

# 1 Universally Composable Almost-Everywhere 2 Secure Computation

3 Nishanth Chandran ✉

4 Microsoft Research, Bangalore, India

5 Pouyan Forghani ✉

6 Texas A&M University, College Station, TX, USA

7 Juan Garay ✉

8 Texas A&M University, College Station, TX, USA

9 Rafail Ostrovsky ✉

10 University of California, Los Angeles, CA, USA

11 Rutvik Patel ✉

12 Texas A&M University, College Station, TX, USA

13 Vassilis Zikas ✉

14 Purdue University, West Lafayette, IN, USA

## 15 — Abstract —

---

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

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

32 **2012 ACM Subject Classification** Theory of computation → Cryptographic protocols; Security and  
33 privacy → Information-theoretic techniques; Security and privacy → Formal security models

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

36 **Digital Object Identifier** 10.4230/LIPIcs.ITC.2022.14

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

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



© Nishanth Chandran, Pouyan Forghani, Juan Garay, Rafail Ostrovsky, Rutvik Patel, and Vassilis Zikas;

licensed under Creative Commons License CC-BY 4.0

3rd Conference on Information-Theoretic Cryptography (ITC 2022).

Editor: Dana Dachman-Soled; Article No. 14; pp. 14:1–14:25



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

46 The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes  
47 not withstanding any copyright annotation therein.

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

## 50 **1** Introduction

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

55 Despite great progress on the problem since it was first introduced and proven feasible [44,  
56 27, 8, 18] involving hundreds, if not thousands, of publications in cryptography and security,  
57 and, more recently, even implemented systems, the overwhelming majority of the solutions  
58 assume a *complete* communication network of either authenticated (aka reliable) or secure  
59 (both authenticated and private) point-to-point channels. In fact, with only a few exceptions,  
60 this is the case for both practical and theoretical works on MPC, and in particular for works  
61 on composable security of MPC—indeed, the latter almost exclusively assume a network  
62 that cannot be disconnected by the adversary. This creates a disconnect between the vast  
63 MPC literature and modern *ad-hoc* networks, such as the Internet, where the communication  
64 might be occurring over an incomplete graph, with some nodes even being routing nodes.

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

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

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

---

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

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

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

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

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

## 119 1.1 Related Work

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

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

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

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

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

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

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

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

---

<sup>2</sup> 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.

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

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

## 191 1.2 Overview of Our Results

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

## 306 2 Preliminaries

### 307 2.1 UC Basics

308 We briefly summarize the UC framework [13] here. Protocol machines, ideal functionalities,  
 309 the adversary, and the environment are all modeled as interactive Turing machine (ITM)  
 310 instances, or ITIs. An execution of protocol  $\pi$  consists of a series of activations of ITIs,  
 311 starting with the environment  $\mathcal{Z}$  who provides inputs to and collects outputs from the parties  
 312 and the adversary  $\mathcal{A}$ ; parties can also give input to and collect output from sub-parties,  
 313 and  $\mathcal{A}$  can communicate with parties via messages. Corruption of parties is modeled by a  
 314 special **corrupt** message sent from  $\mathcal{A}$  to the party; upon receipt of this message, the party  
 315 sends its entire local state to  $\mathcal{A}$ , and in all future activations follows the instructions of  $\mathcal{A}$ .  
 316 Note that a party  $p_i$  can only be corrupted once  $\mathcal{A}$  receives a special (**corrupt**  $p_i$ ) input  
 317 from  $\mathcal{Z}$ . Denote by  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}$  the probability distribution ensemble corresponding to the  
 318 output of  $\mathcal{Z}$  at the end of an execution of  $\pi$  with adversary  $\mathcal{A}$ . The ideal-world process for  
 319 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  $\pi$  *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  $\pi$  is a  $(\mathcal{G}_1, \dots, \mathcal{G}_n)$ -hybrid protocol (i.e., making subroutine calls to  $\text{IDEAL}_{\mathcal{G}_1}, \dots, \text{IDEAL}_{\mathcal{G}_n}$ ), we say that  $\pi$  UC-realizes  $\mathcal{F}$  in the  $(\mathcal{G}_1, \dots, \mathcal{G}_n)$ -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  $\Pi$  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  $\vec{\gamma} = (\gamma_1, \dots, \gamma_n)$ , 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  $\Pi_{\text{PUB-SMT}}(\vec{\gamma}, \text{Pub}, m, l)$  presented in Figure 4 (Appendix A), which was given in [25] and tolerates  $n - 1$  wire corruptions, assuming access to a public channel  $\text{Pub}$  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  $\Pi$  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  $f(x_1^*, \dots, x_n^*)$ .*
- **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  $W_1, \dots, W_n$  ( $\vec{W}$  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.<sup>3</sup>

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].

