Almost-everywhere Secure Computation

JUAN A. GARAY* RAFAIL OSTROVSKY†

Abstract

Secure multi-party computation (MPC) is a central problem in cryptography. Unfortunately, it is well known that MPC is possible if and only if the underlying communication network has very large connectivity—specifically, $\Omega(t)$, where t is the number of potential corruptions in the network. This impossibility result renders existing MPC results far less applicable in practice, since most deployed networks have in fact a very small degree.

In this paper, we show how to circumvent this impossibility result and achieve meaningful security guarantees for graphs with small degree (such as expander graphs and several other topologies). In fact, the notion we introduce, which we call *almost-everywhere MPC*, building on the notion of almost-everywhere agreement due to Dwork, Peleg, Pippenger and Upfal, allows the degree of the network to be much smaller than the total number of allowed corruptions. In essence, our definition allows the adversary to *implicitly* wiretap some of the good nodes by corrupting sufficiently many nodes in the "neighborhood" of those nodes. We show protocols that satisfy our new definition, retaining both correctness and privacy for most nodes despite small connectivity, no matter how the adversary chooses his corruptions.

Instrumental in our constructions is a new model and protocol for the *secure message transmission* (SMT) problem, which we call *SMT by public discussion*, and which we use for the establishment of pairwise secure channels in limited connectivity networks.

Key words: Secure multi-party computation, almost-everywhere agreement, secure message transmission, expander graphs, bounded-degree networks.

1 Introduction

Secure multi-party computation (MPC) [Yao82, GMW87, BGW88, CCD88] is one of the most fundamental problems in cryptography. Simply put, in MPC n players jointly compute and obtain the value of an arbitrary n-ary polynomial-time computable function on their inputs, in such way that even if some fraction of the players are corrupted by a malicious adversary, the correct outputs as well as the privacy of the inputs of the uncorrupted (honest) players are guaranteed. After its formulation, MPC has been studied extensively, and many flavors with regards to, for example, the type of corruptions allowed by the adversary, its computational power, and definitions of security, have been considered in the literature.

In this paper our focus is on so-called unconditional, or information-theoretic, secure multi-party computation, as considered in [BGW88, CCD88], where no restrictions are placed on the computational power of the adversary. In this setting, assuming that players can communicate with every other player over not only dedicated but also private

*Bell Labs, Alcatel-Lucent, 600 Mountain Ave., Murray Hill, NJ 07974, USA. E-mail: `garay@research.bell-labs.com`.

†Department of Computer Science, University of California Los Angeles, Los Angeles, CA 90095-1596. E-mail: `rafail@cs.ucla.edu`.

communication channels, MPC is achievable as long as more than $\frac{2}{3}$ of the players remain uncorrupted. Moreover, this bound on the number of players is tight.

The above scenario assumes point-to-point private communication channel between every pair of players. In fact, with a few noted exceptions (more on this below), this is not only true for unconditional MPC, for most of the work on other models as well. Since reliable, let alone private communication is costly to achieve, fully connected networks sound like a costly proposition. Indeed, this question was posed by Dolev [Dol82], and later by Dolev, Dwork, Waarts and Yung [DDWY93], whose combined results show that in fact if there are t corrupted players, then $(2t+1)$-connectivity is both necessary and sufficient for unconditional MPC.

On the other hand, typically in practical networks most nodes have a very small connectivity, which is independent of the size of the network. In this paper, we show that meaningful statements about unconditional MPC can be made even if every node has small, even constant, connectivity. Clearly, in such a setting, we must give up on some of the "good" nodes, for example, in the case of such a node being totally "surrounded" in the network by corrupted nodes; still, we would like to be able to guarantee the security of a large fraction of uncorrupted nodes.

Our notion of "giving up" nodes originates from the notion of *almost-everywhere agreement* proposed by Dwork, Peleg, Pippenger and Upfal [DPPU86], who study how to achieve Byzantine agreement [PSL80, LSP82] in a limited connectivity setting. We build on the notions developed in [DPPU86], and in some sense, our central result can be viewed as a generalization of theirs. In essence, given a broadcast channel (implied by Byzantine agreement) and a constant number of uncorrupted paths among a large subset of nodes in the network, we show how to implement secure (i.e., reliable and private) channels among them, and thus achieve *almost-everywhere MPC*. "In essence" because apart from the construction, as opposed to just achieving the correctness property for a particular MPC instance (Byzantine agreement), substantial additional efforts are required in order to capture the privacy requirement of general MPC, particularly in the case of adaptive corruptions, and proving it correct.

One of the tools that we use to achieve the above transformation, which we call *secure message transmission by public discussion*, might be of independent of interest. As mentioned above, Dolev *et al.* [DDWY93] show a tight $(2t+1)$-connectivity bound for the (perfectly) secure message transmission (SMT) problem, of one node sending a message perfectly privately and correctly to another node over a network. That many channels, in fact, are needed to establish a public channel (no privacy concerns, only reliability). In our model, a public channel for large subsets of nodes can be constructed in a different way, and thus we are able to let the adversary corrupt all but one of the channels connecting those nodes, at the expense of a small error.

We stress that while in almost-everywhere MPC we give up the privacy and correctness of some of the nodes, we consider this a realistic assumption. Indeed, the requirements on a corrupted node are also given up in the classical model of MPC, and what we define here is a model where the adversary by corrupting some t notes, can potentially corrupt another set of t' nodes, which depends on t and the particular network. However, as long as the entire set is sufficiently small, i.e., $t + t' < \frac{n}{3}$, it is guaranteed that the rest of the nodes can achieve the requirements of secure multi-party computation, even on networks with bounded degree.

Related work. As already mentioned, our new notion is closely related to the concept of almost-everywhere agreement introduced in [DPPU86], and further explored in [Upf92, BG93]. We review some of the results in [Upf92] in Section 4.3, where we describe especial networks where almost-everywhere MPC can be achieved.

We also already mentioned the relation of our tool for secure channels to the problem of *perfectly secure message transmission* [DDWY93], which has been further studied in, e.g., [SA96, SNP04, ACH06, FFGV07]. In [FW98], Franklin and Wright take a different approach and study the necessary and sufficient conditions for secure message transmissions over multicast lines.

