Universally Composable Almost-Everywhere Secure Computation

Nishanth Chandran
Microsoft Research, Bangalore, India

Pouyan Forghani
Texas A&M University, College Station, TX, USA

Juan Garay
Texas A&M University, College Station, TX, USA

Rafail Ostrovsky
University of California, Los Angeles, CA, USA

Rutvik Patel
Texas A&M University, College Station, TX, USA

Vassilis Zikas
Purdue University, West Lafayette, IN, USA

Abstract

Most existing work on secure multi-party computation (MPC) ignores a key idiosyncrasy of modern communication networks, that there are a limited number of communication paths between any two nodes, many of which might even be corrupted. The problem becomes particularly acute in the information-theoretic setting, where the lack of trusted setups (and the cryptographic primitives they enable) makes communication over sparse networks more challenging. The work by Garay and Ostrovsky [EUROCRYPT'08] on *almost-everywhere MPC* (AE-MPC), introduced "best-possible security" properties for MPC over such incomplete networks, where necessarily some of the honest parties may be excluded from the computation.

In this work, we provide a universally composable definition of *almost-everywhere security*, which allows us to automatically and accurately capture the guarantees of AE-MPC (as well as AE-communication, the analogous "best-possible security" version of secure communication) in the Universal Composability (UC) framework of Canetti. Our results offer the first simulation-based treatment of this important but under-investigated problem, along with the first simulation-based proof of AE-MPC. To achieve that goal, we state and prove a general composition theorem, which makes precise the level or "quality" of AE-security that is obtained when a protocol's hybrids are replaced with almost-everywhere components.

2012 ACM Subject Classification Theory of computation → Cryptographic protocols; Security and privacy → Information-theoretic techniques; Security and privacy → Formal security models

Keywords and phrases Secure multi-party computation, universal composability, almost-everywhere secure computation, sparse graphs, secure message transmission

Digital Object Identifier 10.4230/LIPIcs.ITC.2022.14

Related Version *Full Version*: `https://eprint.iacr.org/2021/1398` [16]

Funding *Juan Garay*: Work partially supported by NSF grants CNS-2001082 and CNS-2055694. *Rafail Ostrovsky*: Supported in part by DARPA under Cooperative Agreement HR0011-20-2-0025, NSF grant CNS-2001096, US-Israel BSF grant 2015782, Cisco Research Award, Google Faculty Award, JP Morgan Faculty Award, IBM Faculty Research Award, Xerox Faculty Research Award, OKAWA Foundation Research Award, B. John Garrick Foundation Award, Teradata Research Award, Lockheed-Martin Research Award and Sunday Group. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, the Department of Defense, or the U.S. Government.

Vassilis Zikas: Research supported in part by NSF grants CNS-2055599 and CNS-2001096, BSF grant 2020277, and by Sunday Group.

1 Introduction

Secure multi-party computation (MPC) allows n parties communicating over a network to compute a function on their private inputs so that an adversary corrupting some of the parties can neither disrupt the computation (correctness) nor learn more than (what can be inferred from) the output of the function being computed (privacy).

Despite great progress on the problem since it was first introduced and proven feasible [44, 27, 8, 18] involving hundreds, if not thousands, of publications in cryptography and security, and, more recently, even implemented systems, the overwhelming majority of the solutions assume a *complete* communication network of either authenticated (aka reliable) or secure (both authenticated and private) point-to-point channels. In fact, with only a few exceptions, this is the case for both practical and theoretical works on MPC, and in particular for works on composable security of MPC—indeed, the latter almost exclusively assume a network that cannot be disconnected by the adversary. This creates a disconnect between the vast MPC literature and modern *ad-hoc* networks, such as the Internet, where the communication might be occurring over an incomplete graph, with some nodes even being routing nodes.

At first approximation, there are two situations that might present themselves in such an incomplete network: Either the adversary is able to disconnect the communication graph—by corrupting nodes whose edges are in cuts of the graph—or not. In the former case, it is known that if the parties do not share an authentication-enabling setup, such as a PKI, then the best that can be achieved is the so-called *secure computation without authentication* [6]: The adversary is able to break down the player set into connected components, so that parties in different connected components compute different instances of the function with inputs from the component—and all other inputs chosen by the adversary, and potentially different for each component. Even this weak form of security is only achievable for computationally bounded adversaries; if one is after information-theoretic (aka unconditional) security, where the adversary is unbounded, then the above guarantee is too much to ask for.

Notwithstanding, even in the latter case, where the adversary cannot disconnect the network, the situation is trickier than one might expect. Indeed, if a PKI-like setup is not assumed[1] then it is known that secure communication between any two parties requires the existence of $O(n)$ paths among them (known to or discoverable by the receiver), the majority of which must remain uncorrupted. This is the well-known *secure message transmission* (SMT) problem [20]. The result holds even for the *reliable message transmission* (RMT) problem, in which only correctness is required.

That leads to the following natural question: What is the "best-possible" MPC security we can obtain in such a situation where SMT cannot be in general guaranteed? Towards answering this question, Garay and Ostrovsky [25] introduced the properties of *almost-everywhere MPC* (AE-MPC), which extended the concept of AE reliable communication previously studied by Dwork *et al.* [21]. In a nutshell, the paradigm of *almost-everywhere security* (AE-security) recognizes that when even all-to-all SMT is not possible (and only AE-SMT is available), then inevitably there will be uncorrupted parties for which we are

[1] A PKI trivializes this case as a complete graph can be built by gossip (i.e., flooding) of signed messages.

unable to offer the security guarantees that honest parties enjoy in MPC (privacy, correctness, etc). The core mission is then to minimize the number of such left-out (aka *doomed*) parties in an AE-secure construction, while tolerating the maximum number of corruptions.

However, despite a number of elegant combinatorial arguments to achieve the above goal, the security definition used by these constructions has not caught up with the state of the art in MPC security. In particular, to the best of our knowledge, there exists no simulation-based treatment of AE-security. This means that one cannot directly compose the elegant constructions of AE-secure primitives into a higher level protocol. For example, one would hope to be able to prove that running a standard MPC protocol over an AE-SMT network yields an AE-MPC protocol which does not leave more doomed parties than the underlying AE-SMT construction. Given the state of the art, such a modular statement would be impossible, and one would need to prove AE-MPC security from scratch. Instead, a simulation-based treatment in one of the composable security frameworks would inherit a modular composition theorem making such statements tractable and simpler.

This work's main goal is to derive such a treatment in the Universal Composability (UC) framework of Canetti [13]. A major challenge, which we tackle, is to obtain a generic definition of AE-security which can be applied to any type of functionality and captures both AE-communication and AE-computation, two primitives whose treatment has been very different. In fact, we achieve this goal by introducing a generic, composition-preserving transformation from a secure variant of a functionality to its AE-secure counterpart. We show that the derived AE-secure functionalities for secure communication (AE-RMT and AE-SMT) and for secure MPC (AE-MPC): (1) preserve all the desired properties of the previous definitions, and (2) are realized by (straightforward UC adaptations of) classical AE-secure protocols. Since our treatment preserves composability of the (AE-)security statements, we obtain, as a simple corollary, the first simulation-based proof of AE-MPC.

In passing, we note that although we adopt the language of UC, our definitional framework is generic and can be applied to any of the main-stream composable security frameworks for cryptographic protocols [3, 15, 37, 31, 11, 4]. Before providing more details on our results, we first provide some necessary literature background.

1.1 Related Work

The origins of the "almost-everywhere" (AE) notion can be traced back to the work of Dwork *et al.* [21], who considered the task of Byzantine agreement [39, 36] over sparse communication networks. In such networks, correctness cannot be guaranteed for all honest parties, since for example the adversary can isolate a node by corrupting all its neighbors. Thus, some honest parties must be given up, and correctness is guaranteed only almost-everywhere, i.e., only for the remaining honest parties. The AE notion can be applied to other distributing computing tasks as well: Given a set of parties P and an adversary who corrupts $T \subseteq P$, the parties in some set $D \subseteq P - T$ (D for "doomed") are considered abandoned and the correctness conditions of the task are only guaranteed for the parties in $W = P - T - D$ (called "privileged"). Note that both D and W are functions of T as well as of the underlying protocol and graph. The number of doomed parties thus becomes another parameter to the problem, and the goal is to construct a low-degree network (ideally of constant degree) admitting a protocol that tolerates a large number t of corruptions (ideally, a constant fraction) while dooming as few nodes as possible (ideally $O(t)$ for constant-degree networks).

Returning to the problem of Byzantine agreement, Dolev [19] showed that it requires connectivity at least $2t + 1$ to solve, which implies that every node in the network must have degree $\Omega(t)$. Given this high connectivity requirement, Dwork *et al.* [21] proposed the notion

of *AE agreement*, in which the agreement and validity properties are guaranteed only for the privileged parties. They showed how to simulate, over an incomplete network, an agreement protocol designed for a complete network by replacing the point-to-point communication with a transmission scheme that works over multiple paths between any two nodes. Thus, they reduced AE agreement to AE reliable message transmission (AE-RMT), which guarantees that any two privileged nodes can communicate perfectly reliably.

Dwork *et al.* gave a number of constructions achieving AE-RMT with various combinations of parameters; the most important is a constant-degree graph admitting an AE-RMT scheme tolerating $t = O(n/\log n)$ corruptions while dooming $O(t)$ nodes. Several follow-up works have obtained improved parameters for AE-RMT (and thus also for AE agreement). Upfal [43] gave a transmission scheme tolerating $t = O(n)$ corruptions and dooming $O(t)$ nodes in a network of constant degree, which is the optimal set of parameters, but the protocol is inefficient. Chandran *et al.* [17] proposed a scheme tolerating $t = O(n)$ corruptions and dooming $O(t/\log n)$ nodes in a network of polylogarithmic degree. Most recently, Jayanti *et al.* [32] used the probabilistic method to show the existence of a logarithmic-degree graph admitting an AE-RMT scheme with the same parameters, strictly improving the [17] result.

Due to the results in [19, 20], standard MPC (guaranteeing correctness and privacy for all honest parties) is possible only in networks with connectivity at least $2t + 1$. To circumvent this high-connectivity requirement and still obtain a meaningful notion of (property-based) MPC over sparse networks, Garay and Ostrovsky [25] introduced the notion of AE-MPC, which guarantees correctness and privacy only for the privileged parties[2].

"Regular" information-theoretic MPC (i.e., MPC over a complete network) requires $t < n/3$ [8, 18]. In the AE setting, the effect of dooming nodes is equivalent to letting the adversary corrupt some additional t' nodes (which are doomed) by requesting the corruption of t nodes (which are actually corrupted). As shown by Garay and Ostrovsky, AE-MPC in the information-theoretic setting can be achieved when $t + t' < n/3$. Their approach resembles that of Dwork *et al.* [21] for simulating a protocol meant for a complete network, but to replace point-to-point secure channels, they introduced a new model for the existing (perfectly) SMT problem termed *secure message transmission by public discussion* (SMT-PD).

The original SMT problem [20] considers two honest parties, a sender S and a receiver R, connected by n disjoint "wires". The task is for S to send a message to R in the presence of a computationally unbounded adversary $\mathcal{A}$ who can adaptively corrupt up to t of the wires. SMT requires that the message be conveyed perfectly reliably to R, and also that no information about the message leaks to $\mathcal{A}$. While the simpler task of RMT (with no secrecy requirement) can be achieved for $t < n/2$ by simply duplicating the message over all wires, Dolev *et al.* [20] showed that SMT is also possible if and only if $t < n/2$. Since their initial feasibility result, much more efficient protocols have been introduced [40, 42, 1, 35, 28, 41].

The SMT-PD model overcomes the necessity of $2t + 1$ wires in SMT by allowing access to an authentic and reliable public channel. Given such a channel (which can be constructed using, e.g., a broadcast protocol), Garay and Ostrovsky [25] gave a protocol that is secure as long as one wire remains honest, at the cost of a small error. To use their SMT-PD protocol over sparse networks (in effect achieving AE-SMT), the wires are replaced by multiple paths between a pair of nodes and the public channel is replaced by AE broadcast. Garay and Ostrovsky showed how to construct graphs that admit SMT-PD from any of the networks in the AE agreement literature, with asymptotically preserved parameters. Finally, they showed

[2] Technically, they considered the related task of *secure function evaluation* (SFE). We do the same, although for consistency we still refer to the functionality that we realize as AE-MPC.

how to "compile" a standard information-theoretic MPC protocol into an AE-MPC protocol over any such graph, dooming the same number of parties as the underlying network.