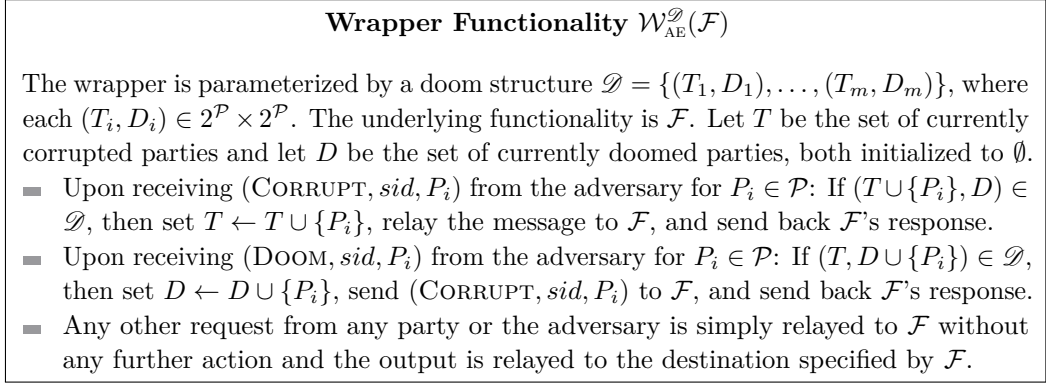
► **Proposition 3.** *Protocol  $\Pi_{\text{wc}}(S, R, \vec{W})$  UC-realizes  $\mathcal{F}_{\text{wc}}^{S,R,\vec{W}}$  in the  $\mathcal{F}_{\text{sc}}^{S,R,\vec{W}}$ -hybrid model.*

#### 3.1 Universally Composable RMT and SMT

We model the task of RMT in UC with the authenticated channel functionality  $\mathcal{F}_{\text{AUTH}}^{\mathcal{P},\text{rnd}}$  (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  $\mathcal{F}_{\text{AUTH}}^{\mathcal{P},\text{rnd}}$  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

<sup>3</sup> 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.



■ **Figure 1** AE wrapper functionality.

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

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

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

414 To realize  $\mathcal{F}_{\text{SMT}}^{\mathcal{P},\text{rnd}}$  in the wire-party model assuming only a minority of the wire-parties  
 415 get corrupted, we can use protocol  $\Pi_{\text{SMT}}(S, R, \vec{W})$  (outlined in the full version [16]), which  
 416 is essentially the FastSMT protocol from [20] adapted for our UC treatment. That is, the  
 417 sender (who may actually be  $R$ ) runs protocol  $\Pi_{\text{DDWY}}(\vec{\gamma}, \tau, m)$  (Figure 3) with the receiver,  
 418 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  
 419 separate instances of  $\mathcal{F}_{\text{AUTH}}^{\{S,R\},2}$  in phases 2 and 3 as a substitute for the authenticated channel.

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

## 422 3.2 Corrupted Majorities of Wire-Parties

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

<sup>4</sup> 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.

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

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

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

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

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

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

### 446 3.3 Universally Composable SMT-PD

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

454 ► **Theorem 8.** *Protocol  $\Pi_{\text{SMT-PD}}(S, R, \vec{W}, l)$  statistically UC-realizes  $\mathcal{F}_{\text{SMT}}^{\{S,R\},\text{rnd}}$  for  $\text{rnd} =$   
455  $2 + 3 \cdot \text{rnd}'$  in the  $(\mathcal{F}_{\text{WC}}^{S,R,\vec{W}}, \mathcal{F}_{\text{AUTH}}^{\{S,R\},\text{rnd}'})$ -hybrid model, against an adversary corrupting all but  
456 one of the wire-parties.*

## 457 4 Almost-Everywhere Remote RMT and SMT

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

467 ► **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.*

### 468 4.1 AE Remote RMT

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

472 **Graphs of Constant Degree.** We first describe a scheme due to Dwork *et al.* [21], which  
 473 guarantees AE *reliable* communication in classes of constant-degree graphs (such as their  
 474 “butterfly” network) that admit a certain three-phase transmission scheme. At a high-level,  
 475 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  
 476 fixed size  $s$ . In addition, (not necessarily vertex-disjoint) paths from a node to its sets are  
 477 specified, as well as (vertex-disjoint) paths from one node’s fan-out set to another node’s  
 478 fan-in set. Node  $u$  transmits a message to node  $v$  by first sending it to all members of  $\Gamma_{\text{out}}(u)$ ;  
 479 each member then forwards the message to its connected (via a path) node in  $\Gamma_{\text{in}}(v)$ ; and  
 480 finally each member of  $\Gamma_{\text{in}}(v)$  forwards the message to  $v$ , who simply takes majority.

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

- 484 ■ For any corruption set  $T_i$ , let  $D_{\text{DPPU}}(T_i)$  be a subset of participants  $P$  such that at least  $\frac{1}{8}$   
 485 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.
- 486 ■ Now, let  $(T_i, D_i) \in \mathcal{D}_{\text{DPPU}}$  if and only if  $|T_i| < s/4$  and  $D_i \subseteq D_{\text{DPPU}}(T_i)$ .