Our model for secure message transmission by public discussion is also related to the one used in *privacy amplification* and *secret key agreement* ([BBR88, BBCM95] and extensive number of follow-ups), where there also is an authentic public channel, and a private channel which the adversary is allowed to eavesdrop and/or tamper, depending on the various sub-models. Our problem can be viewed as a special instance of secret key agreement in the presence of severe tampering and transmission errors, but for a very specialized tampering "function," if we view all the channels as a combined channel.

In [FHM98], Fitzi, Hirt and Maurer considered a "hybrid" failure model where the adversary is allowed to maliciously corrupt some fraction of the players and in addition allowed to eavesdrop on some additional players. In our model, the potential additional eavesdropping (as well as violation of correctness) is defined structurally, given the graph topology and the location of the truly corrupted nodes.

Finally, the problem statement of almost-everywhere secure computation, as well as the overall approach are joint work with Shailesh Vaya [Vay06, GOVa, GOVb]. (See Acknowledgements for a more detailed account on this collaboration.) The work reported in [Vay07] follows this approach as well: adding privacy to networks that admit almost-everywhere agreement, using a protocol for achieving secret key agreement by public discussion (see above). Besides several other issues that would require technical improvement, a salient difference (shortcoming) in [Vay07] is the simulation-based security definition [Gol02], where the ideal-world adversary (the simulator), besides having access to the inputs of the corrupted players, is also given access to the inputs of the honest players that are given up; such a strong assumption makes the proof of security relatively straightforward, and gives the simulator an additional unfair advantage compared to the real-world execution. In contrast, in this paper we propose an indistinguishability-based security definition, known to be weaker than simulation-based, but meaningful. Further remarks on definitional issues are included in Sections 2.2 and 5.

Organization of the rest of the paper and our contributions. The rest of the paper is organized as follows. In Section 2 we present the model for and our definition of almost-everywhere MPC, together with other building blocks that will be used in our construction. In Section 3 we present the new model and protocol for SMT by public discussion, which we then use to obtain secure channels. Section 4 is dedicated to almost-everywhere MPC. First (Section 4.1), we define a class of graphs with special properties, which we call *almost-everywhere admissible graphs*. The literature on almost-everywhere agreement describes several such graphs; however, not all of them satisfy the requirements of our application. We show an efficient transformation of graphs G with degree d that allow almost-everywhere agreement into a graph of degree $O(d)$ that is almost-everywhere admissible. We then show (Section 4.2) how to construct protocols for almost-everywhere MPC on such graphs satisfying our definition, followed by concrete results for some specific networks (Section 4.3). We conclude in Section 5 with a summary and directions for future work.

2 Model, Definitions and Tools

In this paper we consider networks (graphs) $G = (V, E)$ that are *not* fully connected, as in [DPPU86, Upf92, BG93]. We let $|V| = n$. We will also refer to the nodes in V as "players," and to the edges in E as (communication) links or channels. The networks are synchronous, and the computation can be divided into rounds; in each round, a player may send a (possibly different) message on each of its incident links, and messages sent in one round are delivered before the next round. Up to t of the players can be actively corrupted by an adversary $\mathcal{A}$; we will use $T \subset V$, $|T| = t$ to denote the set of corrupted players, and sometimes we will refer to $\mathcal{A}$ as a t-adversary. We assume that $\mathcal{A}$ has unlimited computational power, and, furthermore, that $\mathcal{A}$ is *rushing*, meaning he can learn the messages sent by the uncorrupted players in each round before deciding on the messages of corrupted players for this round, and *adaptive*, meaning that information obtained from a set of corrupted players at a particular round can affect the choice of the next player(s) to be corrupted.

2.1 Building blocks

Our protocols for almost-everywhere MPC will be using several building blocks, including *almost-everywhere agreement* [DPPU86], *verifiable secret sharing* (VSS) [CGMA85], and a new primitive introduced here that we call *secure message transmission by public discussion* (Section 3).

Byzantine agreement and almost-everywhere agreement. We start with the standard definition of Byzantine agreement [LSP82, PSL80]. Here the network model is that of a fully connected network of pairwise authenticated channels.

Definition 2.1 *A protocol for parties $\{P_1, \ldots, P_n\}$, each holding an initial value v_i, is a* Byzantine agreement protocol *if the following conditions hold for any t-adversary:*

- AGREEMENT: *All honest parties output the same value.*
- VALIDITY: *If for all honest parties $v_i = v$, then all honest parties output v.* $\diamond$

It is known that $n > 3t$ is necessary and sufficient for Byzantine agreement [LSP82, PSL80], and there exist efficient (polynomial-time and round-optimal) deterministic protocols achieving it [GM98]. We will in fact rely on the related task called *broadcast*, where there is a distinguished player (the *sender*) P^* holding an initial value v. The agreement condition remains the same as above; validity requires that if the sender is honest, then all honest players output v. Broadcast easily reduces to Byzantine agreement, preserving the above bound (in the information-theoretic setting).

In [DPPU86], Dwork *et al.* relax the full (more specifically, $\Omega(t)$) connectivity requirement of the original Byzantine agreement formulation, proposing *almost-everywhere agreement* – "almost everywhere" because with partial connectivity agreement involving all the honest players is not possible and one must settle for agreement with *exceptions*, where some of the honest players are left out. Thus, in this context, the number of exceptions constitutes another relevant parameter for agreement protocols. Dwork *et al.* consider several classes of networks depending on their degree; the general approach, however, is to show how to simulate the transmission of a message between two players in such a way that if none of them belongs to a set, call it T^+, which includes the set of corrupted players (T) plus the left-out honest players, then the simulation is faithful. In turn, this makes it possible to simulate any Byzantine agreement protocol for fully connected networks (which does not rely on the privacy of the links) by treating players from T^+ as corrupted. We further review and apply some of the results in [DPPU86], as well as those in follow-up work [Upf92, BG93], in Section 4.3.

As before in the full connectivity case, an *almost-everywhere broadcast* protocol can be derived by having the sender first (attempt to) send his value to all other players using the transmission simulation scheme, and then having all players run the almost-everywhere agreement protocol; for an honest sender in T^+ the validity condition is not guaranteed, but agreement guarantees that all the players in $V - T^+$ will output the same value.