To reiterate, all the above constructions are shown secure in a property-based manner. Other related notions include hybrid failure models (e.g., [26, 23]) and the model of Alon *et al.* [2]. In the AE setting, adversarial corruptions also have the effect of indirectly influencing the behavior of some of the honest parties (those who are doomed), but in our model this other type of failure is defined structurally, based on the graph and the set of corruptions. Also related is the work by King and Saia [34] and follow-ups (e.g., [9, 10]), who considered full (not AE) Byzantine agreement over complete networks, but without all-to-all communication.

1.2 Overview of Our Results

In this work we put forth the first composable (simulation-based) definition and treatment of AE-security. In particular, we devise a definition in Canetti's UC framework [13] and prove that the (UC adaptation of) existing AE-secure communication/computation protocols achieve this definition. We emphasize that all of our constructions tolerate adaptive corruptions.

There are several challenges associated with such a task. First, as should be evident from the above discussion, the related literature—from RMT/SMT, to Byzantine agreement, to MPC, and even their AE counterparts—treats the underlying network in different ways: e.g., in MPC, the network is typically a complete graph of point-to-point channels, whereas the literature on (AE-)RMT assumes multiple paths (wires or indirect paths) between two parties. Thus, to derive a formulation general enough to capture the security of the above constructions, one first needs to develop a unified approach. Towards this goal, we adopt the graph model as a basis for all these protocols, and express the wires in the RMT/SMT literature as a simple graph which for each wire includes a path going through a unique "wire-party." We can then model corrupted wires as standard (party) corruptions in UC.

The second, and more thorny challenge is regarding the (simulation of) doomed parties. Recall that those are parties that due to their poor connectivity (which might be the result of the sparsity of the graph and the corruption choices of the adversary) cannot enjoy the security guarantees that the protocol is designed to offer to honest parties (e.g., correctness and privacy for an MPC protocol). A strawman approach would be to capture those parties plainly as corrupted. This, however, is problematic in several ways: First, corrupted parties lose their security guarantees as soon as they become corrupted, unlike doomed parties who might, at the adversary's discretion, still be allowed some level of security. In particular, the real-world adversary might allow those parties to receive their outputs, which would mean that in the ideal world, the simulator would also need to allow them to produce an output on their output tape, which is not allowed by the UC corruption mechanism.

An attempt to fix the above issue would be to define weaker corruption types corresponding to the flexible guarantees offered to the doomed parties. This, however, is also problematic, as corruptions in UC are by default known to (and declared by) the adversary/environment, whereas the actual identities of doomed parties are not, and depend on the behavior of the adversary (not just the identities of malicious parties). In particular, an adversary following, e.g., a random strategy might not even be aware who is becoming doomed by this strategy.

A third attempt would be to completely change the corruption mechanism of UC so that certain corruptions are not to be declared by the environment. But this would immediately invalidate the composition theorem, which defeats the purpose of using UC in the first place.

It might seem like we are in a deadlock, but the second attempt above is the one that breaks through. In particular, we observe that although the adversary might not include in its view the identities of the doomed parties, its behavior still defines these identities and the

corresponding guarantees they receive. This is similar to how inputs of corrupted parties are treated in standard UC security: It is the job of the simulator to extract them from the adversary and hand them over to the functionality.

Accordingly, instead of modifying the foundations of UC, we define a class of functionalities which take requests from their adversary (simulator) to mark parties as doomed, and allow the simulator to use these parties as if they were corrupted, but without declaring them as corrupted to the framework and without grounding their input/output tapes (e.g., the simulator might still instruct this new functionality to deliver output for doomed parties). In fact, this is done in a black-box manner, by *wrapping* an underlying (non-AE) functionality.

In more detail, our AE wrapper builds the entire infrastructure of UC around it (including a fake corruption directory), and whenever a doom request comes in, the wrapper pretends towards its wrapped functionality to be an adversary that corrupts this party. This way, the party remains honest as far as the UC experiment is concerned, but the wrapper now has the ability to give full control over this party to the simulator it interacts with.

The final piece of the puzzle is capturing different ratios of corrupted vs doomed parties while making a composable statement. Here we use an idea inspired by [5]: We parameterize the wrapper by the set of all allowable corruption/doom patterns, and make sure that any request outside this set is ignored. For example, to prove security of AE-MPC with $t < \alpha n$ corruptions and $d < \beta n$ doomed parties, we can parameterize the wrapper with the pair (α, β) and ignore requests of simulators that do not respect the above requirements.

In fact, to allow for the tightest possible results that accurately translate non-threshold corruption/doom patterns (the types of results we get by using structural properties of the underlying graph), we draw inspiration from the mixed general adversary literature [29, 7]. That is, we parameterize the wrapper with a corruption/doom structure ("doom structure" for short), consisting of all allowed pairs (C, D) where parties in D can be doomed simultaneously to parties in C being corrupted. As is common in the general adversary literature, such a structure might be exponentially large. Although this is not an issue in our definition, we note that all our concrete instantiations consider structures that have a polynomial representation.

We then apply our definitional framework to capture known AE-secure constructions, as well as (simulation-based) AE-MPC. Next, we describe our results in greater detail.

Almost-Everywhere RMT and SMT. We start in Section 3 by modeling the tasks of RMT and SMT (with a dedicated sender and receiver connected by a number of corruptible wires). As part of this, we show how these primitives, which have classically only been considered for an honest majority of wires, can be captured so that their security is defined independently of the number of corrupted wires. To apply a unified treatment, we cast the problem by modeling each wire with a (corruptible) dummy party called a "wire-party," which simply relays messages between S and R. In Section 3.1, we confirm that classical RMT/SMT protocols [20] are UC-secure (in the ordinary, non-AE sense) in our model against corrupted minorities of wire-parties. To handle corrupted majorities (and more generally to capture AE-security), in Section 3.2 we introduce an *AE wrapper functionality* that is parameterized by a doom structure as defined above. We can then state the AE-security of RMT/SMT protocols, by using a simple doom structure like the one that allows dooming S or R when a majority of the wire-parties are corrupted. We finish up in Section 3.3 with a universally composable treatment of the SMT-PD problem [25]. We model the public channel using access to the same functionality that we use to capture RMT security. Looking ahead, we will need SMT-PD when we want to elevate RMT to SMT over some classes of sparse graphs.

Almost-Everywhere *Remote* RMT and SMT. In Section 4, we consider the more complicated case where an incomplete graph connects several parties and yet all-to-all communication is desired. Interestingly, we show that the same wrapper from Section 3.2, which allowed for the simulation-based treatment of tasks like RMT and SMT even against corrupted majorities of wires, can also be used to model AE-security of the all-to-all versions of those tasks. In particular, in Section 4.1 we use the same ideal functionalities and wrapper (with more complex doom structures) from Section 3 to provide the first universally composable treatment of (AE) reliable communication over the sparse graphs constructed in [21, 43, 17, 32], which we refer to as AE *remote* RMT. In Section 4.2, we extend our treatment to AE *remote* SMT for all of these graphs. First, we show that a perfect SMT protocol from [20] can be adapted to realize perfectly secure AE-SMT over a class of sparse graphs constructed in [21]. In general, the same approach cannot be directly extended to achieve privacy for other graphs. To overcome this, we adapt an SMT-PD protocol from [25] to realize AE-SMT over the graphs in [43, 17, 32], at the cost of obtaining only statistical UC security.

Almost-Everywhere Secure Computation. Lastly, we study the composability of AE-security guarantees, with the ultimate goal of realizing AE-MPC. In Section 5.1, we prove a general composition theorem, which makes precise the level or "quality" of AE-security (as captured in a doom structure) that is obtained when a protocol's hybrids are replaced with AE counterparts. We emphasize that this *AE compiler* need not replace all of the hybrids with AE-wrapped versions using the *same* doom structure; thus, we are able to explain, for example, what happens when a protocol uses subprotocols that provide differing levels of AE-security. Our composition theorem applies even to protocols that already carry some level of AE-security, and therefore the compiled protocol can easily be composed with higher-level protocols. As a simple corollary, we show that a protocol achieving standard (non-AE) security using a single hybrid can be compiled into an AE-secure protocol while preserving the doom structure associated with the wrapped hybrid. In Section 5.2, we apply this corollary to obtain the first simulation-based proof of AE-MPC, over any of the classes of sparse graphs considered in the AE agreement literature [21, 43, 17, 32]. Depending on which class of sparse graphs is used, we obtain either perfect or statistical UC security.

Next, we review some preliminaries. Due to space limitations, some of the functionalities, protocols, and proofs are presented in the appendix or in the full version of the paper [16].

2 Preliminaries

2.1 UC Basics

We briefly summarize the UC framework [13] here. Protocol machines, ideal functionalities, the adversary, and the environment are all modeled as interactive Turing machine (ITM) instances, or ITIs. An execution of protocol π consists of a series of activations of ITIs, starting with the environment $\mathcal{Z}$ who provides inputs to and collects outputs from the parties and the adversary $\mathcal{A}$; parties can also give input to and collect output from sub-parties, and $\mathcal{A}$ can communicate with parties via messages. Corruption of parties is modeled by a special `corrupt` message sent from $\mathcal{A}$ to the party; upon receipt of this message, the party sends its entire local state to $\mathcal{A}$, and in all future activations follows the instructions of $\mathcal{A}$. Note that a party p_i can only be corrupted once $\mathcal{A}$ receives a special (`corrupt` p_i) input from $\mathcal{Z}$. Denote by $\text{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}$ the probability distribution ensemble corresponding to the output of $\mathcal{Z}$ at the end of an execution of π with adversary $\mathcal{A}$. The ideal-world process for functionality $\mathcal{F}$ is simply defined as an execution of the ideal protocol $\text{IDEAL}_{\mathcal{F}}$, in which the

so-called "dummy" parties just forward inputs from $\mathcal{Z}$ to $\mathcal{F}$ and forward outputs from $\mathcal{F}$ to $\mathcal{Z}$. The corresponding ensemble is denoted by $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$.

We are interested in unconditional security. Thus, we say that a protocol π *UC-realizes* an ideal functionality $\mathcal{F}$ if for any computationally unbounded adversary $\mathcal{A}$, there exists a simulator $\mathcal{S}$ (polynomial in the complexity of $\mathcal{A}$) such that for any computationally unbounded environment $\mathcal{Z}$, we have $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \equiv \text{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}$. *Statistical* UC-realization requires only that the two ensembles be indistinguishable. When π is a $(\mathcal{G}_1,\ldots,\mathcal{G}_n)$-hybrid protocol (i.e., making subroutine calls to $\text{IDEAL}_{\mathcal{G}_1},\ldots,\text{IDEAL}_{\mathcal{G}_n}$), we say that π UC-realizes $\mathcal{F}$ in the $(\mathcal{G}_1,\ldots,\mathcal{G}_n)$-hybrid model. It turns out that (regular) UC-realization is equivalent to UC-realization with respect to the "dummy" adversary $\mathcal{D}$, which simply follows the instructions of $\mathcal{Z}$ on which messages to send, and reports all received messages to $\mathcal{Z}$.

We will assume synchronous computation (i.e., our protocols proceed in rounds), and the adversary is assumed to be rushing. Although our treatment is in the (G)UC setting, to avoid over-complicating the exposition we use the standard round-based language of, e.g., [12, 38]. Notwithstanding, such specifications can be translated to the synchronous UC model of Katz *et al.* [33] by assuming a clock functionality and bounded (zero) delay channels.

2.2 Building Blocks

Here we present some building blocks that we will use in our constructions, as well as (somewhat informal) property-based definitions to contrast with our UC formulations.

▶ **Definition 1** (SMT). *A protocol Π achieves (t-)*SMT *if it allows S to send a message $m \in \mathcal{M}$ to R such that the following hold for any adversary $\mathcal{A}$ corrupting up to t of the wires:*

- RELIABILITY: *R correctly outputs $m' = m$.*
- SECRECY: *$\mathcal{A}$ learns no information about m.*

We define RMT by omitting the secrecy property, and AE-RMT and AE-SMT are defined by only requiring the properties to hold for privileged S and R (over some sparse graph).