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

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

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

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

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

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

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

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

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

521 ■ First, for any set of corruptions  $T_i$ , let  $D_{\text{CGO}}(T_i)$  be the set of all participants  $P$  such  
522 that  $P \in T_i$  or at least  $\frac{1}{6}$  of  $P$ 's helper committees are unprivileged. A committee is  
523 considered corrupted if more than  $\frac{1}{4}$  of its members are corrupted, and committees are  
524 categorized as unprivileged according to the  $D_{\text{DPPU}}(\cdot)$  function defined above.

525 ■ Now, let  $(T_i, D_i) \in \mathcal{D}_{\text{CGO}}$  if and only if corrupting the nodes in  $T_i$  causes at most  
526  $\frac{n \log^k n}{4 \log(n \log^k n)}$  committees to become corrupted, and  $D_i \subseteq D_{\text{CGO}}(T_i)$ .

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

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

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

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

545 ■ First, in each layer of  $G_n^{\text{JRV}}$ , if a committee contains more than  $\frac{1}{72} s$  corruptions, call it  
546 *bad*. If the total number of bad committees in a layer exceeds  $\frac{n/s}{4 \log(n/s)}$ , call the layer *bad*.

547 ■ Next, for any set of corruptions  $T_i$ , let  $D_{\text{JRV}}(T_i)$  be the set of all participants  $P$  such that  
548  $P \in T_i$  or  $P$  is doomed in more than  $\frac{1}{10} z$  layers among all the good layers. A node is  
549 considered doomed in a layer if it is located in a doomed committee (with respect to  
550  $D_{\text{DPPU}}(\cdot)$ ) or is doomed itself within its committee (with respect to  $D_{\text{UPFAL}}(\cdot)$ ).

551 ■ Now, let  $(T_i, D_i) \in \mathcal{D}_{\text{JRV}}$  if and only if corrupting the nodes in  $T_i$  causes at most  $\frac{1}{5}$  of the  
552 layers to become bad, and  $D_i \subseteq D_{\text{JRV}}(T_i)$ .

553 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  
554 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:

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

## 557 4.2 AE Remote SMT

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

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

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

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

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

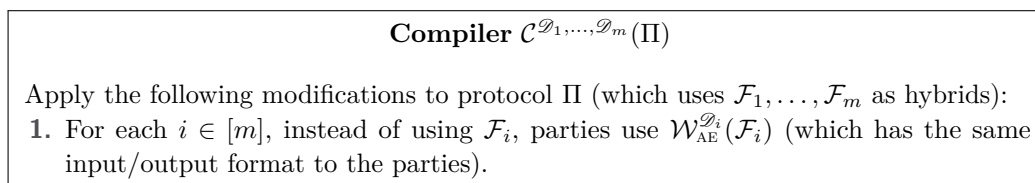
582 Now, let  $G_n = (V, E)$  be a graph with polynomially many paths of length at most  $\ell$   
 583 specified between every pair of nodes. Suppose we already know how to realize wrapped  $\mathcal{F}_{\text{AUTH}}^{\text{V}, \text{rnd}}$   
 584 for some  $\text{rnd}$ , with respect to a doom structure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

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

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

## 598 5 Almost-Everywhere Secure Computation

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



■ **Figure 2** The AE compiler.

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

## 613 5.1 A General Composition Theorem

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

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

637 ► **Theorem 16.** *Let  $\mathcal{D}, \mathcal{D}_1, \dots, \mathcal{D}_m$  be AE-monotone doom structures over the same partici-*  
 638 *part set  $\mathcal{P}$ . Let  $\mathcal{T} = \text{dom}(\mathcal{D})$  and  $\mathcal{T}' = (\bigcap_{i=1}^m \text{dom}(\mathcal{D}_i)) \cap \mathcal{T}$ . For any  $T' \in \mathcal{T}'$ , define*

$$639 \quad D(T') = \bigcup_{i=1}^m \left( \bigcup_{(T', D_j) \in \mathcal{D}_i} D_j \right).$$

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

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

649 ► **Corollary 17.** *Let  $\mathcal{D}$  be a  $t'$ -complete,  $D$ -monotone, and AE-monotone doom structure.*

650 *Let  $t = \max_{|T'|=t'} \left| \left( \bigcup_{(T', D_i) \in \mathcal{D}} D_i \right) \cup T' \right|$ . If protocol  $\Pi$  UC-realizes  $\mathcal{F}$  in the  $\mathcal{G}$ -hybrid model  
 651 against a  $t$ -adversary, then  $\mathcal{C}^{\mathcal{D}}(\Pi)$  UC-realizes  $\mathcal{W}_{\text{AE}}^{\mathcal{D}}(\mathcal{F})$  in the  $\mathcal{W}_{\text{AE}}^{\mathcal{D}}(\mathcal{G})$ -hybrid model against  
 652 a  $t'$ -adversary.*