Verifiable secret sharing. Here the network model is that of a fully connected network of pairwise secure channels. One of the players is given a special role of being the *dealer* D. A VSS protocol consists of two phases: in the first phase, the dealer D distributes a secret s, while in the second, taking place possibly at a later time, the players cooperate in order to retrieve it. A more detailed specification is as follows:

Sharing phase: The dealer initially holds secret $s \in K$ where K is a finite field of sufficient size; at the end of the phase each player P_i holds some private information v_i.

Reconstruction phase: Each player P_i reveals his private information v_i. Then, on the revealed information v'_i (a corrupted player may reveal $v'_i \neq v_i$), a reconstruction function is applied in order to compute the secret, i.e., $s = Rec(v'_1, \cdots, v'_n)$.

The guarantees that are required from a VSS protocol are as follows.

Definition 2.2 *An n-player protocol is called a* (perfect) (n, t)-VSS protocol *if, for any t-adversary, the following condition holds:*

- PRIVACY: *If D is honest, then the adversary's view during the sharing phase reveals no information about s. More formally, the adversary's view is identically distributed under all different values of s.*
- CORRECTNESS: *If D is honest, then the reconstructed value is equal to the secret s.*
- COMMITMENT*: After the sharing phase, a unique value s^* is determined which will be reconstructed in the reconstruction phase; i.e., $s^* = Rec(v'_1, \cdots, v'_n)$ regardless of the information provided by the corrupted players.* ◇

It is known that $n > 3t$ is necessary and sufficient for VSS [BGW88], and there exist efficient protocols achieving it [GIKR02, FGG$^+$06][1]. If an (negligible) error is allowed, and additionally a broadcast channel is given, then $n > 2t$ suffices [RB89].

The last tool that we will be using, *secure message transmission by public discussion*, we treat separately in Section 3.

2.2 Almost-everywhere MPC

We now turn to the formulation of *almost-everywhere secure multi-party computation*. It follows from results in [Dol82, DDWY93] that in the type of networks that we are considering, it is not possible to establish secure channels between every pair of nodes, a known requirement for MPC. Indeed, depending on connectivity patterns, some nodes in V may have a majority (or even all) of the links coming from nodes controlled by $\mathcal{A}$. Thus, and as in [DPPU86, Upf92, BG93] in the context of almost-everywhere agreement, our approach to secure multi-party computation on such networks is also to "give up" on those nodes.

More formally, let $\mathcal{X} : 2^T \to 2^V$ be a function with the following properties:

1. $\mathcal{X}$ is monotically increasing, i.e., $T_1 \subset T_2$, implies $\mathcal{X}(T_1) \subset \mathcal{X}(T_2)$; and
2. $T \subset \mathcal{X}(T)$.

We say a protocol Π achieves X *secure multi-party computation* (X-MPC for short), where $X \stackrel{\text{def}}{=} \max_{T\subset V,|T|=t} (|\mathcal{X}(T)|)$, if for every subset T of nodes controlled by the t-adversary by the end of the protocol, there exists a set $W \subset V$ of uncorrupted players, $|W| \geq n - X$, such that all the players in W are able to perform secure multi-party computation. In the case of a fully connected network, $\mathcal{X}(T) = T$. Sometimes we will refer to the players in W as *privileged*, and to the players in $\mathcal{X}(T) - T$ as *doomed*.

Recall that the two main requirements in MPC are correctness of the output of the function being computed and privacy of the honest players' inputs. Prior work mentioned above for the limited connectivity setting was only concerned with the correctness of a function; given the additional privacy requirement of MPC, specifying what "to able to perform secure multi-party computation" means becomes more challenging. This gets further complicated

[1] In fact, these protocols additionally assume the availability of a broadcast channel, which can be implemented on the fully connected point-to-point network, since $n > 3t$.

by the fact that we are considering adaptive adversaries, which implies that the sets defined above might change (in particular, the set of given-up players will grow) during the execution of the protocol, and we would like, for any protocol, to state security guarantees for the honest players as these sets change.

The "commit-and-compute" paradigm. Typically, MPC protocols to compute a function on the inputs of the players $f(x_1, x_2, \cdots, x_n)$ (assuming for simplicity that all the players get the same result) tolerating active adversaries would start with the players executing a commitment phase, where the players' inputs are shared among the rest of the players, followed by a computation phase, followed by an output phase. For X-MPC, we make the commitment phase explicit and part of the definition, as this will allow us to precisely state the conditions on nodes in an unfavorable connectivity situation.

Definition 2.3 *Let $G = (V, E)$ be a network, and T, X and W, as defined above. An n-player two-phase protocol is an X* secure multi-party computation protocol *if for 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 V commit to their inputs.*

- BINDING: *For all $P_i \in V$, there is a uniquely defined value x_i^*; if $P_i \in W$, then $x_i^* = x_i$.*
- PRIVACY: *For all players $P_i \in W$, x_i^* is information-theoretically hidden.*

Computation phase:

- CORRECTNESS: *For all players $P_i \in W$, $f(x_1^*, x_2^*, \cdots, x_n^*)$ is the value output by P_i.*
- PRIVACY: *For conciseness, let $\overrightarrow{x_S^*}$ denote the vector of committed inputs corresponding to players in a given set S. If for all $\overrightarrow{x_W^*}$, $\overrightarrow{z_{\mathcal{X}(T_1)}^*}$, $\overrightarrow{y_W^*}$, $\overrightarrow{z_{\mathcal{X}(T_2)}^*}$ such that $\mathcal{X}(T_1) = \mathcal{X}(T_2)$ it holds that $f(\overrightarrow{x_W^*}, \overrightarrow{z_{\mathcal{X}(T_1)}^*}) = f(\overrightarrow{y_W^*}, \overrightarrow{z_{\mathcal{X}(T_2)}^*})$, then the adversary's views in the two protocol runs are statistically indistinguishable.* $\diamond$

We now make some remarks regarding our X-MPC definition.