For simplicity, we will use the 3-phase SMT protocol $\Pi_{\text{DDWY}}(\vec{\gamma},\tau,m)$ presented in Figure 3 (Appendix A), which is essentially the FastSMT protocol from [20]. The n wires are denoted by $\vec{\gamma} = (\gamma_1,\ldots,\gamma_n)$, and $\tau = \lceil \frac{n}{2} \rceil - 1$ specifies how many corrupted wires can be tolerated. Although the protocol assumes access to an authenticated channel between S and R, this can be implemented by simply sending the message over all wires and having R take majority.

We will sometimes need the SMT-PD protocol $\Pi_{\text{PUB-SMT}}(\vec{\gamma},Pub,m,l)$ presented in Figure 4 (Appendix A), which was given in [25] and tolerates $n-1$ wire corruptions, assuming access to a public channel Pub and allowing a small probability of error (determined by l).

Finally, we present the security definition for (property-based) AE-MPC that was given in [25]. Recall that W is the set of privileged nodes, as a function of the set of corruptions.

▶ **Definition 2** (AE-MPC, [25]). *An n-player two-phase protocol Π achieves* AE-MPC *if for any initial value x_i for party P_i for each $i \in [n]$, any probabilistic polynomial-time computable function f, and any adversary $\mathcal{A}$ corrupting a set T of parties, there exists a subset W of honest parties such that the following properties hold at the end of the respective phases:*

Commitment phase: *During this phase, all players commit to their inputs.*

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

Computation phase:

- CORRECTNESS: *All $P_i \in W$ output $f(x_1^*,\ldots,x_n^*)$.*
- PRIVACY: *For all $P_i \in W$, no information about x_i^* beyond what can be inferred from the output of the corrupted parties leaks to $\mathcal{A}$ by this phase.*

3 Almost-Everywhere RMT and SMT

In this section, we use the UC framework to capture classical RMT and SMT protocols, which work in a model where the sender S and receiver R are connected by n disjoint *wires*, as in the abstract formulation of [20]. Although this is a simple model, here we give a novel treatment of these tasks that also serves as a warm-up to our later results, which look at these tasks over sparse graphs. Since the classical protocols may not provide security when enough of the wires are corrupted, we also introduce an AE *wrapper* that allows parties interacting with the underlying functionality to be marked as "doomed" in such cases. In Section 4, where we consider *remote* RMT and SMT, we will realize the same functionalities for RMT and SMT defined in this section, just in a wrapped form with different parameters.

We begin by modeling the disjoint wires from the classical setting as virtual wires that are represented by UC parties, which we call *wire-parties* and denote by $W_1, \ldots, W_n$ ($\vec{W}$ for short). The idea is that a wire-party can securely forward a message from S to R or vice versa as long as it is not corrupted, just as a wire in the classical model can securely transmit a message between S and R as long as it is free of corruptions. Since the basic communication model in UC is completely unprotected, we assume access to the ideal secure channel functionality $\mathcal{F}_{\text{SC}}^{S,R,\vec{W}}$ (see the full version [16] for a formal specification), which provides secure communication between an honest S or R and an honest wire-party over a single round. Looking ahead, this functionality is very similar to the functionality we use to capture secure channels between every pair of nodes connected by an edge in a sparse graph.[3]

For convenience, we use $\mathcal{F}_{\text{SC}}^{S,R,\vec{W}}$ to realize the wire channel functionality $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ presented in Figure 5 (Appendix A), which abstracts the process of sending a message to a wire-party, who then forwards it to S or R. The functionality actually allows sending a potentially different message through each wire-party in parallel, and it provides security for a given message as long as S, R, and the wire-party in question are all honest. Since we are considering virtual wires that consist of just one intermediate node, the functionality requires two rounds to generate output. In $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ (and all of our functionalities), $l(\cdot)$ refers to length and INFL is short for "influence" (see, e.g., [24]).

We can use the protocol $\Pi_{\text{WC}}(S, R, \vec{W})$, which simply routes each message m_i from S to R (or R to S) via W_i using two instances of $\mathcal{F}_{\text{SC}}^{S,R,\vec{W}}$, to realize $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$. The proof, as well as a formal specification of $\Pi_{\text{WC}}(S, R, \vec{W})$, can be found in the full version [16].

▶ **Proposition 3.** *Protocol $\Pi_{\text{WC}}(S, R, \vec{W})$ UC-realizes $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ in the $\mathcal{F}_{\text{SC}}^{S,R,\vec{W}}$-hybrid model.*

3.1 Universally Composable RMT and SMT

We model the task of RMT in UC with the authenticated channel functionality $\mathcal{F}_{\text{AUTH}}^{\mathcal{P},\mathsf{rnd}}$ (see the full version [16] for a formal specification), which is essentially Canetti's $\mathcal{F}_{\text{AUTH}}$ [14] with synchrony (the rnd parameter). There is also a parameter $\mathcal{P}$ representing the set of possible senders and receivers (the functionality itself is single-use). This parameter allows the functionality to verify that the actual sender and receiver can be identified as specific nodes in the network topology over which it is being realized, which is necessary because the realizing protocol will need to perform the same verification.

To realize $\mathcal{F}_{\text{AUTH}}^{\mathcal{P},\mathsf{rnd}}$ in the wire-party model ($\mathcal{P} = \{S, R\}$) assuming only a minority of the wire-parties get corrupted, we can simply duplicate the message through all wire-parties using

[3] Our RMT protocols only require reliable edges. However, we eventually need secure channels to achieve SMT and MPC, so for simplicity we present everything in the secure channels hybrid model.

Wrapper Functionality $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$

The wrapper is parameterized by a doom structure $\mathscr{D} = \{(T_1, D_1), \ldots, (T_m, D_m)\}$, where each $(T_i, D_i) \in 2^{\mathcal{P}} \times 2^{\mathcal{P}}$. The underlying functionality is $\mathcal{F}$. Let T be the set of currently corrupted parties and let D be the set of currently doomed parties, both initialized to $\emptyset$.

- Upon receiving $(\text{CORRUPT}, sid, P_i)$ from the adversary for $P_i \in \mathcal{P}$: If $(T \cup \{P_i\}, D) \in \mathscr{D}$, then set $T \leftarrow T \cup \{P_i\}$, relay the message to $\mathcal{F}$, and send back $\mathcal{F}$'s response.
- Upon receiving (DOOM, sid, P_i) from the adversary for $P_i \in \mathcal{P}$: If $(T, D \cup \{P_i\}) \in \mathscr{D}$, then set $D \leftarrow D \cup \{P_i\}$, send $(\text{CORRUPT}, sid, P_i)$ to $\mathcal{F}$, and send back $\mathcal{F}$'s response.
- Any other request from any party or the adversary is simply relayed to $\mathcal{F}$ without any further action and the output is relayed to the destination specified by $\mathcal{F}$.

Figure 1 AE wrapper functionality.

a single instance of $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$, and have the receiver (who may actually be S) take majority. We formally define protocol $\Pi_{\text{AUTH}}(S, R, \vec{W})$ and give a proof in the full version [16].

▶ **Theorem 4.** *Protocol $\Pi_{\text{AUTH}}(S, R, \vec{W})$ UC-realizes $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}}$ for $\mathsf{rnd} = 2$ in the $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$-hybrid model, against an adversary corrupting up to a minority of the wire-parties.*

Next, we capture SMT in UC with the secure channel functionality $\mathcal{F}_{\text{SMT}}^{\mathcal{P},\mathsf{rnd}}$ (see the full version [16] for a formal specification), which is essentially Canetti's $\mathcal{F}_{\text{SMT}}$ [14] with synchrony.

To realize $\mathcal{F}_{\text{SMT}}^{\mathcal{P},\mathsf{rnd}}$ in the wire-party model assuming only a minority of the wire-parties get corrupted, we can use protocol $\Pi_{\text{SMT}}(S, R, \vec{W})$ (outlined in the full version [16]), which is essentially the FastSMT protocol from [20] adapted for our UC treatment. That is, the sender (who may actually be R) runs protocol $\Pi_{\text{DDWY}}(\vec{\gamma}, \tau, m)$ (Figure 3) with the receiver, using $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ in phase 1 as a substitute for sending messages through the wires in $\vec{\gamma}$, and separate instances of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$ in phases 2 and 3 as a substitute for the authenticated channel.

▶ **Theorem 5.** *Protocol $\Pi_{\text{SMT}}(S, R, \vec{W})$ UC-realizes $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ for $\mathsf{rnd} = 6$ in the $(\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}, \mathcal{F}_{\text{AUTH}}^{\{S,R\},2})$-hybrid model, against an adversary corrupting up to a minority of the wire-parties.*

3.2 Corrupted Majorities of Wire-Parties

In the wire-party model, $\mathcal{F}_{\text{AUTH}}$ and $\mathcal{F}_{\text{SMT}}$ can only be realized when the adversary is restricted to corrupting only a minority of wire-parties. When corrupted majorities are allowed, the sender and receiver may essentially become doomed. To allow the simulator to handle such cases, we introduce an *AE wrapper functionality* (Figure 1) that allows parties to be marked as doomed according to the current set of corruptions. The wrapper accepts "doom" requests according to an adversary structure, and it processes them by simply having the underlying functionality treat doomed parties as fully corrupted. Recall that an adversary structure is a set of c-vectors of subsets of a participant set $\mathcal{P}$, where each component of a vector represents corruptions of a certain type. We consider adversary structures that consist of *doubles* of subsets, corresponding to a corrupted set and a doomed set, respectively, although the two may intersect[4]. We call such structures *doom structures*.

[4] This is a technicality, which simplifies some of our definitions and results. For example, the definition of AE-monotonicity (Section 5.1) would not be quite as short and intuitive otherwise.

In the wire-party model, we can realize *wrapped* $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}}$ and $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ with doom structure $\mathscr{D}_{\text{PSMT}}$, defined as follows using participant set $\mathcal{P} = \{S, R, W_1, \ldots, W_n\}$:

- $(T_i, D_i) \in \mathscr{D}_{\text{PSMT}}$ if and only if either $|T_i - \{S,R\}| < \frac{n}{2}$ and $D_i = \emptyset$ or $|T_i - \{S,R\}| \geq \frac{n}{2}$ and $D_i \subseteq \{S,R\}$

▶ **Theorem 6.** *Protocol* $\Pi_{\text{AUTH}}(S, R, \vec{W})$ *UC-realizes* $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{PSMT}}}(\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}})$ *for* $\mathsf{rnd} = 2$ *in the* $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$*-hybrid model, even against corrupted majorities of wire-parties.*

To realize wrapped $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$, define protocol $\Pi'_{\text{SMT}}(S, R, \vec{W})$ by replacing invocations of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$ in protocol $\Pi_{\text{SMT}}(S, R, \vec{W})$ with invocations of $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{PSMT}}}(\mathcal{F}_{\text{AUTH}}^{\{S,R\},2})$.

▶ **Theorem 7.** *Protocol* $\Pi'_{\text{SMT}}(S, R, \vec{W})$ *UC-realizes* $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{PSMT}}}(\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}})$ *for* $\mathsf{rnd} = 6$ *in the* $(\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}, \mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{PSMT}}}(\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}))$*-hybrid model, even against corrupted majorities of wire-parties.*

Next we turn to SMT-PD, which offers an alternative way to achieve SMT against a corrupted majority of wires, in the presence of a public channel.

3.3 Universally Composable SMT-PD

To capture SMT-PD in UC, we use our wire-party model from before, with the public channel modeled by assuming access to $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}'}$ for some rnd'. Protocol $\Pi_{\text{SMT-PD}}(S, R, \vec{W}, l)$ (see the full version [16] for a formal specification) is essentially the Pub-SMT protocol from [25] adapted for our UC treatment. In particular, the sender transmits a message v to the receiver by essentially executing protocol $\Pi_{\text{PUB-SMT}}(\vec{\gamma}, Pub, v, l)$ (Figure 4), where $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ is used in the first phase as a substitute for sending messages through the wires in $\vec{\gamma}$, and separate instances of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}'}$ are used in the remaining three phases as a substitute for Pub.