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

## 658 5.2 AE-MPC

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

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

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

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



684 — **References** —

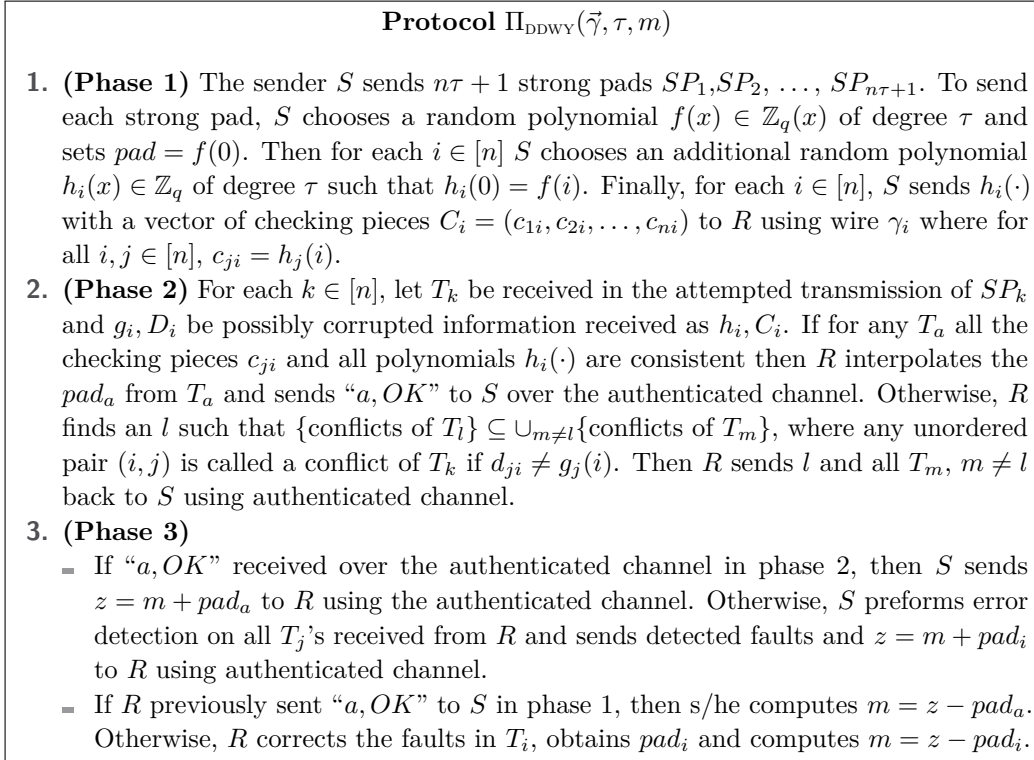
- 685 1 Saurabh Agarwal, Ronald Cramer, and Robbert de Haan. Asymptotically optimal two-round  
686 perfectly secure message transmission. In Cynthia Dwork, editor, *Advances in Cryptology -*  
687 *CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California,*  
688 *USA, August 20-24, 2006, Proceedings*, volume 4117 of *Lecture Notes in Computer Science*,  
689 pages 394–408. Springer, 2006. doi:10.1007/11818175\24.
- 690 2 Bar Alon, Eran Omri, and Anat Paskin-Cherniavsky. MPC with friends and foes. In Daniele  
691 Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th*  
692 *Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August*  
693 *17-21, 2020, Proceedings, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages  
694 677–706. Springer, 2020. doi:10.1007/978-3-030-56880-1\24.
- 695 3 Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic  
696 library with nested operations. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger,  
697 editors, *Proceedings of the 10th ACM Conference on Computer and Communications Security,*  
698 *CCS 2003, Washington, DC, USA, October 27-30, 2003*, pages 220–230. ACM, 2003. doi:  
699 10.1145/948109.948140.
- 700 4 Christian Badertscher, Ran Canetti, Julia Hesse, Björn Tackmann, and Vassilis Zikas. Universal  
701 composition with global subroutines: Capturing global setup within plain UC. In Rafael Pass  
702 and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC*  
703 *2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, volume 12552 of *Lecture*  
704 *Notes in Computer Science*, pages 1–30. Springer, 2020. doi:10.1007/978-3-030-64381-2\1.
- 705 5 Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction  
706 ledger: A composable treatment. In Jonathan Katz and Hovav Shacham, editors, *Advances*  
707 *in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa*  
708 *Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes*  
709 *in Computer Science*, pages 324–356. Springer, 2017. doi:10.1007/978-3-319-63688-7\11.
- 710 6 Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin. Secure computation  
711 without authentication. *J. Cryptol.*, 24(4):720–760, 2011. doi:10.1007/s00145-010-9075-9.
- 712 7 Zuzana Beerliová-Trubíniová, Matthias Fitzl, Martin Hirt, Ueli M. Maurer, and Vassilis Zikas.  
713 MPC vs. SFE: perfect security in a unified corruption model. In Ran Canetti, editor, *Theory of*  
714 *Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March*  
715 *19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 231–250. Springer,  
716 2008. doi:10.1007/978-3-540-78524-8\14.
- 717 8 Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-  
718 cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon,  
719 editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4,*  
720 *1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988. doi:10.1145/62212.62213.
- 721 9 Elette Boyle, Ran Cohen, Deepesh Data, and Pavel Hubáček. Must the communication  
722 graph of MPC protocols be an expander? In Hovav Shacham and Alexandra Boldyreva,  
723 editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology*  
724 *Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, volume  
725 10993 of *Lecture Notes in Computer Science*, pages 243–272. Springer, 2018. doi:10.1007/  
726 978-3-319-96878-0\9.
- 727 10 Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the  $o(\sqrt{n})$ -bit barrier: Byzantine  
728 agreement with polylog bits per party. In Avery Miller, Keren Censor-Hillel, and Janne H.  
729 Korhonen, editors, *PODC '21: ACM Symposium on Principles of Distributed Computing,*  
730 *Virtual Event, Italy, July 26-30, 2021*, pages 319–330. ACM, 2021. doi:10.1145/3465084.  
731 3467897.
- 732 11 Jan Camenisch, Stephan Krenn, Ralf Küsters, and Daniel Rausch. iuc: Flexible universal  
733 composability made simple. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in*  
734 *Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application*  
735 *of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings*,