Remark 2.4 In the adaptive-adversary setting, the sets T, $\mathcal{X}(T)$ and W might change dynamically during the execution of a protocol. Thus, we stress that in the definition above these sets are always defined with respect to the completion of a phase.

Remark 2.5 It is well known that an information-theoretic definition of privacy in terms of indistinguishability is weaker than a simulation-based counterpart. For example, consider a secure – according to our definition – multi-party protocol to compute $f(x)$ for a one-way permutation f, where x should remain hidden from all players. Information theoretically, the computation of $f(x)$ and the computation that just reveals x reveals the same amount of information to an infinitely powerful adversary; however, in the latter case, clearly x does not remain hidden. This example, due to Canetti, illustrates that one should not "mix" information-theoretic notions and computational notions, and that only suitable properties, such as those guaranteed by information theoretically secure MPC protocols [BGW88, CCD88, RB89], will remain secure according to our definition. See Section 5 for further remarks on simulation-based definitions for the X-MPC setting.

Before turning to protocols for X-MPC, in the next section we introduce the last tool that our protocols will be using, which will then allow for the establishment of secure channels in the limited connectivity setting, and which might be of independent interest.

3 Secure Message Transmission by Public Discussion

Let us first specify the (new) communication model that we are considering in this section; we will then relate this model to the X-MPC context. Here we consider just two players, $\mathcal{S}$ and $\mathcal{R}$, connected by a set of channels $\mathcal{C} = \{C_1, ..., C_N\}$, the contents of all but one of which can be eavesdropped and modified (in an arbitrary manner) by an adaptive, computationally unbounded adversary $\mathcal{A}$. Additionally, $\mathcal{S}$ and $\mathcal{R}$ have at their disposal an authentic and reliable public channel *Pub*.

The goal is to realize, using this communication model, a means for $\mathcal{S}$ to securely send messages to $\mathcal{R}$, a functionality known as *secure message transmission* (SMT) [DDWY93]. We will later be using this version of SMT in Section 4, to realize secure channels between nodes that are not directly connected, but with a connectivity pattern that can be abstracted out as the one considered in this section. First, we recall the properties of SMT.

Definition 3.1 *A protocol between $\mathcal{S}$ and $\mathcal{R}$ is a* secure message transmission protocol *if it transmits, using the network described above, a message from $\mathcal{S}$ to $\mathcal{R}$ such that the following two conditions are satisfied:*

- CORRECTNESS: *$\mathcal{R}$ learns the message except with probability ε.*
- PRIVACY: *$\mathcal{A}$ does not get any information about the message being transmitted.* $\diamond$

We now describe such protocol. Let $\mathcal{M}$ denote the space of (without loss of generality) q-bit messages. Let ℓ be such that:

1. $\ell \geq q$, and
2. $\ell > c \log \frac{N}{\varepsilon}$, for suitable c (specified later).

We also assume the availability of an error-correcting code tolerating a constant fraction of errors and constant blow-up; for concreteness, say up to $\frac{1}{4}$ of errors can be corrected, and that a q-bit message maps to a $12q$-bit codeword. Let *Enc* and *Dec* be the code's associated functions. The protocol, called PUB-SMT, is shown in Figure 1.

Theorem 3.2 *Protocol* PUB-SMT *is a four-round SMT protocol according to Definition 3.1, transmitting $O(\max(q, \log \frac{N}{\varepsilon}))$ bits on each of the N channels and $N \cdot O(\max(q, \log \frac{N}{\varepsilon}))$ bits over the public channel.*

Proof:

CORRECTNESS**:** Correctness would not hold if adversary $\mathcal{A}$ is able to corrupt Round 1 messages over any of the N channels and remain undetected. For each channel C_i, this would happen if $\mathcal{A}$ is able to corrupt more than 3ℓ bits. The probability of detecting one of these changes when one bit is revealed in Round 2 is at least $\frac{1}{5}$; thus, the probability that $\mathcal{A}$ remains undetected when 3ℓ bits are revealed is less than $(\frac{4}{5})^{3\ell}$. That's for each individual channel. The probability that $\mathcal{A}$ succeeds on any channel is $N(\frac{4}{5})^{3\ell}$. Setting $N(\frac{4}{5})^{3\ell} < \varepsilon$ yields $\ell > \frac{\log \frac{N}{\varepsilon}}{3 \log \frac{5}{4}} = O(\log \frac{N}{\varepsilon})$.

PRIVACY**:** Since according to the formulation of the problem, at least one channel (say, channel C_j) remains hidden from the adversary, this channel will always remain in set $\overline{\mathcal{C}}$, the set of non-faulty channels, and any message will be masked by this channel's bits. Hence, for all messages $M_1, M_2 \in \mathcal{M}$ and for all adversaries $\mathcal{A}$, the distribution of $\mathcal{A}$'s view when M_1 is transmitted is identical to the distribution when M_2 is transmitted.

The communication complexity is easily established by inspection. $\square$

The availability of the public channel makes it possible to tolerate a powerful adversary, who is allowed to eavesdrop and/or change the contents of all but one of the N channels. As mentioned at the beginning of the section,

Protocol PUB-SMT($\mathcal{S}$, $\mathcal{R}$, M, $\mathcal{C}$)

1. $\mathcal{S} \to \mathcal{R}$: Over each channel $C_i \in \mathcal{C}$, $\mathcal{S}$ sends to $\mathcal{R}$ uniformly chosen random bit string R_i, $|R_i| = 15\ell$. Let R'_i, $1 \le i \le N$, be the string received by $\mathcal{R}$ on channel C_i. $\mathcal{R}$ rejects all channels where $|R'_i| \neq 15\ell$.

2. $\mathcal{S} \to \mathcal{R}$: Let R^*_i denote R_i with 12ℓ randomly chosen positions replaced with "$*$." For each channel $C_i \in \mathcal{C}$, $\mathcal{S}$ sends R^*_i to $\mathcal{R}$ over *Pub*.

3. $\mathcal{R} \to \mathcal{S}$: For all channels $C_i \in \mathcal{C}$, if R^*_i and R'_i differ in any of the "opened" bits, $\mathcal{R}$ declares channel C_i as "faulty." I.e., $\mathcal{R}$ sends to $\mathcal{S}$ over *Pub* an N-bit string which identifies the faulty channels (say, as 0).