▶ **Theorem 8.** *Protocol* $\Pi_{\text{SMT-PD}}(S, R, \vec{W}, l)$ *statistically UC-realizes* $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ *for* $\mathsf{rnd} = 2 + 3 \cdot \mathsf{rnd}'$ *in the* $(\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}, \mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}'})$*-hybrid model, against an adversary corrupting all but one of the wire-parties.*

4 Almost-Everywhere *Remote* RMT and SMT

In this section, we consider *remote*—i.e. over a sparse graph G_n—RMT and SMT. As in Section 3, we model the network topology using the parameterized secure channel functionality $\mathcal{F}_{\text{SC}}^{G_n}$ presented in Figure 6 (Appendix A), which provides secure channels only between parties that are connected in G_n. Instead of always working directly with $\mathcal{F}_{\text{SC}}^{G_n}$, we also use it to realize the remote secure channel functionality $\mathcal{F}_{\text{R-SC}}^{G_n}$ (see the full version [16] for a formal specification), the counterpart to $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ from Section 3. This functionality provides secure communication over a single path, as long as no node on the path is corrupted. Using protocol $\Pi_{\text{R-SC}}(G_n)$ (see the full version [16] for details), we can realize $\mathcal{F}_{\text{R-SC}}^{G_n}$ by simply forwarding the message along the path, which leads to the following statement (proof omitted).

▶ **Proposition 9.** *Protocol* $\Pi_{\text{R-SC}}(G_n)$ *UC-realizes* $\mathcal{F}_{\text{R-SC}}^{G_n}$ *in the* $\mathcal{F}_{\text{SC}}^{G_n}$*-hybrid model.*

4.1 AE Remote RMT

We first show how classical AE-RMT protocols from the AE agreement literature can be adapted to UC-realize our wrapped $\mathcal{F}_{\text{AUTH}}^{\mathcal{P},\mathsf{rnd}}$ functionality, using doom structures that are derived from the protocol and the underlying sparse graph.

Graphs of Constant Degree. We first describe a scheme due to Dwork *et al.* [21], which guarantees AE *reliable* communication in classes of constant-degree graphs (such as their "butterfly" network) that admit a certain three-phase transmission scheme. At a high-level, the scheme associates with every node v a *fan-in* set $\Gamma_{\text{in}}(v)$ and a *fan-out* set $\Gamma_{\text{out}}(v)$ of a fixed size s. In addition, (not necessarily vertex-disjoint) paths from a node to its sets are specified, as well as (vertex-disjoint) paths from one node's fan-out set to another node's fan-in set. Node u transmits a message to node v by first sending it to all members of $\Gamma_{\text{out}}(u)$; each member then forwards the message to its connected (via a path) node in $\Gamma_{\text{in}}(v)$; and finally each member of $\Gamma_{\text{in}}(v)$ forwards the message to v, who simply takes majority.

Let $G_{\text{DPPU}} = (V_{\text{DPPU}}, E_{\text{DPPU}})$ be a graph that admits such a scheme. To realize wrapped $\mathcal{F}_{\text{AUTH}}^{V_{\text{DPPU}},\mathsf{rnd}}$, we use protocol $\Pi_{\text{R-AUTH}}^{\text{DPPU}}$ (outlined in the full version [16]), which is a straightforward adaptation of the scheme in the $\mathcal{F}_{\text{R-SC}}^{G_{\text{DPPU}}}$-hybrid model, and the doom structure $\mathscr{D}_{\text{DPPU}}$:

- For any corruption set T_i, let $D_{\text{DPPU}}(T_i)$ be a subset of participants P such that at least $\frac{1}{8}$ of the paths from P to $\Gamma_{\text{out}}(P)$ or at least $\frac{1}{8}$ of the paths from $\Gamma_{\text{in}}(P)$ to P are corrupted.
- Now, let $(T_i, D_i) \in \mathscr{D}_{\text{DPPU}}$ if and only if $|T_i| < s/4$ and $D_i \subseteq D_{\text{DPPU}}(T_i)$.

▶ **Theorem 10.** *Protocol $\Pi_{\text{R-AUTH}}^{\text{DPPU}}$ UC-realizes $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{DPPU}}}(\mathcal{F}_{\text{AUTH}}^{V_{\text{DPPU}},\mathsf{rnd}})$ for some $\mathsf{rnd} \in O(\log n)$ in the $\mathcal{F}_{\text{R-SC}}^{G_{\text{DPPU}}}$-hybrid model, against an adversary corrupting less than $s/4$ nodes.*

Upfal [43] proposed an alternative transmission scheme for constant-degree graphs, which actually works over *any* graph; however, his optimal result is achieved only on constant-degree expander graphs with specific parameters. The main limitation of the scheme is that it is computationally expensive. Node u transmits a message to node v by sending it through all the simple paths connecting them. As the message travels along a path to v, each node on the path appends the ID of the previous node to the message. This way each message received from a corrupted path will contain at least one ID of a corrupted node, and the receiver can enumerate over all the possible corruption sets to recover the message.

Let $G_n^{\text{UPFAL}} = (V_{\text{UPFAL}}, E_{\text{UPFAL}})$ be a d-regular expander graph. To realize wrapped $\mathcal{F}_{\text{AUTH}}^{V_{\text{UPFAL}},\mathsf{rnd}}$, we use protocol $\Pi_{\text{R-AUTH}}^{\text{UPFAL}}$ (outlined in the full version [16]), which is a straightforward adaptation of Upfal's scheme in the $\mathcal{F}_{\text{SC}}^{G_n^{\text{UPFAL}}}$-hybrid model, and the following doom structure $\mathscr{D}_{\text{UPFAL}}$:

- First, define $D_{\text{UPFAL}}(T_i)$ by the following iterative process: Starting with the set $S = T_i$, repeatedly add all participants $Q \notin S$ such that at least $\frac{1}{5}$ of Q's neighbors are in S.
- Now, let $(T_i, D_i) \in \mathscr{D}_{\text{UPFAL}}$ if and only if $|T_i| < t < 1/72n$ and $D_i \subseteq D_{\text{UPFAL}}(T_i)$.

▶ **Theorem 11.** *Protocol $\Pi_{\text{R-AUTH}}^{\text{UPFAL}}$ UC-realizes $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{UPFAL}}}(\mathcal{F}_{\text{AUTH}}^{V_{\text{UPFAL}},\mathsf{rnd}})$ for some $\mathsf{rnd} \in O(\log n)$ in the $\mathcal{F}_{\text{SC}}^{G_n^{\text{UPFAL}}}$-hybrid model, against an adversary corrupting less than $1/72n$ nodes.*

Although the simulator we construct is inefficient, that seems reasonable since the protocol itself runs in exponential time.

Graphs of Poly-Logarithmic Degree. Chandran *et al.* [17] proposed an AE-RMT scheme over a randomly constructed graph of poly-logarithmic degree. A very high-level idea of their construction is to transmit a message via multiple paths, while also performing some sort of error correction along the way. Their graph is comprised of some randomly generated, overlapping, fully connected committees that are themselves connected via the butterfly network. They also assign and connect to each node a number of those committees as *helpers*. Node u transmits a message to node v by sending it to all of u's helper committees, who then transmit it to v's helper committees using the transmission scheme of Dwork *et al.* [21] at the committee level. Finally, v takes majority over the values received from its helpers. As the

message travels from one committee to another, error correction occurs using a *differential agreement* protocol [22] run by the nodes in the destination committee.

Let $G_n^{\text{CGO}} = (V_{\text{CGO}}, E_{\text{CGO}})$ be a graph constructed as above. To realize wrapped $\mathcal{F}_{\text{AUTH}}^{V_{\text{CGO}},\mathsf{rnd}}$, we use protocol $\Pi_{\text{R-AUTH}}^{\text{CGO}}$ (outlined in the full version [16]), which is a straightforward adaptation of the above scheme in the $\mathcal{F}_{\text{SC}}^{G_n^{\text{CGO}}}$-hybrid model, and the following doom structure $\mathscr{D}_{\text{CGO}}$:

- First, for any set of corruptions T_i, let $D_{\text{CGO}}(T_i)$ be the set of all participants P such that $P \in T_i$ or at least $\frac{1}{6}$ of P's helper committees are unprivileged. A committee is considered corrupted if more than $\frac{1}{4}$ of its members are corrupted, and committees are categorized as unprivileged according to the $D_{\text{DPPU}}(\cdot)$ function defined above.
- Now, let $(T_i, D_i) \in \mathscr{D}_{\text{CGO}}$ if and only if corrupting the nodes in T_i causes at most $\frac{n \log^k n}{4 \log(n \log^k n)}$ committees to become corrupted, and $D_i \subseteq D_{\text{CGO}}(T_i)$.

Chandran *et al.* [17] proved that there exist constants $\alpha_{\text{CGO}}, \beta_{\text{CGO}}$ such that for any T_i with $|T_i| < \alpha_{\text{CGO}} n$, it holds that $|D_{\text{CGO}}(T_i)| < \beta_{\text{CGO}} \frac{|T_i|}{\log n}$. For those constants we have:

▶ **Theorem 12.** *Protocol* $\Pi_{\text{R-AUTH}}^{\text{CGO}}$ *UC-realizes* $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{CGO}}}(\mathcal{F}_{\text{AUTH}}^{V_{\text{CGO}},\mathsf{rnd}})$ *for* $\mathsf{rnd} \in O(\log n \cdot \log\log n)$ *in the* $\mathcal{F}_{\text{SC}}^{G_n^{\text{CGO}}}$*-hybrid model, against an adversary corrupting less than* $\alpha_{\text{CGO}} n$ *nodes.*

Graphs of Logarithmic Degree. Jayanti *et al.* [32] recently proposed a transmission scheme over logarithmic-degree graphs. Their graphs consist of multiple layers that are all constructed using the same method but over randomly permuted sets of nodes. In each layer, nodes are partitioned into committees of size s that are connected via the butterfly network and have Upfal's [43] expander graph instantiated inside them. We call this family of graphs $\mathcal{G}_{\text{JRV}}$. To transmit a message from node u to node v, in each layer u sends the message to all its committee members using Upfal's transmission scheme, and then the committee transmits it to v's committee using the transmission scheme of Dwork *et al.* [21] at the committee level. Finally, all of v's committee members send the value to v so that it can take majority over what is received from all the layers combined. There is also some type of error correction when messages travel from one committee to another.

Let $G_n^{\text{JRV}} = (V_{\text{JRV}}, E_{\text{JRV}}) \in \mathcal{G}_{\text{JRV}}$ with $|V_{\text{JRV}}| = n$. To realize wrapped $\mathcal{F}_{\text{AUTH}}^{V_{\text{JRV}},\mathsf{rnd}}$, we use protocol $\Pi_{\text{R-AUTH}}^{\text{JRV}}$ (outlined in the full version [16]), which is a straightforward adaptation of the above scheme in the $\mathcal{F}_{\text{SC}}^{G_n^{\text{JRV}}}$-hybrid model, and the following doom structure $\mathscr{D}_{\text{JRV}}$:

- First, in each layer of G_n^{JRV}, if a committee contains more than $\frac{1}{72}s$ corruptions, call it *bad*. If the total number of bad committees in a layer exceeds $\frac{n/s}{4 \log(n/s)}$, call the layer *bad*.
- Next, for any set of corruptions T_i, let $D_{\text{JRV}}(T_i)$ be the set of all participants P such that $P \in T_i$ or P is doomed in more than $\frac{1}{10}z$ layers among all the good layers. A node is considered doomed in a layer if it is located in a doomed committee (with respect to $D_{\text{DPPU}}(\cdot)$) or is doomed itself within its committee (with respect to $D_{\text{UPFAL}}(\cdot)$).
- Now, let $(T_i, D_i) \in \mathscr{D}_{\text{JRV}}$ if and only if corrupting the nodes in T_i causes at most $\frac{1}{5}$ of the layers to become bad, and $D_i \subseteq D_{\text{JRV}}(T_i)$.