- 736 *Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 191–221. Springer, 2019.  
737 doi:10.1007/978-3-030-34618-8\_7.
- 738 **12** Ran Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptol.*,  
739 13(1):143–202, 2000. doi:10.1007/s001459910006.
- 740 **13** Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols.  
741 In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October*  
742 *2001, Las Vegas, Nevada, USA*, pages 136–145. IEEE Computer Society, 2001. doi:10.1109/  
743 SFCS.2001.959888.
- 744 **14** Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols.  
745 Cryptology ePrint Archive, Report 2000/067, December 2005. Latest version at <https://ia.cr/2000/067>.  
746
- 747 **15** Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable  
748 security with global setup. In Salil P. Vadhan, editor, *Theory of Cryptography, 4th Theory of*  
749 *Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007,*  
750 *Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 61–85. Springer, 2007.  
751 doi:10.1007/978-3-540-70936-7\_4.
- 752 **16** Nishanth Chandran, Pouyan Forghani, Juan Garay, Rafail Ostrovsky, Rutvik Patel, and  
753 Vassilis Zikas. Universally composable almost-everywhere secure computation. Cryptology  
754 ePrint Archive, Report 2021/1398, 2021. <https://ia.cr/2021/1398>.
- 755 **17** Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Improved fault tolerance and  
756 secure computation on sparse networks. In Samson Abramsky, Cyril Gavoille, Claude Kirchner,  
757 Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and*  
758 *Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10,*  
759 *2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 249–260.  
760 Springer, 2010. doi:10.1007/978-3-642-14162-1\_21.
- 761 **18** David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure  
762 protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM*  
763 *Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19.  
764 ACM, 1988. doi:10.1145/62212.62214.
- 765 **19** Danny Dolev. Unanimity in an unknown and unreliable environment. In *22nd Annual*  
766 *Symposium on Foundations of Computer Science, Nashville, Tennessee, USA, 28-30 October*  
767 *1981*, pages 159–168. IEEE Computer Society, 1981. doi:10.1109/SFCS.1981.53.
- 768 **20** Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message  
769 transmission. In *31st Annual Symposium on Foundations of Computer Science, St. Louis,*  
770 *Missouri, USA, October 22-24, 1990, Volume I*, pages 36–45. IEEE Computer Society, 1990.  
771 doi:10.1109/FSCS.1990.89522.
- 772 **21** Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. Fault tolerance in networks  
773 of bounded degree (preliminary version). In Juris Hartmanis, editor, *Proceedings of the 18th*  
774 *Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California,*  
775 *USA*, pages 370–379. ACM, 1986. doi:10.1145/12130.12169.
- 776 **22** Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential  
777 consensus. In Elizabeth Borowsky and Sergio Rajsbaum, editors, *Proceedings of the Twenty-*  
778 *Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston,*  
779 *Massachusetts, USA, July 13-16, 2003*, pages 211–220. ACM, 2003. doi:10.1145/872035.  
780 872066.
- 781 **23** Matthias Fitzi, Martin Hirt, and Ueli M. Maurer. Trading correctness for privacy in uncondi-  
782 tional multi-party computation (extended abstract). In Hugo Krawczyk, editor, *Advances in*  
783 *Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara,*  
784 *California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer*  
785 *Science*, pages 121–136. Springer, 1998. doi:10.1007/BFb0055724.
- 786 **24** Juan A. Garay, Aggelos Kiayias, and Hong-Sheng Zhou. A framework for the sound specification  
787 of cryptographic tasks. In *Proceedings of the 23rd IEEE Computer Security Foundations*