   Let $\overline{\mathcal{C}} = \{\overline{C_1}, \overline{C_2}, ..., \overline{C_L}\}$, $L \le N$, denote the set of remaining, non-faulty channels, and $\overline{R_i}$, $|\overline{R_i}| = 12\ell$, $1 \le i \le L$, denote the corresponding string of unopened bits; let $\overline{R'_i}$ be the corresponding string in $\mathcal{R}$'s possession.

4. $\mathcal{S} \to \mathcal{R}$: Let $|M| = q$ (if $|M| < q$, pad M accordingly). For $1 \le i \le L$, $\mathcal{S}$ chooses M_i such that $M = M_1 \oplus M_2 \oplus \cdots \oplus M_L$, and sends $S_i = \mathit{Enc}(M_i) \oplus \overline{R_i}$, $1 \le i \le L$, over *Pub*.

   For $1 \le i \le L$, $\mathcal{R}$ first computes $M'_i = \mathit{Dec}(S_i \oplus \overline{R'_i})$, and then $M' = M'_1 \oplus M'_2 \oplus \cdots \oplus M'_L$ to retrieve the message.

Figure 1: *Protocol for secure message transmission by public discussion.*

our application of SMT by public discussion to almost-everywhere MPC will be to provide secure channels between nodes that are not directly connected in the underlying network, and this section's channels will be instantiated by disjoint paths. Thus, in order to guarantee privacy not only with respect to the adversary, but also with respect to the other honest players, we will be requiring that at least *two*, instead of just one, of the channels (paths) remain untouched (i.e., the corresponding nodes remain uncorrupted) by the adversary. We show how to achieve this in the next section.

4 Almost-everywhere Secure Multi-Party Computation

In this section we first consider graphs with some special properties, which we call *almost-everywhere admissible graphs*, and which will constitute our candidate networks for almost-everywhere MPC. The literature on almost-everywhere agreement [DPPU86, Upf92, BG93] describes several classes of such graphs; however, not all of them satisfy the privacy requirement of almost-everywhere MPC mentioned above. First, given graphs with degree $d = d(n)$ that allow almost-everywhere agreement – more specifically, almost-everywhere broadcast, we show an explicit transformation to a new graph with degree $O(d)$ satisfying the requirement. We then show an explicit protocol for almost-everywhere MPC on this type of graphs, followed by instantiations of our results on concrete networks.

4.1 Almost-everywhere admissible graphs

First, a more general definition, to succintly express graphs whose sets of privileged nodes have a minimum of uncorrupted paths connectivity as well as a broadcast channel.

Definition 4.1 *Let $G_n = (V, E)$, $|V| = n$ be a graph, $T \subset V$, $|T| \leq t$, $\mathcal{X} : 2^T \to 2^V$ a monotically increasing function and $W = V - \mathcal{X}(T)$. We say that G_n is* almost-everywhere (i,t)-admissible *((i,t)-admissible for short) if the following two conditions are satisfied:*

1. *Nodes in W can successfully run almost-everywhere broadcast protocols with polynomial message complexity[2] ; and*
2. *there exists a computable map* SELECT-PATH(G_n, u, v) *outputting a set* PATHS(u,v) *such that*
   (a) *for all $u, v \in V$, $|\text{PATHS}(u,v)| \in O(\text{poly}(n))$;*
   (b) *for all $u, v \in W$, PATHS(u,v) contains at least i disjoint paths fully contained in W.*

Further, if both procedures in conditions 1 and 2 – the almost-everywhere broadcast protocol and map SELECT-PATH, respectively – are efficiently computable, we call G_n an *efficient* (i,t)-admissible graph.

$(2,t)$-admissible graphs are required by our application as, as mentioned before, two disjoint paths are needed in order to guarantee privacy with respect to intermediate nodes in the paths between nodes, even if those nodes are not corrupted. On the other hand, $(1,t)$-admissible graphs are of particular interest, as there exist constructions for graphs of bounded degree that yield large sets W, while tolerating sets T with the largest sizes, i.e., $|T| = O(n)$ ([Upf92]; see Section 4.3). Given that, we now show a transformation to turn $(1,t)$-admissible graphs into $(2,t)$-admissible, while (asymptotically) maintaining the original graphs' desired properties. Recall that we let $X = \max_{T \subset V, |T|=t}(|\mathcal{X}(T)|)$.

Lemma 4.2 *Let $G_n = (V, E)$ and $G'_{2n} = (V', E')$ both be $(1,t)$-admissible graphs according to Definition 4.1. Then, one can construct a $(2,t)$-admissible graph $G''_{2n} = (V'', E'')$ with subset W'' such that $|W''| \geq 2n - O(X'')$, where $X'' = X + X'$.*

Proof: Graph G''_{2n} is constructed as follows. First, take two copies of G_n, call them G_1 and G_2. Define $V'' = V_1 \bigcup V_2$ and add additional edges between the isomorphic vertices of G_1 and G_2. Note that the resulting graph so far has $2n$ vertices, and $2|E| + |V|$ edges.

Next, order the V'' vertex set in an arbitrary order and add to it all the *edges* from graph G'_{2n}; the resulting edge set is E'', with $|E''| = 2|E| + |V| + |E'|$. For convenience, call G_3 the instance of G'_{2n} applied to G''_{2n}.

We note that we allow *any* (but up to) t nodes to be corrupted in G''_{2n}. We account for every node corrupted in G''_{2n} as two corruptions: one in either (vertex set of) G_1 or G_2, and simultaneously as a corruption in G_3, since G_3 "reuses" the vertex sets of G_1 and G_2.

Now, for a subset of nodes $S_1 \subset V_1$, let $I(S_1)$ be the set of nodes in V_2 isomorphic to the nodes in S_1; define set $I(S_2)$ similarly. Let $T'' = T_1 \bigcup T_2 \bigcup T_3$, and let W_1 (respectively, W_2 and W_3) be the subset of nodes in G_1 (respectively, G_2, G_3) satisfying the premises of the lemma – G_n and G'_{2n} being $(1,t)$-admissible graphs.

