

[25] Shu-Ching Wang and Kuo-Qin Yan. Revisiting fault diagnosis agreement in a new territory. *Operating Systems Review*, 38(2):41–61, 2004.

[26] Kuo-Qin Yan, Yeh-Hao Chin, and Shu-Ching Wang. Optimal agreement protocol in malicious faulty processors and faulty links. *IEEE Trans. Knowl. Data Eng.*, 4(3):266–280, 1992.

[27] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS, 23rd Annual Symposium on Foundations of Computer Science, 3-5 November 1982, Chicago, Illinois, USA*, pages 160–164, 1982.

[28] Vassilis Zikas, Sarah Hauser, and Ueli M. Maurer. Realistic failures in secure multi-party computation. In *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, pages 274–293, 2009.

A Additional A.E. Definitions and Protocols

Here we present the definition of a.e. computation (in the node-corruptions model) found in [15]. The definition of a.e. computation (in the edge-corruptions model) follows trivially from this.

Definition 6 [GO08] *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $|\mathcal{V}| = n$ be a network. Let $\mathcal{T}_{nodes} \subset \mathcal{V}$ be the set of corrupted nodes and let the privileged set of nodes be denoted by $\mathcal{P}$ (i.e., the nodes in $\mathcal{P}$ will be the nodes for which we will guarantee the secure computation). Let $|\mathcal{T}_{nodes}| \leq \alpha n$ (with α being such that $|\mathcal{P}| \geq \mu n$, for some constant $0 < \mu < 1$, for all sets $\mathcal{T}_{nodes}$). An n-player two-phase protocol Π is a* secure a.e. multi-party computation protocol *if for all sets of corrupted nodes $\mathcal{T}_{nodes}$ controlled by a single adversary $\mathcal{A}$, and any probabilistic polynomial-time computable function f, the following two conditions are satisfied at the end of the respective phases:*

Commitment phase: *During this phase, all players in $\mathcal{V}$ commit to their inputs.*

- Binding: *For all $P_i \in \mathcal{V}$, there is a uniquely defined value x_i^*; if $P_i \in \mathcal{P}$, then $x_i^* = x_i$.*
- Privacy: *For all players $P_i \in \mathcal{P}$, x_i^* is information-theoretically hidden.*

Computation phase:

- Correctness: *For all players $P_i \in \mathcal{P}$, $f(x_1^*, x_2^*, \cdots, x_n^*)$ is the value output by P_i.*
- Privacy: *Let $\overrightarrow{x_S^*}$ denote the vector of committed inputs corresponding to players in a given set S, and $\mathsf{PrivComp}_{\mathcal{A},\Pi}$ denote the following indistinguishability experiment:*
  1. *The adversary $\mathcal{A}$ specifies a set $\mathcal{P}$ and input vectors $\overrightarrow{x^*_{\mathcal{P},0}}$, $\overrightarrow{x^*_{\mathcal{P},1}}$ and $\overrightarrow{z^*_{\mathcal{V}\setminus\mathcal{P}}}$ such that $f(\overrightarrow{x^*_{\mathcal{P},0}}, \overrightarrow{z^*_{\mathcal{V}\setminus\mathcal{P}}}) = f(\overrightarrow{x^*_{\mathcal{P},1}}, \overrightarrow{z^*_{\mathcal{V}\setminus\mathcal{P}}})$.*
  2. *A random bit $b \leftarrow \{0,1\}$ is chosen (by the challenger). Then Π is executed on inputs $\overrightarrow{x^*_{\mathcal{P},b}}, \overrightarrow{z^*_{\mathcal{V}\setminus\mathcal{P}}}$.*
  3. *$\mathcal{A}$ outputs a bit b'.*
  4. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise; we write $\mathsf{PrivComp}_{\mathcal{A},\Pi} = 1$ if the output is 1.*

  Then it holds that
  $$\Pr[\mathsf{PrivComp}_{\mathcal{A},\Pi} = 1] \leq \frac{1}{2} + \epsilon,$$
  where ϵ is negligible in the input size.

We next state the result from [15]:

Theorem 7 ([15]—informal) *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with maximum degree d, and let $\mathcal{P} \subseteq \mathcal{V}$ be such that any two nodes $u, v \in \mathcal{P}$ can execute a κ-public channel protocol, call it $\mathsf{PC}^{\mathcal{G}}_{u,v}(m)$ (for negligible κ). Then, there exists a graph $\mathcal{G}'$ with maximum degree $2d$ such that there exists a set $\mathcal{P}'$ in $\mathcal{G}'$, with $|\mathcal{P}'| \geq \theta|\mathcal{P}|$ (for some constant $0 < \theta < 1$) such that*

1. *Every pair of nodes in $\mathcal{G}'$ can execute a (κ_1, κ_2)-secure channel protocol, call it $\mathsf{SC}_{u,v}(m)$ (for negligible κ_1, κ_2);*
2. *All nodes in $\mathcal{P}'$ can execute an a.e. MPC protocol (satisfying Definition 6).*

The above theorem, as stated, applies to the node-corruptions model. However, it is easy to see that one can get an analogous theorem in the edge-corruptions model as well, using the same techniques as in [15].

B Properties of Erdős-Renyi Random Graphs

We show below that the Erdős-Renyi random graph $G(n,p)$, with $p = \frac{\log n}{\sqrt{n}}$ satisfies the two properties we need with high probability.

Proposition 1 *Let d_v be the degree of vertex $v \in \mathcal{V}$. Then, in $G(n,p)$, except with probability $\mathsf{p_d} = 2ne^{-\frac{pn}{3}}$, we have $\frac{pn}{2} \leq d_v \leq 2pn$ for all $v \in \mathcal{V}$.*

Proof. Let d be the expected degree of a vertex $v \in \mathcal{V}$. Then $\mathsf{d} = pn$. By the Chernoff bound, $\Pr[d_v \leq \frac{pn}{2}] \leq e^{-2pn}$ and $\Pr[d_v \geq 2pn] \leq e^{-\frac{pn}{3}}$. So $\Pr[\frac{pn}{2} \leq d_v \leq 2pn] \leq 2e^{-\frac{pn}{3}}$. Applying the union bound, we get that except with probability $2ne^{-\frac{pn}{3}}$, $\frac{pn}{2} \leq d_v \leq 2pn$ for all $v \in \mathcal{V}$. □

Proposition 2 *Except with probability $\mathsf{p_a} = \frac{n^2}{e^{2p^2n}}$, we have $|\mathsf{Paths}_2|(u,v) \geq \frac{p^2n}{2}$ for all $u, v \in \mathcal{V}$ in graph $G(n,p)$.*

Proof. For any given vertex $w \in \mathcal{V}$, probability that $u - w - v$ is a path of length 2 in the graph is p^2. Hence, the expected number of paths between u and v is p^2n. Hence, by the Chernoff bound, we get that $\Pr[|\mathsf{Paths}_2(u,v)| \leq \frac{p^2n}{2}] \leq e^{-2p^2n}$. Applying the union bound, we get the required proposition. □

Note, that when $p = \frac{\log n}{\sqrt{n}}$, both $\mathsf{p_d}$ and $\mathsf{p_a}$ are negligible (in n) and hence properties 1 and 2 describe above hold in $G(n, \frac{\log n}{\sqrt{n}})$, except with negligible probability.