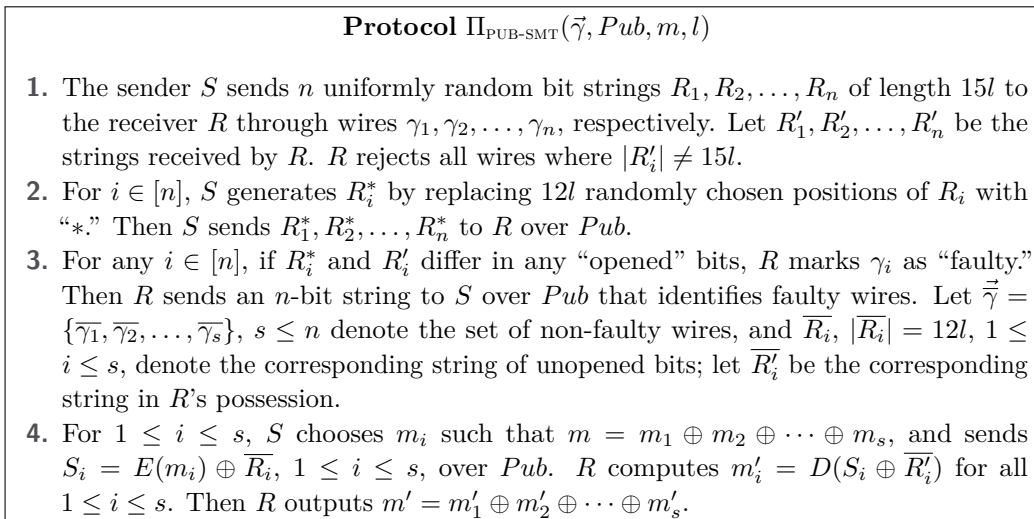
- 788 *Symposium, CSF 2010, Edinburgh, United Kingdom, July 17-19, 2010*, pages 277–289. IEEE  
789 Computer Society, 2010. doi:10.1109/CSF.2010.26.
- 790 25 Juan A. Garay and Rafail Ostrovsky. Almost-everywhere secure computation. In Nigel P.  
791 Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International*  
792 *Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey,*  
793 *April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages  
794 307–323. Springer, 2008. doi:10.1007/978-3-540-78967-3\_18.
- 795 26 Juan A. Garay and Kenneth J. Perry. A continuum of failure models for distributed computing.  
796 In Adrian Segall and Shmuel Zaks, editors, *Distributed Algorithms, 6th International Workshop,*  
797 *WDAG '92, Haifa, Israel, November 2-4, 1992, Proceedings*, volume 647 of *Lecture Notes in*  
798 *Computer Science*, pages 153–165. Springer, 1992. doi:10.1007/3-540-56188-9\_11.
- 799 27 Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A  
800 completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings*  
801 *of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York,*  
802 *USA*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.
- 803 28 Jacopo Griggio. *Perfectly secure message transmission protocols with low communication*  
804 *overhead and their generalization*. PhD thesis, Universiteit Leiden, 2012.
- 805 29 Martin Hirt and Ueli M. Maurer. Complete characterization of adversaries tolerable in  
806 secure multi-party computation (extended abstract). In James E. Burns and Hagit Attiya,  
807 editors, *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed*  
808 *Computing, Santa Barbara, California, USA, August 21-24, 1997*, pages 25–34. ACM, 1997.  
809 doi:10.1145/259380.259412.
- 810 30 Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. In Henri Gilbert, editor, *Advances*  
811 *in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and*  
812 *Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010.*  
813 *Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 466–485. Springer,  
814 2010. doi:10.1007/978-3-642-13190-5\_24.
- 815 31 Dennis Hofheinz and Victor Shoup. GNUC: A new universal composability framework. *J.*  
816 *Cryptol.*, 28(3):423–508, 2015. doi:10.1007/s00145-013-9160-y.
- 817 32 Siddhartha Jayanti, Srinivasan Raghuraman, and Nikhil Vyas. Efficient constructions for  
818 almost-everywhere secure computation. In Anne Canteaut and Yuval Ishai, editors, *Advances*  
819 *in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory*  
820 *and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings,*  
821 *Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 159–183. Springer, 2020.  
822 doi:10.1007/978-3-030-45724-2\_6.
- 823 33 Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable  
824 synchronous computation. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of*  
825 *Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume  
826 7785 of *Lecture Notes in Computer Science*, pages 477–498. Springer, 2013. doi:10.1007/  
827 978-3-642-36594-2\_27.
- 828 34 Valerie King and Jared Saia. From almost everywhere to everywhere: Byzantine agreement  
829 with  $\tilde{O}(n^{3/2})$  bits. In Idit Keidar, editor, *Distributed Computing, 23rd International Symposium,*  
830 *DISC 2009, Elche, Spain, September 23-25, 2009. Proceedings*, volume 5805 of *Lecture Notes*  
831 *in Computer Science*, pages 464–478. Springer, 2009. doi:10.1007/978-3-642-04355-0\_47.
- 832 35 Kaoru Kurosawa and Kazuhiro Suzuki. Truly efficient 2-round perfectly secure message  
833 transmission scheme. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT*  
834 *2008, 27th Annual International Conference on the Theory and Applications of Cryptographic*  
835 *Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in*  
836 *Computer Science*, pages 324–340. Springer, 2008. doi:10.1007/978-3-540-78967-3\_19.
- 837 36 Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The byzantine generals problem.  
838 *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982. URL: [http://doi.acm.org/10.1145/](http://doi.acm.org/10.1145/357172.357176)  
839 [357172.357176](http://doi.acm.org/10.1145/357172.357176), doi:10.1145/357172.357176.