Finally, let $W'' = (W_1 - I(T_2) \bigcup W_2 - I(T_1)) \bigcap W_3$. We now show that nodes in W'' can successfully run an almost-everywhere broadcast protocol, and that for all $u, v \in W''$, *two* disjoint paths fully contained in W'' exist connecting them. We have the following cases:

[2]By "successfully" we mean that for privileged senders (i.e., senders in W) the validity condition is satisfied (see Section 2).

1. $u, v \in V_1 \bigcap W''$: Almost-everywhere broadcast is obtained from G_3, specifically by running the protocol solely on G_3's edges.[3] One path of uncorrupted nodes between nodes u and v is given to us by the premises of the lemma with respect to G_1. The second path is as follows: 1) $u \to u'$, where $u' = I(u)$, 2) $u' \to v'$, where $v' \in I(v)$, and 3) $v' \to v$.

2. $u, v \in V_2 \bigcap W''$: Similar to case 1.

3. $u \in (V_1 \bigcap W'')$ and $v \in (V_2 \bigcap W'')$: Again, almost-everywhere broadcast is given to us by G_3. Let $u' = I(u)$ and $v' = I(v)$. The two paths containing nodes in W'' are as follows:
   (a) $u \to u'$; $u' \rightsquigarrow v$: a path in W'' assumed by the lemma for G_2;
   (b) $u \rightsquigarrow v'$, a path in W'' assumed by the lemma for G_1; $v' \to v$.

4. $u \in (V_2 \bigcap W'')$ and $v \in (V_1 \bigcap W'')$: Similar to case 3.

Let us now estimate the size of the subset W'':

$$\begin{aligned} |W''| &\geq |V''| - |\mathcal{X}(T_1)| - |I(T_2)| - |\mathcal{X}(T_2)| - |I(T_1)| - |\mathcal{X}'(T_3)| \\ &\geq |V''| - 4X - X' \qquad (\text{since } |I(T_i) \leq |\mathcal{X}(T_i)| \leq X, i = 1, 2) \\ &= 2n - O(X'') \end{aligned}$$

where $X'' = X + X'$. $\square$

In the next section we show how to construct X-MPC protocols on $(2, t)$-admissible graphs.

4.2 Almost-everywhere MPC protocols

We will be using several building blocks, including protocol PUB-SMT from Section 3, as well as protocols for unconditional VSS (see Section 2.1) and MPC [BGW88], the last two defined on a fully connected network.

However, first we would like to modify the specification of information-theoretically secure MPC (on a fully connected network and tolerating active adversaries) somewhat, so that it suits our purposes. Typically, the definition postulates an ideal model (equipped with a trusted third party) and compares it to the real model, demanding that in real life the adversary does not gain any advantage compared to what happens in the ideal model [Gol02]. In order to achieve this goal, all known implementations of MPC follow a "commit-and-compute" paradigm. It is convenient for us to recast those results in that paradigm. Recall that there are n players $P_1, ..., P_n$, each P_i holding a private value x_i, and wishing to jointly compute some function $f(x_1, \cdots, x_n)$. We call the modified protocol C&C-MPC, consisting of two phases:

Commit phase: Players commit to their inputs by acting as dealers in the sharing phase of a $(n, \frac{n}{3})$-VSS protocol – i.e., an unconditional, optimally resilient VSS protocol (e.g., [GIKR02, FGG+06]). (n executions of the protocol are run.) At the end of this phase, each player P_i holds a vector of n secret values (shares) $x_i^* = (v_i^1, ...v_i^n)$, one for each VSS invocation.

Computation phase: Players execute the original MPC protocol to compute an "augmented" function f^* defined as the composition of f and n invocations of function Rec:

$$f^*(x_1^*, x_2^*, ..., x_n^*) = f(Rec(v_1^1, v_2^1, ..., v_n^1), Rec(v_1^2, v_2^2, ..., v_n^2), ..., Rec(v_1^n, v_2^n, ..., v_n^n)),$$

where Rec is the reconstruction function of the $(n, \frac{n}{3})$-VSS protocol.

[3]This is important, as this property is not preserved under edge addition.

We stress that the Rec protocol is not executed "in the open," as one normally would, but as part of the MPC protocol. Thus, the results of each Rec invocation remain hidden within the MPC computation. Assuming the security of the VSS protocol, it is easy to see that C&C-MPC satisfies the same requirements as the original MPC protocol (correctness, privacy, and independence of inputs).

Having specified this version of MPC, our general approach to almost-everywhere MPC will be to have the players simulate C&C-MPC on the incomplete network, chosen with a suitable set of parameters, with the sending (and receiving) of messages on the secure channels substituted by invocations to protocol PUB-SMT, and invocations to the public channel (in PUB-SMT) and broadcast (VSS protocol) substituted by invocations to the almost-everywhere broadcast protocol. We give a more detailed description of the protocol and argue its security below.

Theorem 4.3 *Let $G_n = (V, E)$ be $(2, t)$-admissible graph, with T, X and W as in Definition 4.1, and such that $X < \frac{n}{3}$. Then there exists a protocol to achieve X secure multi-party computation against an adaptive, rushing t-adversary.*

Proof sketch: First, we specify the communication structure of the simulation. Each round of protocol C&C-MPC for complete networks is thought of as a "super-round." Each super-round has the same structure, with players taking turns[4] (in, say, lexicographic order) to perform the simulation of sends and receives required in the original round. More specifically, at the onset, each player P_i locally invokes procedure SELECT-PATH(G_n, P_i, P_j), the computable map given by G_n, to obtain set PATHS(P_i, P_j), for every P_j. Whenever P_i is required to send message m to P_j, P_i and P_j run PUB-SMT$(P_i, P_j, m, \text{PATHS}(P_i, P_j))$; invocations to the public channel by P_i (resp., P_j) in PUB-SMT are substituted by invocations to the almost-everywhere broadcast protocol, also given by G_n, with P_i (resp. P_j) acting as the sender. Similarly, invocations by P_i to broadcast in the $(n, \frac{n}{3})$-VSS protocol are replaced by an invocation to the almost-everywhere broadcast protocol with P_i as the sender.