Jayanti *et al.* [32] proved there exists a graph $G_n^{\text{JRV}} \in \mathcal{G}_{\text{JRV}}$ and constants $\alpha_{\text{JRV}}, \beta_{\text{JRV}}$ such that for T_i with $|T_i| < \alpha_{\text{JRV}} n$, it holds that $|D_{\text{JRV}}(T_i)| < \beta_{\text{JRV}} \frac{|T_i|}{\log n}$. For such a graph we have:

▶ **Theorem 13.** *Protocol* $\Pi_{\text{R-AUTH}}^{\text{JRV}}$ *UC-realizes* $\mathcal{W}_{\text{AE}}^{\mathscr{D}_{\text{JRV}}}(\mathcal{F}_{\text{AUTH}}^{V_{\text{JRV}},\mathsf{rnd}})$ *for some* $\mathsf{rnd} \in O(\log n \cdot \log\log\log n)$ *in the* $\mathcal{F}_{\text{SC}}^{G_n^{\text{JRV}}}$*-hybrid model, against adversaries corrupting less than* $\alpha_{\text{JRV}} n$ *nodes.*

4.2 AE Remote SMT

To achieve AE *secure* communication over the constant-degree graphs studied by Dwork *et al.* [21], we can apply the approach that we used in Section 3.2 to obtain AE-SMT in

the wire-party model. That is, we can adapt protocol $\Pi'_{\text{SMT}}(S, R, \vec{W})$ to work over the three-phase paths in G_{DPPU}, and the resulting protocol $\Pi^{\text{DPPU}}_{\text{R-SMT}}$ (formally outlined in the full version [16]) realizes wrapped $\mathcal{F}^{V_{\text{DPPU}},\mathsf{rnd}'}_{\text{SMT}}$ for some rnd' with the same doom structure $\mathscr{D}_{\text{DPPU}}$ from Section 4.1. Let ℓ denote the maximum length of any of the three-phase paths.

▶ **Theorem 14.** *Protocol $\Pi^{\text{DPPU}}_{\text{R-SMT}}$ UC-realizes $\mathcal{W}^{\mathscr{D}_{\text{DPPU}}}_{\text{AE}}(\mathcal{F}^{V_{\text{DPPU}},2\cdot\mathsf{rnd}+\ell}_{\text{SMT}})$ in the $(\mathcal{W}^{\mathscr{D}_{\text{DPPU}}}_{\text{AE}}(\mathcal{F}^{V_{\text{DPPU}},\mathsf{rnd}}_{\text{AUTH}}), \mathcal{F}^{G_{\text{DPPU}}}_{\text{R-SC}})$-hybrid model for $\mathsf{rnd} \in O(\log n)$, against an adversary corrupting less than $s/4$ nodes.*

The above technique cannot in general be extended to other AE-RMT schemes, because it requires a majority of honest paths between any pair of privileged nodes to realize a secure link between them. Many transmission schemes, such as Upfal's [43], do not guarantee such a property for privileged nodes. To realize AE-SMT using other transmission schemes, one approach is to use SMT-PD, which only requires a single honest path between sender and receiver to establish a secure channel, assuming access to a public channel. This approach can be used to make any AE-RMT scheme secure, since these schemes realize an authenticated channel (between privileged nodes) and guarantee at least one honest path between any pair of privileged nodes. The downside is that only statistical security is obtained.

We first introduce some notation. Given a doom structure $\mathscr{D}$ (with participant set $\mathcal{P}$), denote by $\mathsf{dom}(\mathscr{D})$ the set of values that appear as a first component in $\mathscr{D}$ (in other words, the set of all corruption sets allowed by $\mathscr{D}$). Say that $\mathscr{D}$ is *t-complete* if $\max_{(T_i,D_i)\in\mathscr{D}} |T_i| = t$, and $T \in \mathsf{dom}(\mathscr{D})$ for all $T \subseteq \mathcal{P}$ with $|T| \leq t$ (in other words, if all possible sets of corruptions of size at most t are allowed by $\mathscr{D}$). Moreover, say that a doom structure $\mathscr{D}$ is *D-monotone* if whenever $(T_j, D_j) \in \mathscr{D}$ and $D_i \subseteq D_j$, it holds that $(T_j, D_i) \in \mathscr{D}$. We note that all of our doom structures are t-complete and D-monotone.

Now, let $G_n = (V, E)$ be a graph with polynomially many paths of length at most ℓ specified between every pair of nodes. Suppose we already know how to realize wrapped $\mathcal{F}^{V,\mathsf{rnd}}_{\text{AUTH}}$ for some rnd, with respect to a doom structure $\mathscr{D}_{\text{SMT-PD}}$ (with $\mathcal{P} = V$) that is t-complete and D-monotone and moreover satisfies the following condition: For all $T \subseteq V$ with $|T| \leq t$, at least one of the specified paths between any pair of nodes in $V - T - \cup_{(T,D_i)\in\mathscr{D}_{\text{SMT-PD}}} D_i$ is completely contained in $V - T$. Then, we can realize wrapped $\mathcal{F}^{V,\mathsf{rnd}'}_{\text{SMT}}$ using protocol $\Pi_{\text{R-SMT-PD}}(G_n)$ (formally outlined in the full version [16]), which is essentially protocol $\Pi_{\text{SMT-PD}}(S, R, \vec{W}, l)$ from Section 3.3 adapted to work over the specified paths in G_n, and the same doom structure $\mathscr{D}_{\text{SMT-PD}}$. For such a doom structure we obtain the following statement:

▶ **Theorem 15.** *Protocol $\Pi_{\text{R-SMT-PD}}(G_n)$ statistically UC-realizes $\mathcal{W}^{\mathscr{D}_{\text{SMT-PD}}}_{\text{AE}}(\mathcal{F}^{V,\ell+3\cdot\mathsf{rnd}}_{\text{SMT}})$ in the $(\mathcal{F}^{G_n}_{\text{R-SC}}, \mathcal{W}^{\mathscr{D}_{\text{SMT-PD}}}_{\text{AE}}(\mathcal{F}^{V,\mathsf{rnd}}_{\text{AUTH}}))$-hybrid model against a t-adversary.*

Observe that t-completeness allows for statements against threshold adversaries, and D-monotonicity is required in the simulation since the simulator can only doom parties one by one. According to [21], all the realizable doom structures for AE remote RMT satisfy the above condition. Therefore, protocol $\Pi_{\text{R-SMT-PD}}(G_n)$ can be used with any of the classes of sparse graphs discussed in Section 4.1 to achieve AE remote SMT with statistical security.

5 Almost-Everywhere Secure Computation

In this section, we consider general UC-secure computation in the almost-everywhere setting. We start by proving a composition theorem that shows how to compile a protocol Π realizing some functionality $\mathcal{F}$ with the help of several hybrids into an *almost-everywhere* version of Π, by wrapping each hybrid with a potentially different doom structure $\mathscr{D}_i$. These structures can be arbitrary, subject only to a certain monotonicity property, although they must correspond

Compiler $\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi)$

Apply the following modifications to protocol Π (which uses $\mathcal{F}_1,\ldots,\mathcal{F}_m$ as hybrids):

1. For each $i \in [m]$, instead of using $\mathcal{F}_i$, parties use $\mathcal{W}_{\text{AE}}^{\mathscr{D}_i}(\mathcal{F}_i)$ (which has the same input/output format to the parties).

Figure 2 The AE compiler.

to the same participant set (indeed, composition would not make much sense otherwise); the compiled protocol is then shown to realize a *wrapped* version of $\mathcal{F}$, using a new doom structure $\mathscr{D}'$. Moreover, we allow the original protocol Π to itself realize a wrapped functionality associated with some doom structure $\mathscr{D}$. This, along with the fact that the monotonicity property carries over to the new doom structure $\mathscr{D}'$, make the compiled protocol readily amenable to further composition. We conclude by applying a special case of the composition theorem to obtain AE-MPC over the sparse graphs that were considered in Section 4. Rather than constructing protocols from scratch, we simply apply our generic AE compiler to replace the secure channels that are used in standard MPC protocols with AE remote SMT.

5.1 A General Composition Theorem

Let us first introduce some notation. Say that a doom structure $\mathscr{D}$ is *AE-monotone* if whenever $(T_i, D_i) \in \mathscr{D}$ and $T_i \subseteq T_j$ for $T_j \in \mathsf{dom}(\mathscr{D})$, it holds that $(T_j, D_i) \in \mathscr{D}$. Different from the standard notion of monotonicity in the general adversary literature, AE-monotonicity captures the intuitive property that when additional parties are corrupted, parties that were previously doomed are still doomed (or newly corrupted). AE-monotonicity seems to be important for simulatability; for example, the simulator may want to make a doom request for a newly doomed party only after some additional parties are corrupted in the meantime, and in such a case the doom structure needs to admit that request. Fortunately, all of our doom structures are AE-monotone.

The AE compiler is shown in Figure 2. It takes as input a protocol Π realizing some wrapped functionality $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ in the $(\mathcal{F}_1,\ldots,\mathcal{F}_m)$-hybrid model and turns it into a protocol that works in the $(\mathcal{W}_{\text{AE}}^{\mathscr{D}_1}(\mathcal{F}_1),\ldots,\mathcal{W}_{\text{AE}}^{\mathscr{D}_m}(\mathcal{F}_m))$-hybrid model. Of course, the compiled protocol will not in general realize wrapped $\mathcal{F}$ with the same doom structure $\mathscr{D}$. In the following theorem, we construct a new doom structure $\mathscr{D}'$ representing the level of AE-security that is retained. Since we consider general adversaries, the compiled protocol can tolerate a set T' of corruptions only if T' can be tolerated by *all* of the assumed doom structures (i.e., $\mathscr{D}$ as well as $\mathscr{D}_1,\ldots,\mathscr{D}_m$). Furthermore, the set of parties in the compiled protocol that are considered doomed (relative to T') can consist of, roughly speaking, parties that are doomed with respect to *any* of the wrapped hybrids (such parties are collected in $D(T')$ below) or that would have been doomed in the original protocol Π (the parties denoted by A). In fact, since Π may already carry some level of AE-security, as captured by $\mathscr{D}$, we must expand the latter set to include parties that only become doomed when some or all of the parties in the former set are actually corrupted. Thus, we require that $T' \cup D(T')$ is also tolerated by $\mathscr{D}$.

▶ **Theorem 16.** *Let $\mathscr{D}, \mathscr{D}_1,\ldots,\mathscr{D}_m$ be AE-monotone doom structures over the same participant set $\mathcal{P}$. Let $\mathcal{T} = \mathsf{dom}(\mathscr{D})$ and $\mathcal{T}' = (\bigcap_{i=1}^m \mathsf{dom}(\mathscr{D}_i)) \cap \mathcal{T}$. For any $T' \in \mathcal{T}'$, define*

$$D(T') = \bigcup_{i=1}^{m} \left(\bigcup_{(T', D_j) \in \mathscr{D}_i} D_j \right).$$

Suppose that for all $T' \in \mathcal{T}'$, it holds that $T' \cup D(T') \in \mathcal{T}$. If protocol Π UC-realizes $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}}(\mathcal{F})$ in the $(\mathcal{F}_1, \ldots, \mathcal{F}_m)$-hybrid model against a $\mathcal{T}$-adversary, then $\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi)$ UC-realizes $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}'}(\mathcal{F})$ in the $(\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}_1}(\mathcal{F}_1), \ldots, \mathcal{W}_{\mathrm{AE}}^{\mathscr{D}_m}(\mathcal{F}_m))$-hybrid model against a $\mathcal{T}'$-adversary, where $\mathscr{D}'$ is defined as follows: For all $T' \in \mathcal{T}'$, we have $(T', D \cup A) \in \mathscr{D}'$ if $D \subseteq D(T')$ and $(T' \cup D(T'), A) \in \mathscr{D}$. Moreover, $\mathscr{D}'$ is AE-monotone.