- 840 37 Ueli Maurer and Renato Renner. Abstract cryptography. In Bernard Chazelle, editor,  
841 *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January*  
842 *7-9, 2011. Proceedings*, pages 1–21. Tsinghua University Press, 2011. URL: [http://conference.](http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/14.html)  
843 [iiis.tsinghua.edu.cn/ICS2011/content/papers/14.html](http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/14.html).
- 844 38 Jesper Buus Nielsen. *On Protocol Security in the Cryptographic Model*. PhD thesis, University  
845 of Aarhus, 2003.
- 846 39 Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence  
847 of faults. *J. ACM*, 27(2):228–234, 1980. URL: <http://doi.acm.org/10.1145/322186.322188>,  
848 [doi:10.1145/322186.322188](https://doi.org/10.1145/322186.322188).
- 849 40 Hasan Md. Sayeed and Hosame Abu-Amara. Efficient perfectly secure message transmission  
850 in synchronous networks. *Inf. Comput.*, 126(1):53–61, 1996. [doi:10.1006/inco.1996.0033](https://doi.org/10.1006/inco.1996.0033).
- 851 41 Gabriele Spini and Gilles Zémor. Perfectly secure message transmission in two rounds. In  
852 Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International*  
853 *Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings,*  
854 *Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 286–304, 2016. [doi:](https://doi.org/10.1007/978-3-662-53641-4_12)  
855 [10.1007/978-3-662-53641-4\\_12](https://doi.org/10.1007/978-3-662-53641-4_12).
- 856 42 K. Srinathan, Arvind Narayanan, and C. Pandu Rangan. Optimal perfectly secure message  
857 transmission. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th*  
858 *Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19,*  
859 *2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 545–561. Springer,  
860 2004. [doi:10.1007/978-3-540-28628-8\\_33](https://doi.org/10.1007/978-3-540-28628-8_33).
- 861 43 Eli Upfal. Tolerating linear number of faults in networks of bounded degree. In Norman C.  
862 Hutchinson, editor, *Proceedings of the Eleventh Annual ACM Symposium on Principles of*  
863 *Distributed Computing, Vancouver, British Columbia, Canada, August 10-12, 1992*, pages  
864 83–89. ACM, 1992. [doi:10.1145/135419.135437](https://doi.org/10.1145/135419.135437).
- 865 44 Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In  
866 Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber, editors,  
867 *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982,*  
868 *San Francisco, California, USA*, pages 128–136. ACM, 1982. [doi:10.1145/800070.802185](https://doi.org/10.1145/800070.802185).

**A** Functionalities and Protocols



■ **Figure 3** The SMT protocol from [20].



■ **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  $\vec{W} = (W_1, \dots, W_n)$ . At the first activation, verify that  $sid = (P_s, P_r, sid')$ , where  $\{P_s, P_r\} = \{S, R\}$ . Initialize variables  $m_1, \dots, m_n$  to  $\perp$ .

- Upon receiving input (SEND,  $sid, W_i, v_i$ ) from  $P_s$  in round  $\rho$  (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 (SENDALEAK,  $sid, W_i, m_i$ ) to the adversary; otherwise send (SENDALEAK,  $sid, W_i, l(m_i)$ ).
- Upon receiving (INFLSEND,  $sid, W_i, m'_i$ ) from the adversary: If any  $P \in \{P_s, P_r, W_i\}$  is corrupted, and (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  $\rho'$ : If  $P_r$  is corrupted, then send (FETCHLEAK,  $sid, W_i$ ) to the adversary; otherwise, if  $\rho' = \rho + 2$ , then output (SENT,  $sid, W_i, m_i$ ) to  $P_r$  if it has not yet been sent.
- Upon receiving (OUTPUT,  $sid, W_i$ ) from the adversary: If  $P_r$  is corrupted, then output (SENT,  $sid, W_i, m_i$ ) to  $P_r$  if it has not yet been sent.
- Upon receiving (CORRUPT,  $sid, P$ ) from the adversary for  $P \in \{P_s, P_r, W_1, \dots, W_n\}$ , mark  $P$  as corrupted. If  $P$  is some wire-party  $W_i$ , then send (SENDALEAK,  $sid, m_i$ ) to the adversary; otherwise, send (SENDALEAK,  $sid, m_1, \dots, m_n$ ). 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  $\rho$ , record  $m \leftarrow v$ . If  $P_i$  or  $P_j$  is marked as corrupted, then send (SENDALEAK,  $sid, m$ ) to the adversary; otherwise send (SENDALEAK,  $sid, l(m)$ ).
- Upon receiving (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 (CORRUPT,  $sid, P$ ) from the adversary for  $P \in \{P_i, P_j\}$ , mark  $P$  as corrupted and send (SENDALEAK,  $sid, m$ ) to the adversary.