Let $f(x_1, x_2, \cdots, x_n)$ be the function to be computed, where x_i is P_i's private input. Players now simulate the execution of the C&C-MPC protocol: first the commit phase – let $x_1^*, x_2^*, ..., x_n^*$ be the values held by the players at the end of this phase, followed by the computation of the "augmented" function f^*.

First, note that the communication structure of the simulation within the super-round (serialized, one player at a time, in turn one edge at a time) does not introduce any security vulnerability, as the original protocols are robust against rushing adversaries, who are allowed to learn the messages sent by the honest players in a round before deciding on the messages for the same round.

We now argue that the conditions of Definition 2.3 are satisfied. The premise of the theorem guarantees that $|W| > \frac{2n}{3}$. Thus, it follows from the (simulation of the) sharing phase of the $(n, \frac{n}{3})$-VSS protocol and the properties of almost-everywhere broadcast that for every player $P_i \in V$, there is a value x_i^* uniquely defined by its shares v_i^j, $1 \leq j \leq n$. For players in W in particular, $x_i^* = x_i$, since they are able to run almost-everywhere broadcast successfully (see Definition 4.1). We stress that players in $\mathcal{X}(T)$, not only the corrupted ones but also the doomed ones, might provide modified values or not be able to provide any input at all; regardless, they will be unique and well defined per the properties above. This gives the binding property of the commitment phase. The privacy of these values for players in W follows from the privacy condition of PUB-SMT, which again these players are able to execute successfully, and which guarantees that the views of the adversary (as well as of other honest players, since the graph is $(2, t)$-admissible) under the transmission of any two messages are identical.

Regarding the correctness of the computation phase, again since $|W| > \frac{2n}{3}$ and players in W can send private messages and simulate broadcast faithfully, they can carry on the reconstruction and the computation on the uniquely defined shared values in the commitment phase, following the protocol for fully connected MPC. Privacy of the computation phase follows from a hybrid argument and reduction to the privacy of the message transmission scheme.

[4]This for simplicity, and to avoid a more detailed analysis of possible interference.

(Recall Definition 2.3.) In a fully connected network, the condition of indistinguishable views for the adversary for all $\overrightarrow{x^*_{V-T}}$, $\overrightarrow{y^*_{V-T}}$, $\overrightarrow{z^*_T}$ such that the output of the function is the same, i.e., $f(\overrightarrow{x^*_{V-T}}, \overrightarrow{z^*_T}) = f(\overrightarrow{y^*_{V-T}}, \overrightarrow{z^*_T})$, is known to hold for an information-theoretically secure MPC protocol as long as the corrupted sets are the same [BGW88]. Thus, if the adversary is able to distinguish the two views with non-negligible advantage in the simulated execution, then there would be a particular super-round – in turn, player turn; in turn, message transmission – where the adversary can distinguish the two runs on G_n, but does not distinguish them in the fully connected network. This, in turn, contradicts the security of the message transmission protocol between two privileged players. □

4.3 Almost-everywhere MPC on classes of networks

In this section we enumerate several classes of networks where almost-everywhere MPC is possible, as a corollary of admissible graphs given in the almost-everywhere agreement literature.

Corollary 4.4 *For every $\epsilon > 0$ there exists a network $G_n = (V, E)$ of degree $O(n^\epsilon)$ and $|T| = O(n)$ on which $O(n)$-MPC is possible.*

The corollary follows from a recursive construction of networks of unbounded degree in [DPPU86] that yields $(2, O(n))$-admissible graphs with $X = O(n)$.

Corollary 4.5 *There exists a constant-degree network with $|T| = O(\frac{n}{\log n})$ on which $O(\frac{n}{\log n})$-MPC is possible.*

This network is constructed by taking a butterfly network, which constitutes a $(2, O(\frac{n}{\log n}))$-admissible graph, with $X = \tilde{O}(\frac{n}{\log n})$, and superimposing a 5-regular graph; this yields a regular graph of degree 8, on which a *compression* procedure can be run to "sharpen" the X term [DPPU86].

Finally, Upfal [Upf92] shows how to explicitly construct constant-degree expander graphs that yield $(1, O(n))$-admissible graphs – i.e., tolerating large (linear) number of corruptions. Applying the construction given in Lemma 4.2, we obtain:

Corollary 4.6 *There exist constant-degree networks with $|T| = O(n)$ on which $O(n)$-MPC is possible.*

The protocol achieving it, however, is not efficient (i.e., polynomial-time), as the resulting admissible graph is not efficient; specifically, the almost-everywhere broadcast component has polynomial message complexity but requires exponential computation.

5 Summary and Future Work

In this paper we introduced the notion of almost-everywhere secure multi-party computation for incompletely connected networks, and showed how to achieve meaningful security guarantees whenever possible. We proposed a definition for X-MPC, and a protocol satisfying it. We also gave concrete examples for specific networks, building on work from almost-everywhere agreement.

Regarding our definitional approach, *à la* [KKMO99], it is well known that simulation-based definitions of security are stronger than and preferable to indistinguishability-based ones. However, in the setting of almost-everywhere secure computation, the simulation-based approach encounters the following problem: it is not clear how to define, in a meaningful and network-independent way, the simulation and the adversarial view of the state of the doomed players (i.e., $z_{\mathcal{X}(T)}$), or indeed how to even deal with this dynamically growing set. It is clear that these nodes are not part of the nodes for which we guarantee a correct output, but it is not clear what view of these nodes an adversary gets. Indeed, for some of the doomed nodes the adversary could learn all the information and be able to change their inputs, while for others the adversary would only get partial control. We leave the refinement of and alternatives to our almost-everywhere MPC definition as a subject for future research. We stress though that in many situations, the security guarantees given by our approach are sufficient, especially if running information-theoretically secure protocols, such as [BGW88].

Regarding our new model for SMT by public discussion, it would be interesting to reduce the communication, in particular on the public channel (say, to sublinear in N), and provide some measure of optimality.

Finally, providing a polynomial-time protocol for almost-everywhere agreement – and thus for almost-everywhere MPC – on networks of bounded degree tolerating a constant fraction of corruptions remains an interesting open problem.