In the specific case that Π realizes an *unwrapped* functionality $\mathcal{F}$ (indeed, one can always apply our AE wrapper to $\mathcal{F}$ with a doom structure of the form $\{(T_i, \emptyset)\}_i$, which is trivially AE-monotone, in order to obtain an equivalent functionality) in the $\mathcal{G}$-hybrid model against a threshold adversary, we obtain the following corollary:

▶ **Corollary 17.** *Let $\mathscr{D}$ be a t'-complete, D-monotone, and AE-monotone doom structure. Let $t = \max_{|T'|=t'} \left| \left(\bigcup_{(T', D_i) \in \mathscr{D}} D_i \right) \cup T' \right|$. If protocol Π UC-realizes $\mathcal{F}$ in the $\mathcal{G}$-hybrid model against a t-adversary, then $\mathcal{C}^{\mathscr{D}}(\Pi)$ UC-realizes $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}}(\mathcal{F})$ in the $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}}(\mathcal{G})$-hybrid model against a t'-adversary.*

Observe that t'-completeness allows the simulator to handle a threshold adversary that can corrupt *any* t' parties, and D-monotonicity is needed for the doom structure $\mathscr{D}$ used to wrap $\mathcal{G}$ to be preserved when wrapping $\mathcal{F}$. By construction, all of our doom structures satisfy these two properties. We remark that t has a natural interpretation: the maximum number of parties that can become unprivileged (with respect to $\mathscr{D}$) when t' parties are corrupted.

5.2 AE-MPC

We now present our main result: how to achieve almost-everywhere MPC over several classes of sparse graphs in a composable manner. We assume a protocol that achieves "regular" MPC over a complete network of point-to-point secure channels, and show how to transform it into a protocol that achieves AE-MPC (with a lower corruption threshold) over a sparse graph with secure channels only between connected parties, using our AE compiler. To capture the MPC task for n-ary function f, we use the functionality $\mathcal{F}_{\mathrm{MPC}}^{f,\mathcal{P},\mathsf{rnd}}$ (see the full version [16] for a formal specification), which is essentially Canetti's $\mathcal{F}_{\mathrm{SFE}}$ [14] with synchrony.

Although standard information-theoretic MPC protocols tolerating $t < \frac{n}{3}$ corruptions are known [8, 18], they assume access to a broadcast channel, noting that broadcast can be achieved when $t < \frac{n}{3}$. However, [30] showed that classical broadcast protocols are not adaptively secure in a simulation-based setting, and gave a VSS-based protocol that does in fact realize adaptively secure broadcast with perfect security for $t < \frac{n}{3}$, assuming only secure channels. Therefore, there exists a protocol that UC-realizes $\mathcal{F}_{\mathrm{MPC}}^{f,\mathcal{P},\mathsf{rnd}}$ for any n-ary function f and some rnd in the $\mathcal{F}_{\mathrm{SMT}}^{\mathcal{P},1}$-hybrid model, against an adversary corrupting less than $\frac{n}{3}$ parties. It is clear that this holds even in the $\mathcal{F}_{\mathrm{SMT}}^{\mathcal{P},\ell}$-hybrid model, for arbitrary ℓ. Now, by invoking Corollary 17 (which of course also offers statistical security) and then applying the (regular) UC composition theorem in tandem with our results in Theorems 14 and 15 showing how to achieve AE-SMT over several classes of sparse graphs with either perfect or statistical security, we obtain the following corollaries showing how to achieve AE-MPC over those classes of graphs, with different combinations of parameters (recall that the maximum number of doomed nodes is encoded into each doom structure), for any n-ary function f:

▶ **Corollary 18.** *There exists a protocol that UC-realizes $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}_{\mathrm{DPPU}}}(\mathcal{F}_{\mathrm{MPC}}^{f,V_{\mathrm{DPPU}},\mathsf{rnd}})$ in the $\mathcal{F}_{\mathrm{SC}}^{G_n^{\mathrm{DPPU}}}$-hybrid model against a t-adversary, for some rnd and $t \in O(\frac{n}{\log n})$.*

▶ **Corollary 19.** *Let $\mathrm{x} \in \{\mathrm{UPFAL}, \mathrm{CGO}, \mathrm{JRV}\}$. There exists a protocol statistically UC-realizing $\mathcal{W}_{\mathrm{AE}}^{\mathscr{D}_{\mathrm{x}}}(\mathcal{F}_{\mathrm{MPC}}^{f,V_{\mathrm{x}},\mathsf{rnd}})$ in the $\mathcal{F}_{\mathrm{SC}}^{G_n^{\mathrm{x}}}$-hybrid model against a t-adversary, for some rnd and $t \in O(n)$.*

References

1. Saurabh Agarwal, Ronald Cramer, and Robbert de Haan. Asymptotically optimal two-round perfectly secure message transmission. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, volume 4117 of *Lecture Notes in Computer Science*, pages 394–408. Springer, 2006. doi:10.1007/11818175_24.
2. Bar Alon, Eran Omri, and Anat Paskin-Cherniavsky. MPC with friends and foes. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020. doi:10.1007/978-3-030-56880-1_24.
3. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic library with nested operations. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS 2003, Washington, DC, USA, October 27-30, 2003*, pages 220–230. ACM, 2003. doi:10.1145/948109.948140.
4. Christian Badertscher, Ran Canetti, Julia Hesse, Björn Tackmann, and Vassilis Zikas. Universal composition with global subroutines: Capturing global setup within plain UC. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 1–30. Springer, 2020. doi:10.1007/978-3-030-64381-2_1.
5. Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction ledger: A composable treatment. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 324–356. Springer, 2017. doi:10.1007/978-3-319-63688-7_11.
6. Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin. Secure computation without authentication. *J. Cryptol.*, 24(4):720–760, 2011. doi:10.1007/s00145-010-9075-9.
7. Zuzana Beerliová-Trubíniová, Matthias Fitzi, Martin Hirt, Ueli M. Maurer, and Vassilis Zikas. MPC vs. SFE: perfect security in a unified corruption model. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 231–250. Springer, 2008. doi:10.1007/978-3-540-78524-8_14.
8. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988. doi:10.1145/62212.62213.
9. Elette Boyle, Ran Cohen, Deepesh Data, and Pavel Hubácek. Must the communication graph of MPC protocols be an expander? In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 243–272. Springer, 2018. doi:10.1007/978-3-319-96878-0_9.
10. Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the o($\sqrt{}$ n)-bit barrier: Byzantine agreement with polylog bits per party. In Avery Miller, Keren Censor-Hillel, and Janne H. Korhonen, editors, *PODC '21: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, July 26-30, 2021*, pages 319–330. ACM, 2021. doi:10.1145/3465084.3467897.
11. Jan Camenisch, Stephan Krenn, Ralf Küsters, and Daniel Rausch. iuc: Flexible universal composability made simple. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings,*

Part III, volume 11923 of *Lecture Notes in Computer Science*, pages 191–221. Springer, 2019. doi:10.1007/978-3-030-34618-8_7.

12 Ran Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptol.*, 13(1):143–202, 2000. doi:10.1007/s001459910006.

13 Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 136–145. IEEE Computer Society, 2001. doi:10.1109/SFCS.2001.959888.

14 Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, December 2005. Latest version at https://ia.cr/2000/067.

15 Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In Salil P. Vadhan, editor, *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 61–85. Springer, 2007. doi:10.1007/978-3-540-70936-7_4.

16 Nishanth Chandran, Pouyan Forghani, Juan Garay, Rafail Ostrovsky, Rutvik Patel, and Vassilis Zikas. Universally composable almost-everywhere secure computation. Cryptology ePrint Archive, Report 2021/1398, 2021. https://ia.cr/2021/1398.

17 Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Improved fault tolerance and secure computation on sparse networks. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 249–260. Springer, 2010. doi:10.1007/978-3-642-14162-1_21.

18 David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988. doi:10.1145/62212.62214.

19 Danny Dolev. Unanimity in an unknown and unreliable environment. In *22nd Annual Symposium on Foundations of Computer Science, Nashville, Tennessee, USA, 28-30 October 1981*, pages 159–168. IEEE Computer Society, 1981. doi:10.1109/SFCS.1981.53.

20 Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 36–45. IEEE Computer Society, 1990. doi:10.1109/FSCS.1990.89522.

21 Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. Fault tolerance in networks of bounded degree (preliminary version). In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 370–379. ACM, 1986. doi:10.1145/12130.12169.

22 Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential consensus. In Elizabeth Borowsky and Sergio Rajsbaum, editors, *Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston, Massachusetts, USA, July 13-16, 2003*, pages 211–220. ACM, 2003. doi:10.1145/872035.872066.

23 Matthias Fitzi, Martin Hirt, and Ueli M. Maurer. Trading correctness for privacy in unconditional multi-party computation (extended abstract). In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 121–136. Springer, 1998. doi:10.1007/BFb0055724.

24 Juan A. Garay, Aggelos Kiayias, and Hong-Sheng Zhou. A framework for the sound specification of cryptographic tasks. In *Proceedings of the 23rd IEEE Computer Security Foundations*

Symposium, CSF 2010, Edinburgh, United Kingdom, July 17-19, 2010, pages 277–289. IEEE Computer Society, 2010. doi:10.1109/CSF.2010.26.

25 Juan A. Garay and Rafail Ostrovsky. Almost-everywhere secure computation. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages 307–323. Springer, 2008. doi:10.1007/978-3-540-78967-3_18.

26 Juan A. Garay and Kenneth J. Perry. A continuum of failure models for distributed computing. In Adrian Segall and Shmuel Zaks, editors, *Distributed Algorithms, 6th International Workshop, WDAG '92, Haifa, Israel, November 2-4, 1992, Proceedings*, volume 647 of *Lecture Notes in Computer Science*, pages 153–165. Springer, 1992. doi:10.1007/3-540-56188-9_11.

27 Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.

28 Jacopo Griggio. *Perfectly secure message transmission protocols with low communication overhead and their generalization*. PhD thesis, Universiteit Leiden, 2012.

29 Martin Hirt and Ueli M. Maurer. Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In James E. Burns and Hagit Attiya, editors, *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, Santa Barbara, California, USA, August 21-24, 1997*, pages 25–34. ACM, 1997. doi:10.1145/259380.259412.

30 Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 466–485. Springer, 2010. doi:10.1007/978-3-642-13190-5_24.

31 Dennis Hofheinz and Victor Shoup. GNUC: A new universal composability framework. *J. Cryptol.*, 28(3):423–508, 2015. doi:10.1007/s00145-013-9160-y.

32 Siddhartha Jayanti, Srinivasan Raghuraman, and Nikhil Vyas. Efficient constructions for almost-everywhere secure computation. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 159–183. Springer, 2020. doi:10.1007/978-3-030-45724-2_6.

33 Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 477–498. Springer, 2013. doi:10.1007/978-3-642-36594-2_27.

34 Valerie King and Jared Saia. From almost everywhere to everywhere: Byzantine agreement with $\tilde{o}(n^{3/2})$ bits. In Idit Keidar, editor, *Distributed Computing, 23rd International Symposium, DISC 2009, Elche, Spain, September 23-25, 2009. Proceedings*, volume 5805 of *Lecture Notes in Computer Science*, pages 464–478. Springer, 2009. doi:10.1007/978-3-642-04355-0_47.

35 Kaoru Kurosawa and Kazuhiro Suzuki. Truly efficient 2-round perfectly secure message transmission scheme. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages 324–340. Springer, 2008. doi:10.1007/978-3-540-78967-3_19.

36 Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982. URL: http://doi.acm.org/10.1145/357172.357176, doi:10.1145/357172.357176.

37 Ueli Maurer and Renato Renner. Abstract cryptography. In Bernard Chazelle, editor, *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 1–21. Tsinghua University Press, 2011. URL: `http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/14.html`.

38 Jesper Buus Nielsen. *On Protocol Security in the Cryptographic Model*. PhD thesis, University of Aarhus, 2003.

39 Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980. URL: `http://doi.acm.org/10.1145/322186.322188`, `doi:10.1145/322186.322188`.

40 Hasan Md. Sayeed and Hosame Abu-Amara. Efficient perfectly secure message transmission in synchronous networks. *Inf. Comput.*, 126(1):53–61, 1996. `doi:10.1006/inco.1996.0033`.

41 Gabriele Spini and Gilles Zémor. Perfectly secure message transmission in two rounds. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 286–304, 2016. `doi:10.1007/978-3-662-53641-4\_12`.

42 K. Srinathan, Arvind Narayanan, and C. Pandu Rangan. Optimal perfectly secure message transmission. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 545–561. Springer, 2004. `doi:10.1007/978-3-540-28628-8\_33`.