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

## B Proofs

870

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

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

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

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

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

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

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

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

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

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

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



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

965 **Proof of Theorem 16.** We first prove that  $\mathcal{D}'$  is AE-monotone. Suppose that  $(T_i, D_i) \in \mathcal{D}'$   
966 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  $D \subseteq D(T_i)$   
967 and  $(T_i \cup D(T_i), A) \in \mathcal{D}$ . We want to show that  $(T_j, D_i) \in \mathcal{D}'$ , and it suffices to show  
968 that  $D \subseteq D(T_j)$  and  $(T_j \cup D(T_j), A) \in \mathcal{D}$ . Since  $D(T_i) \subseteq D(T_j)$  (using the fact that  
969  $\mathcal{D}_1, \dots, \mathcal{D}_m$  are all AE-monotone), it follows that  $D \subseteq D(T_j)$ . On the other hand, since  
970  $T_i \cup D(T_i) \subseteq T_j \cup D(T_j)$ , it follows that  $(T_j \cup D(T_j), A) \in \mathcal{D}$  (using the fact that  $\mathcal{D}$  is  
971 AE-monotone and that  $T_j \cup D(T_j) \in \mathcal{T}$ ). We now prove the security of the compiled protocol.

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

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

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

993 Suppose for a contradiction that there is an environment  $\mathcal{Z}'$  such that  $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathcal{D}'}(\mathcal{F}), \mathcal{S}', \mathcal{Z}'} \not\equiv$   
994  $\text{EXEC}_{\mathcal{C}^{\mathcal{D}_1, \dots, \mathcal{D}_m}(\Pi), \mathcal{D}, \mathcal{Z}'}$ . Then, we construct an environment  $\mathcal{Z}$  such that  $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathcal{D}}(\mathcal{F}), \mathcal{S}, \mathcal{Z}} \not\equiv$   
995  $\text{EXEC}_{\Pi, \mathcal{D}, \mathcal{Z}}$ . The environment  $\mathcal{Z}$  will simulate an interaction between  $\mathcal{Z}'$  and  $\mathcal{D}$ , and output  
996 whatever  $\mathcal{Z}'$  outputs, as well as do some additional processing. Whenever  $\mathcal{Z}'$  instructs its  
997 dummy adversary to deliver a message to an instance of  $\mathcal{W}_{\text{AE}}^{\mathcal{D}_i}(\mathcal{F}_i)$ , this is translated by  $\mathcal{Z}$   
998 into a delivery request for a corresponding instance of  $\mathcal{F}_i$  and forwarded to the external  
999 adversary (either  $\mathcal{S}$  or  $\mathcal{D}$ ), except that a request to deliver a DOOM message is converted into  
1000 a request to deliver a CORRUPT message if allowed by  $\mathcal{D}_i$  and dropped otherwise. Corruption  
1001 requests directed to parties are forwarded to the external adversary unmodified.

1002 Next, whenever  $\mathcal{Z}$  receives subroutine output from the external adversary, this is forwarded  
1003 to  $\mathcal{Z}'$ , except that reported messages from instances of  $\mathcal{W}_{\text{AE}}^{\mathcal{D}_i}(\mathcal{F}_i)$  are translated into reported  
1004 messages from corresponding instances of  $\mathcal{F}_i$ . Finally,  $\mathcal{Z}$  simply relays inputs and outputs  
1005 between  $\mathcal{Z}'$  and parties. We conclude by claiming that  $\text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathcal{D}'}(\mathcal{F}), \mathcal{S}', \mathcal{Z}'} \equiv \text{IDEAL}_{\mathcal{W}_{\text{AE}}^{\mathcal{D}}(\mathcal{F}), \mathcal{S}, \mathcal{Z}}$   
1006 and  $\text{EXEC}_{\mathcal{C}^{\mathcal{D}_1, \dots, \mathcal{D}_m}(\Pi), \mathcal{D}, \mathcal{Z}'} \equiv \text{EXEC}_{\Pi, \mathcal{D}, \mathcal{Z}}$ . Indeed, if  $\mathcal{Z}$  interacts with  $\mathcal{W}_{\text{AE}}^{\mathcal{D}}(\mathcal{F})$  and  $\mathcal{S}$ , then  
1007 the view of the simulated  $\mathcal{Z}'$  within  $\mathcal{Z}$  is identical to the view of  $\mathcal{Z}'$  when interacting with  
1008  $\mathcal{W}_{\text{AE}}^{\mathcal{D}'}(\mathcal{F})$  and  $\mathcal{S}'$ , and similarly if  $\mathcal{Z}$  interacts with  $\Pi$  and  $\mathcal{D}$ , then the view of the simulated  
1009  $\mathcal{Z}'$  within  $\mathcal{Z}$  is identical to the view of  $\mathcal{Z}'$  when interacting with  $\mathcal{C}^{\mathcal{D}_1, \dots, \mathcal{D}_m}(\Pi)$  and  $\mathcal{D}$ . ◀