Acknowledgements

The problem statement of almost-everywhere secure computation, as well as the overall approach are joint work with Shailesh Vaya, as reflected in [Vay06, GOVa, GOVb][5]. However, the simulation-based approach to security that was originally considered in the three-author draft, and further developed in [Vay07], we (the current authors) eventually found unsatisfactory (see Sections 1, paragraph on related work, and 5 for a more technical discussion). Hence, we withdrew our names from that manuscript, and proceeded to completely change the protocol, as well as to work out a different model and corresponding proof of security—this is the work presented here. As a courtesy, Vaya was offered co-authorship on the current paper, provided however that the problematic approach be abandoned and not published as a separate work; he chose not to accept our offer.

[5]In [Vay06], Shailesh Vaya's thesis acknowledgement reads: *"Here, let me also thank Juan Garay who along with Rafail was an equal participant in developing the central ideas in this thesis."* It was a surprise to us that in [Vay07], which is based on [Vay06], proper credit is not given.

References

[ACH06] S. Agarwal, R. Cramer, and R.de Haan. Asymptotically optimal two-round perfectly secure message transmission. In *Advances in Cryptology–CRYPTO'06*. Springer-Verlag, 2006.

[BBCM95] C. H. Bennett, G. Brassard, C. Crèpeau, and U. Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41(6):1015–1923, November 1995.

[BBR88] C. H. Bennett, G. Brassard, and J. M. Robert. Privacy amplification by public discussion. *Siam Journal of Computing*, 17(2), April 1988.

[BG93] P. Berman and J. Garay. Fast consensus in networks of bounded degree. *Distributed Computing*, 2(7):62–73, 1993. Preliminary version in *WDAG'90*.

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. 20th Annual ACM Symposium of the Theory of Computation*, pages 1–10, May 1988.

[CCD88] D. Chaum, C. Crepeau, and I. Damgard. Multiparty unconditionally secure protocols. In *Proceedings 20th Annual Symposium on Theory of Computing, STOC*. Association for Computing Machinery, May 1988.

[CGMA85] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and simultaneous broadcast. In *Proc. 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 383–395, 1985.

[DDWY93] D. Dolev, C. Dwork, O. Waarts, and M. Young. Perfectly secure message transmission. *Journal of ACM*, 1(40):17–47, 1993.

[Dol82] D. Dolev. The byzantine generals strike again. *Journal of Algorithms*, 1(3):14–30, 1982.

[DPPU86] C. Dwork, D. Peleg, N. Pippinger, and E. Upfal. Fault tolerance in networks of bounded degree. In *Proc. 18th Annual Symposium on the Theory of Computing*, pages 370–379, 1986.

[FFGV07] M. Fitzi, M. Franklin, J. Garay, and S. Harsha Vardhan. Towards optimal and efficient perfectly secure message transmission. In *Proc. 4th Theory of Cryptography Conference (TCC '07)*, Lecture Notes in Computer Science, February 2007.

[FGG$^+$06] M. Fitzi, J. Garay, S. Gollakota, C. Pandu Rangan, and K. Srinathan. Round-optimal and efficient verifiable secret sharing. In *Proc. 3rd Theory of Cryptography Conference (TCC'06)*, Lecture Notes in Computer Science, pages 329–342. Springer Verlag, March 2006.

[FHM98] M. Fitzi, M. Hirt, and U. Maurer. Trading correctness for privacy in unconditional mpc. In *Advances in Cryptology–CRYPTO'98.*, Lecture Notes in Computer Science. Springer Verlag, August 1998.

[FW98] M. K. Franklin and R. N. Wright. Secure communications in minimal connectivity models. In *Advances in Cryptology–EUROCRYPT'98*, Lecture Notes in Computer Science, pages 346–360. Springer Verlag, June 1998.

[GIKR02] R. Gennaro, Y. Ishai, E. Kushilevitz, and T. Rabin. On 2-round secure multiparty computation. In *Advances in Cryptology–CRYPTO'02*, Lecture Notes in Computer Science. Springer Verlag, August 2002.

[GM98] J. Garay and Y. Moses. Fully polynomial Byzantine agreement for $n > 3t$ processors in $t + 1$ rounds. *SIAM J. Comput.*, 27(1):247–290, 1998. Preliminary version in *STOC '92.*

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for orotocols with honest majority. In *Proc. 19th Annual ACM Symposium on Theory of Computation*, pages 218–229, May 1987.

[Gol02] Oded Goldreich. Secure multi-party computation, final (incomplete) draft, version 1.4, 2002.

[GOVa] J. Garay, R. Ostrovsky, and S. Vaya. Almost-eveywhere secure computation. Rump session presentation at *TCC '07*, Feb. 2007.

[GOVb] J. Garay, R. Ostrovsky, and S. Vaya. Almost-eveywhere secure computation. Presentation at the 2007 Workshop on Cryptographic Protocols (*WCP '07*), Bertinoro, March 2007.

[KKMO99] J. Kilian, E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in private computations. *SIAM Journal on Computing*, 29, 1999.

[LSP82] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, pages 382–401, July 1982.

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM, JACM*, 27(2), April 1980.

[RB89] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st ACM Symposium on the Theory of Computing*, pages 73–85, 1989.

[SA96] H. Sayeed and H. Abu-Amara. Effecient perfectly secure message transmission in synchronous networks. *Information and Computation*, 1(126):53–61, 1996.

[SNP04] K. Srinathan, Arvind Narayanan, and C. Pandu Rangan. Optimal perfectly secure message transmission. In *Advances in Cryptology–CRYPTO'04*, volume 3152 of *Lecture Notes in Computer Science*, pages 545–561. Springer-Verlag, 2004.

[Upf92] E. Upfal. Tolerating linear number of faults in networks of bounded degree. In *Proc. 11th ACM Symposiyum on Principles of Distributed Computing*, pages 83–89, 1992.

[Vay06] S. Vaya. Almost-everywhere secure computation. December 2006. Ph.D. Thesis, University of California at Los Angeles, Los Angeles, California.

[Vay07] S. Vaya. Secure computation on incomplete networks. In Cryptology ePrint archive, Report 2007/346, September 2007.

[Yao82] A. Yao. Protocols for secure computation. In *Proc. 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.