43 Eli Upfal. Tolerating linear number of faults in networks of bounded degree. In Norman C. Hutchinson, editor, *Proceedings of the Eleventh Annual ACM Symposium on Principles of Distributed Computing, Vancouver, British Columbia, Canada, August 10-12, 1992*, pages 83–89. ACM, 1992. `doi:10.1145/135419.135437`.

44 Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber, editors, *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 128–136. ACM, 1982. `doi:10.1145/800070.802185`.

869 # A Functionalities and Protocols

Protocol $\Pi_{\text{DDWY}}(\vec{\gamma}, \tau, m)$

1. **(Phase 1)** The sender S sends $n\tau + 1$ strong pads SP_1,SP_2, ..., $SP_{n\tau+1}$. To send each strong pad, S chooses a random polynomial $f(x) \in \mathbb{Z}_q(x)$ of degree τ and sets $pad = f(0)$. Then for each $i \in [n]$ S chooses an additional random polynomial $h_i(x) \in \mathbb{Z}_q$ of degree τ such that $h_i(0) = f(i)$. Finally, for each $i \in [n]$, S sends $h_i(\cdot)$ with a vector of checking pieces $C_i = (c_{1i}, c_{2i}, \ldots, c_{ni})$ to R using wire γ_i where for all $i, j \in [n]$, $c_{ji} = h_j(i)$.
2. **(Phase 2)** For each $k \in [n]$, let T_k be received in the attempted transmission of SP_k and g_i, D_i be possibly corrupted information received as h_i, C_i. If for any T_a all the checking pieces c_{ji} and all polynomials $h_i(\cdot)$ are consistent then R interpolates the pad_a from T_a and sends "a, OK" to S over the authenticated channel. Otherwise, R finds an l such that $\{\text{conflicts of } T_l\} \subseteq \cup_{m \neq l}\{\text{conflicts of } T_m\}$, where any unordered pair (i, j) is called a conflict of T_k if $d_{ji} \neq g_j(i)$. Then R sends l and all T_m, $m \neq l$ back to S using authenticated channel.
3. **(Phase 3)**
   - If "a, OK" received over the authenticated channel in phase 2, then S sends $z = m + pad_a$ to R using the authenticated channel. Otherwise, S preforms error detection on all T_j's received from R and sends detected faults and $z = m + pad_i$ to R using authenticated channel.
   - If R previously sent "a, OK" to S in phase 1, then s/he computes $m = z - pad_a$. Otherwise, R corrects the faults in T_i, obtains pad_i and computes $m = z - pad_i$.

Figure 3 The SMT protocol from [20].

Protocol $\Pi_{\text{PUB-SMT}}(\vec{\gamma}, Pub, m, l)$

1. The sender S sends n uniformly random bit strings $R_1, R_2, \ldots, R_n$ of length $15l$ to the receiver R through wires $\gamma_1, \gamma_2, \ldots, \gamma_n$, respectively. Let $R'_1, R'_2, \ldots, R'_n$ be the strings received by R. R rejects all wires where $|R'_i| \neq 15l$.
2. For $i \in [n]$, S generates R^*_i by replacing $12l$ randomly chosen positions of R_i with "$*$." Then S sends $R^*_1, R^*_2, \ldots, R^*_n$ to R over Pub.
3. For any $i \in [n]$, if R^*_i and R'_i differ in any "opened" bits, R marks γ_i as "faulty." Then R sends an n-bit string to S over Pub that identifies faulty wires. Let $\vec{\overline{\gamma}} = \{\overline{\gamma_1}, \overline{\gamma_2}, \ldots, \overline{\gamma_s}\}$, $s \leq n$ denote the set of non-faulty wires, and $\overline{R_i}$, $|\overline{R_i}| = 12l$, $1 \leq i \leq s$, denote the corresponding string of unopened bits; let $\overline{R'_i}$ be the corresponding string in R's possession.
4. For $1 \leq i \leq s$, S chooses m_i such that $m = m_1 \oplus m_2 \oplus \cdots \oplus m_s$, and sends $S_i = E(m_i) \oplus \overline{R_i}$, $1 \leq i \leq s$, over Pub. R computes $m'_i = D(S_i \oplus \overline{R'_i})$ for all $1 \leq i \leq s$. Then R outputs $m' = m'_1 \oplus m'_2 \oplus \cdots \oplus m'_s$.

Figure 4 The SMT-PD protocol from [25].

Functionality $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$

The functionality is parameterized by the identities of the sender S, the receiver R, and the n wire-parties $\vec{W} = (W_1, \ldots, W_n)$. At the first activation, verify that $sid = (P_s, P_r, sid')$, where $\{P_s, P_r\} = \{S, R\}$. Initialize variables $m_1, \ldots, m_n$ to $\perp$.

- Upon receiving input $(\text{SEND}, sid, W_i, v_i)$ from P_s in round ρ (which is the same for all W_i), record $m_i \leftarrow v_i$. If any $P \in \{P_s, P_r, W_i\}$ is marked as corrupted, then send $(\text{SENDLEAK}, sid, W_i, m_i)$ to the adversary; otherwise send $(\text{SENDLEAK}, sid, W_i, l(m_i))$.
- Upon receiving $(\text{INFLSEND}, sid, W_i, m_i')$ from the adversary: If any $P \in \{P_s, P_r, W_i\}$ is corrupted, and $(\text{SENT}, sid, W_i, m_i)$ has not yet been sent to P_r, then set $m_i \leftarrow m_i'$.
- Upon receiving (FETCH, sid, W_i) from P_r in round ρ': If P_r is corrupted, then send $(\text{FETCHLEAK}, sid, W_i)$ to the adversary; otherwise, if $\rho' = \rho + 2$, then output $(\text{SENT}, sid, W_i, m_i)$ to P_r if it has not yet been sent.
- Upon receiving $(\text{OUTPUT}, sid, W_i)$ from the adversary: If P_r is corrupted, then output $(\text{SENT}, sid, W_i, m_i)$ to P_r if it has not yet been sent.
- Upon receiving $(\text{CORRUPT}, sid, P)$ from the adversary for $P \in \{P_s, P_r, W_1, \ldots, W_n\}$, mark P as corrupted. If P is some wire-party W_i, then send $(\text{SENDLEAK}, sid, m_i)$ to the adversary; otherwise, send $(\text{SENDLEAK}, sid, m_1, \ldots, m_n)$. If $P = P_r$, then additionally leak any previous fetch requests made by P_r.

Figure 5 The Wire Channel functionality.

Functionality $\mathcal{F}_{\text{SC}}^{G_n}$

The functionality is parameterized by a graph $G_n = (V, E)$ of party identities and communication edges. At the first activation, verify that $sid = (P_i, P_j, sid')$, where $(P_i, P_j) \in E$; else halt. Initialize variable m to $\perp$.

- Upon receiving input (SEND, sid, v) from P_i in round ρ, record $m \leftarrow v$. If P_i or P_j is marked as corrupted, then send $(\text{SENDLEAK}, sid, m)$ to the adversary; otherwise send $(\text{SENDLEAK}, sid, l(m))$.
- Upon receiving $(\text{INFLSEND}, sid, m')$ from the adversary: If P_i or P_j is corrupted, and (SENT, sid, m) has not yet been sent to P_j, then set $m \leftarrow m'$.
- Upon receiving (FETCH, sid) from P_j in round $\rho + 1$, output (SENT, sid, m) to P_j if it has not yet been sent.
- Upon receiving $(\text{CORRUPT}, sid, P)$ from the adversary for $P \in \{P_i, P_j\}$, mark P as corrupted and send $(\text{SENDLEAK}, sid, m)$ to the adversary.

Figure 6 The Secure Channel functionality for (incomplete) graph G_n.

B Proofs

Proof of Theorem 5. Let $\mathcal{A}$ be an adversary in the real world. We construct a simulator $\mathcal{S}$ in the ideal world, such that no environment can distinguish whether it is interacting with $\Pi_{\text{SMT}}(S, R, \vec{W})$ and $\mathcal{A}$, or with $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ and $\mathcal{S}$. The simulator internally runs a copy of $\mathcal{A}$, and plays the roles of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$, $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$, and the parties in a simulated execution of the protocol. All inputs from $\mathcal{Z}$ are forwarded to $\mathcal{A}$, and all outputs from $\mathcal{A}$ are forwarded to $\mathcal{Z}$. Moreover, whenever $\mathcal{A}$ corrupts a party in the simulation, $\mathcal{S}$ corrupts the same party in the ideal world by interacting with $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ (except if the party is a wire-party), and if the corruption was direct (i.e., not via one of the aiding functionalities), then $\mathcal{S}$ sends $\mathcal{A}$ the party's state and follows $\mathcal{A}$'s instructions thereafter for that party.

The simulated execution starts upon $\mathcal{S}$ receiving $(\textsc{SendLeak}, sid, \hat{m})$ from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ in round ρ for $sid = (P_s, P_r, sid')$, where $\hat{m} \in \{m, l(m)\}$ and m is the message to be sent. Now, $\mathcal{S}$ executes the first two phases of the protocol honestly, by simulating sending random strong pads (shares $h_i(\cdot)$ and checking pieces $\vec{C}_i = (c_{1i}, \ldots, c_{ni})$) from P_s to P_r through the n wire-parties (i.e., by simulating leakage from $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ to $\mathcal{A}$, and responding to corruption and influence requests directed from $\mathcal{A}$ to $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$) and by simulating sending the response from P_r to P_s over the authenticated channel (i.e., by appropriately playing the role of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$ for $\mathcal{A}$). For the third phase of the protocol, $\mathcal{S}$ simulates honestly, except for choosing z when P_s and P_r are both honest in which case $\mathcal{S}$ simulates sending a random value z from P_s to P_r over $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$ instead of $z = m \oplus Pad$. When P_s or P_r is corrupted by $\mathcal{A}$, $\mathcal{S}$ learns m via leakage from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ and thus can send $z = m \oplus Pad$ just like in the real protocol. Note that the simulated P_s may need to abort, and that if the simulated P_r aborts by outputting $\perp$, then $\mathcal{S}$ can influence $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$, since this can only happen if $\mathcal{A}$ corrupts P_s or P_r.

Next, we describe how $\mathcal{S}$ simulates P_r's response to a $\textsc{Fetch}$ input from $\mathcal{Z}$ in the real world. If P_r is corrupted by $\mathcal{A}$, then $\mathcal{S}$ can wait to receive $(\textsc{FetchLeak}, sid)$ from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$, upon which it possibly leaks the fetch to $\mathcal{A}$ and then sends $\textsc{InflSend}$ and $\textsc{Output}$ messages to $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ as appropriate. If P_s is corrupted by $\mathcal{A}$, then $\mathcal{S}$ needs to constantly influence $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ during the second phase of the protocol, so that the dummy P_r fetches the correct value. If neither P_s nor P_r is corrupted, then $\mathcal{S}$ can simply let the dummy P_r fetch from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ when instructed by $\mathcal{Z}$. An important case is when both P_s and P_r are honest in the beginning of the third phase (at the time $\mathcal{S}$ decides the value of z) and then at least one of them gets corrupted before the protocol ends (before the output is fetched). In this case, $\mathcal{A}$ receives enough leakage from $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ to interpolate the pad and compute the value of the message from z. Since z is chosen randomly by $\mathcal{S}$, the message learned by $\mathcal{A}$ deviates from what is sent by P_s, causing $\mathcal{Z}$ to distinguish the real and ideal worlds. In such a situation, $\mathcal{S}$ learns the actual value of m via leakage from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$, and hence it can cheat by calculating a fake pad' satisfying $z = m \oplus pad'$ and then simulate leaking from $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ to result in pad'.

The simulation is perfect. Indeed, by corrupting at most τ wires, $\mathcal{A}$ learns nothing about $h_i(0)$ for honest wires, because the $h_i(\cdot)$'s are independent random polynomials of degree τ. Moreover, $f(\cdot)$ is also a random polynomial of degree τ so $\mathcal{A}$ learns nothing about $f(0)$ (i.e., pad used by the protocol looks uniformly random to $\mathcal{Z}$). Thus, choosing a random value z by $\mathcal{S}$ looks perfectly indistinguishable from the real protocol execution to $\mathcal{Z}$. Besides, $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$ acts like an authenticated channel in the protocol, and hence in the real world P_r outputs the sender's input. (See the full version [16] for more details). ◀

Proof of Theorem 7. [Sketch] We construct a simulator $\mathcal{S}$ that is very similar to the simulator in the proof of Theorem 5. However, $\mathcal{S}$ now interacts with a wrapped functionality,

and corruption messages for wire-parties are indeed sent because they can now be processed by the wrapper. Another difference concerns the case in which P_s and P_r are not corrupted by $\mathcal{A}$. If $\mathcal{A}$ corrupts only a minority of the wire-parties, then $\mathcal{S}$ can simply use a random value of z in the third phase of the protocol, and let the dummy P_r fetch its output as before. Otherwise, as soon as enough wire-parties are corrupted, $\mathcal{S}$ sends a DOOM message for P_s to the wrapper, which will be accepted by definition of $\mathscr{D}_{\text{PSMT}}$, and obtains m as leakage. Now, $\mathcal{S}$ can use $z = m \oplus pad$ in the third phase, and influences the wrapper every time the value that the real-world P_r would have output changes (these influence messages will be accepted by the wrapper). Another issue that comes up in the case that P_s and P_r remain honest is that $\mathcal{A}$ might exceed a minority of wire-party corruptions only after $\mathcal{S}$ has already chosen a random z. However, $\mathcal{S}$ can handle this by cheating and computing a fake pad consistent with m, like the simulator in the proof of Theorem 5 does. Finally, $\mathcal{S}$ may need to simulate sender or receiver aborts when $\mathcal{A}$ corrupts a majority of wire-parties but not P_s or P_r. ◀

Proof of Theorem 8. Let $\mathcal{A}$ be an adversary in the real world. We construct a simulator $\mathcal{S}$ in the ideal world, such that no environment $\mathcal{Z}$ can distinguish whether it is interacting with $\Pi_{\text{SMT-PD}}(S, R, \vec{W}, l)$ and $\mathcal{A}$, or with $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ and $\mathcal{S}$. The simulator internally runs a copy of $\mathcal{A}$, and plays the roles of $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$, $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}'}$, and the parties in a simulated execution of the protocol. All inputs from $\mathcal{Z}$ are forwarded to $\mathcal{A}$, and all outputs from $\mathcal{A}$ are forwarded to $\mathcal{Z}$. Moreover, whenever $\mathcal{A}$ corrupts a party in the simulation, $\mathcal{S}$ corrupts the same party in the ideal world by interacting with $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ (except if the party is a wire-party), and if the corruption was direct (i.e., not via either of the aiding functionalities), then $\mathcal{S}$ sends $\mathcal{A}$ the party's state and thereafter follows $\mathcal{A}$'s instructions for that party.

The simulated execution starts upon $\mathcal{S}$ receiving $(\text{SENDLEAK}, sid, \hat{m})$ from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ in round ρ for $sid = (P_s, P_r, sid')$, where $\hat{m} \in \{m, l(m)\}$ and m is the message to be sent. Now, $\mathcal{S}$ simulates the first three phases of the protocol honestly, by simulating sending random bitstrings from P_s to P_r through the n wire-parties (i.e., by simulating leakage from $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$ to $\mathcal{A}$, and responding to corruption and influence requests directed from $\mathcal{A}$ to $\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}$) and by simulating sending a message from P_s to P_r or vice versa over the public channel (by appropriately playing the role of $\mathcal{F}_{\text{AUTH}}^{\{S,R\},\mathsf{rnd}'}$ for $\mathcal{A}$). In the fourth phase, $\mathcal{S}$ chooses random m_i's to be encoded (rather than m_i's such that $m = m_1 \oplus \cdots \oplus m_s$) if P_s and P_r are still honest; if P_s or P_r is corrupted by $\mathcal{A}$, then $\mathcal{S}$ learns m via leakage from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$.

Next, we describe how $\mathcal{S}$ simulates P_r's response to a FETCH input from $\mathcal{Z}$ in the real world. If P_r is corrupted by $\mathcal{A}$, then $\mathcal{S}$ can wait to receive $(\text{FETCHLEAK}, sid)$ from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$, upon which it possibly leaks the fetch to $\mathcal{A}$ and then sends INFLSEND and OUTPUT messages to $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ as appropriate. Otherwise, if P_s is corrupted by $\mathcal{A}$, then $\mathcal{S}$ needs to constantly influence $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ so that the dummy P_r fetches the correct value. Finally, if neither P_s nor P_r is corrupted, then $\mathcal{S}$ simply lets the dummy P_r fetch from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$ when instructed by $\mathcal{Z}$. In this case, the real-world P_r outputs m except with the error probability.

An important issue is that when P_s or P_r is corrupted only after $\mathcal{S}$ has already decided on the random m_i's to be encoded in the fourth phase, $\mathcal{A}$ may be able to recover some m' from its view of the bitstrings sent in the first phase, but m' may not equal m and this could allow $\mathcal{Z}$ to distinguish between the real and ideal worlds. However, $\mathcal{S}$ can handle this case by faking what was sent in the first phase. In particular, at least one bitstring (corresponding to an uncorrupted wire-party) sent in the first phase is not visible to $\mathcal{A}$, so $\mathcal{S}$ can redefine it to be consistent with m (which $\mathcal{S}$ learns from leakage from $\mathcal{F}_{\text{SMT}}^{\{S,R\},\mathsf{rnd}}$).

The simulation is not perfect as there is an error probability, but $\mathcal{Z}$ still cannot distinguish between the two worlds. In particular, when P_s and P_r are not corrupted by $\mathcal{A}$, the assumption

that at most all but one of the wire-parties are corrupted implies that the random bitstring sent on at least one wire in the first phase will mask the value of m from $\mathcal{A}$. ◀

Proof of Theorem 16. We first prove that $\mathscr{D}'$ is AE-monotone. Suppose that $(T_i, D_i) \in \mathscr{D}'$ and $T_i \subseteq T_j$ for $T_j \in \mathcal{T}'$. This means that $D_i = D \cup A$ for some D, A such that $D \subseteq D(T_i)$ and $(T_i \cup D(T_i), A) \in \mathscr{D}$. We want to show that $(T_j, D_i) \in \mathscr{D}'$, and it suffices to show that $D \subseteq D(T_j)$ and $(T_j \cup D(T_j), A) \in \mathscr{D}$. Since $D(T_i) \subseteq D(T_j)$ (using the fact that $\mathscr{D}_1, \ldots, \mathscr{D}_m$ are all AE-monotone), it follows that $D \subseteq D(T_j)$. On the other hand, since $T_i \cup D(T_i) \subseteq T_j \cup D(T_j)$, it follows that $(T_j \cup D(T_j), A) \in \mathscr{D}$ (using the fact that $\mathscr{D}$ is AE-monotone and that $T_j \cup D(T_j) \in \mathcal{T}$). We now prove the security of the compiled protocol.

Let $\mathcal{S}$ be a simulator (guaranteed to exist by the security of Π) such that no environment $\mathcal{Z}$ can distinguish whether it is interacting with Π and the dummy adversary $\mathcal{D}$, or with $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ and $\mathcal{S}$. We use $\mathcal{S}$ to construct a simulator $\mathcal{S}'$ such that no environment $\mathcal{Z}'$ can distinguish whether it is interacting with $\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi)$ and $\mathcal{D}$, or with $\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F})$ and $\mathcal{S}'$.

$\mathcal{S}'$ internally runs $\mathcal{S}$ and plays the role of the environment and $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ for it. Inputs from $\mathcal{Z}'$ are forwarded to $\mathcal{S}$, with some additional processing. When $\mathcal{Z}'$ sends a corruption request directed to a party (i.e., telling $\mathcal{D}$ to corrupt a party directly), this is forwarded without modification. However, when $\mathcal{Z}'$ sends message delivery requests directed to an instance of $\mathcal{W}_{\text{AE}}^{\mathscr{D}_i}(\mathcal{F}_i)$ for some $i \in [m]$ (e.g., telling $\mathcal{D}$ to send a CORRUPT or INFLUENCE message to that functionality), $\mathcal{S}'$ sends message delivery requests directed to a corresponding instance of $\mathcal{F}_i$, with the following exception: a request to deliver a DOOM message is replaced by a request to deliver a CORRUPT message if $\mathscr{D}_i$ would accept it, and is dropped otherwise.

Similarly, outputs from $\mathcal{S}$ are forwarded to $\mathcal{Z}'$, with some additional processing. Assuming that Π uses instances of $\mathcal{F}_1, \ldots, \mathcal{F}_m$ to handle all inter-party communication, these outputs should take the form of reports of incoming messages directed from either a party or an instance of an aiding functionality $\mathcal{F}_i$ to the dummy adversary for Π; thus, the processing done by $\mathcal{S}'$ is that reported messages from an instance of $\mathcal{F}_i$ are replaced by reported messages from an instance of $\mathcal{W}_{\text{AE}}^{\mathscr{D}_i}(\mathcal{F}_i)$. Finally, $\mathcal{S}'$ plays the role of $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ by simply forwarding messages from $\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F})$ to $\mathcal{S}$ as if coming from $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$, and forwarding messages directed to $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ (from $\mathcal{S}$) to $\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F})$, except that CORRUPT messages for doomed parties (i.e., parties that $\mathcal{Z}'$ did not request to corrupt) are replaced by DOOM messages.

Suppose for a contradiction that there is an environment $\mathcal{Z}'$ such that $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F}),\mathcal{S}',\mathcal{Z}'} \not\equiv \text{EXEC}_{\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi),\mathcal{D},\mathcal{Z}'}$. Then, we construct an environment $\mathcal{Z}$ such that $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F}),\mathcal{S},\mathcal{Z}} \not\equiv \text{EXEC}_{\Pi,\mathcal{D},\mathcal{Z}}$. The environment $\mathcal{Z}$ will simulate an interaction between $\mathcal{Z}'$ and $\mathcal{D}$, and output whatever $\mathcal{Z}'$ outputs, as well as do some additional processing. Whenever $\mathcal{Z}'$ instructs its dummy adversary to deliver a message to an instance of $\mathcal{W}_{\text{AE}}^{\mathscr{D}_i}(\mathcal{F}_i)$, this is translated by $\mathcal{Z}$ into a delivery request for a corresponding instance of $\mathcal{F}_i$ and forwarded to the external adversary (either $\mathcal{S}$ or $\mathcal{D}$), except that a request to deliver a DOOM message is converted into a request to deliver a CORRUPT message if allowed by $\mathscr{D}_i$ and dropped otherwise. Corruption requests directed to parties are forwarded to the external adversary unmodified.

Next, whenever $\mathcal{Z}$ receives subroutine output from the external adversary, this is forwarded to $\mathcal{Z}'$, except that reported messages from instances of $\mathcal{W}_{\text{AE}}^{\mathscr{D}_i}(\mathcal{F}_i)$ are translated into reported messages from corresponding instances of $\mathcal{F}_i$. Finally, $\mathcal{Z}$ simply relays inputs and outputs between $\mathcal{Z}'$ and parties. We conclude by claiming that $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F}),\mathcal{S}',\mathcal{Z}'} \equiv \text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F}),\mathcal{S},\mathcal{Z}}$ and $\text{EXEC}_{\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi),\mathcal{D},\mathcal{Z}'} \equiv \text{EXEC}_{\Pi,\mathcal{D},\mathcal{Z}}$. Indeed, if $\mathcal{Z}$ interacts with $\mathcal{W}_{\text{AE}}^{\mathscr{D}}(\mathcal{F})$ and $\mathcal{S}$, then the view of the simulated $\mathcal{Z}'$ within $\mathcal{Z}$ is identical to the view of $\mathcal{Z}'$ when interacting with $\mathcal{W}_{\text{AE}}^{\mathscr{D}'}(\mathcal{F})$ and $\mathcal{S}'$, and similarly if $\mathcal{Z}$ interacts with Π and $\mathcal{D}$, then the view of the simulated $\mathcal{Z}'$ within $\mathcal{Z}$ is identical to the view of $\mathcal{Z}'$ when interacting with $\mathcal{C}^{\mathscr{D}_1,\ldots,\mathscr{D}_m}(\Pi)$ and $\mathcal{D}$. ◀