Topology-Hiding Computation From Key Agreement in Diameter-Two Graphs

D'or Banoun* Elette Boyle† Ran Cohen‡

May 19, 2026

Abstract

Topology-hiding computation (THC) enables a set of parties, communicating over an incomplete network, to execute a secure multiparty computation (MPC) protocol for securely computing a function, while also hiding the network topology from within a given class of graphs. Semi-honest THC can be achieved over arbitrary graph classes, facing an arbitrary number of corruptions, from various assumptions implying oblivious transfer (OT). These assumptions are justified by strong lower bounds, indicating that 2-secure topology-hiding broadcast (THB) over certain diameter-3 graph classes requires OT, as well as 1-secure THC over certain diameter-2 graph classes of variable size.

While THC from weaker assumptions, such as key agreement (KA), is achievable for $t = 1$ over fixed-size graphs, the case of multiple corruptions remains unclear, with no known candidate constructions. Even in the simpler, privacy-free case of THB, tolerating $t > 1$ corruptions without assuming OT is only known for "friendship" graphs (which are diameter-2 graphs of a certain form): in fact, the latter holds information theoretically and for $t < n$.

The state of the art raises two foundational questions: First, considering THC, is OT necessary for protecting against adversaries with multiple points of view in the graph? Second, considering THB, is there a zero-one law for $t > 1$, where given a graph class, THB either holds unconditionally or requires OT?

In this work we study these questions over graphs of diameter 2 (in which the lower bounds requiring OT do not hold) and provide THC protocols for various graph classes supporting many corruptions assuming KA. For some of these results we obtain optimal resilience assuming KA: $t < n$ for THB and $t < n/2$ for THC. We also present new lower bounds, showing that in certain graph classes that support information-theoretic THB for $t = 1$, KA is necessary for $t = 2$.

*Reichman University. E-mail: dor295@gmail.com. Research supported in part by AFOSR Award FA9550-21-1-0046 and by ISF grant 1834/23.

†NTT Research and Reichman University. E-mail: elette.boyle@runi.ac.il. Research supported in part by AFOSR Award FA9550-21-1-0046.

‡Reichman University. E-mail: cohenran@runi.ac.il. Research supported in part by NSF grant no. 2055568 and by ISF grant 1834/23.

Contents

1 Introduction **1**
- 1.1 Our Results . . . 3
- 1.2 Technical Overview . . . 7

2 Preliminaries **15**
- 2.1 Key-Agreement Protocols . . . 15
- 2.2 Topology-Hiding Computation (THC) . . . 16

3 Topology-Hiding RMT from Linear MPC **20**
- 3.1 RMT Protocol in SFE-Hybrid Model . . . 21
- 3.2 Instantiating for d-Validated Graphs . . . 23
- 3.3 Instantiating for Odd-Validated Graphs . . . 26

4 Topology-Hiding RMT from Non-Linear MPC **28**
- 4.1 RMT Protocol in SFE-Hybrid Model with an Honest Majority . . . 28
- 4.2 Instantiating for Degree-Bounded Graphs . . . 30

5 Lower Bound **32**

Bibliography **38**

A UC Framework **40**

1 Introduction

Secure multiparty computation (MPC) [Yao86, GMW87, BGW88, CCD88] enables a set of n parties to compute a function over their inputs, such that no adversary, corrupting up to t parties, can violate correctness and privacy. Topology-hiding computation (THC) [MOR15] additionally maintains privacy of the underlying *communication graph* from within a given, publicly known graph class. This is formulated via the simulation paradigm, by requiring that everything that the adversary learns from the computation can be simulated given the corrupted parties' inputs, outputs, local neighbor-sets, and the graph class. In this paper, we follow current THC literature, and consider semi-honest adversaries that do not deviate from the protocol but try to break privacy.

The research on THC draws parallel lines with the evolution of MPC in the '80s. For MPC, after establishing feasibility facing any number of corruptions assuming oblivious transfer (OT) [Yao86, GMW87], the community sought for justification for this assumption. First, it was shown that cryptographic assumptions can be traded with an honest majority [BGW88, CCD88], and later it was shown that for many two-party functions, every function is either trivial (and can be computed information-theoretically) or complete (and in particular implies OT) [Kil91, BMM99, Kil00]. That is, assumptions weaker than OT cannot be used for constructing MPC protocol for these functions.

THC also began with strong feasibility for any graph class, facing any number of corruptions under assumptions that imply OT. However, trading cryptographic assumptions with an honest majority turned out to be more challenging, with many lower bounds even facing a single corruption and with privacy-free functions. In fact, when the adversary can corrupt 2 parties and obtain multiple viewpoints on the graph, all existing constructions are either degenerate (the local neighborhoods of any choice of corrupted parties already uniquely identifies the full graph) or rely on assumptions that imply OT.

In this work, we study whether cryptographic assumptions weaker than OT can be used for non-degenerate THC. We answer this question in the affirmative, showing that for many diameter-2 graph classes $\mathcal{G}$, THC for $\mathcal{G}$ is equivalent to *key agreement*.

Background. The initial research on THC focused on *feasibility* facing *any* number of corruptions. The blueprint that underlies all generic feasibility results, presented in [MOR15], reduced the problem to the feasibility of topology-hiding *broadcast* (THB), where all parties learn the input bit of a designated sender in a topology-hiding manner. Indeed, given THB, it is possible to construct THC from the assumptions needed for semi-honest MPC in the broadcast model: that is, from key agreement (KA) in the honest-majority setting (needed to establish secure point-to-point channels) and from oblivious transfer (OT) in the dishonest-majority setting. A fruitful line of work constructed THB (hence also THC) for arbitrary graph classes, secure against *any* number of corruptions, from assumptions that imply OT [MOR15, HMTZ16, AM17, ALM17, BBMM18, LZM+18, LZM+20, Li22, BBKM23]. Recently, new feasibility results in "standard" MPC were achieved from new THC protocols based on OT [BCM26] (on graphs of diameter 2).

The OT assumption was justified by [BBMM18], who showed that without an honest majority, THB on even a very simple graph-class already implies OT. Namely, a 2-secure THB on the 4-node graph class of diameter 3, consisting of 4 parties connected in a line or in a square, implies the existence of OT; see Figure 1.[1]

[1] A note regarding notation: We consider *labelled graphs*, where each of the n nodes is assigned with a label from

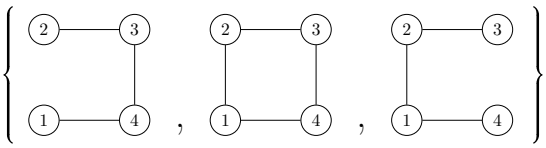


Figure 1: The diameter-3 graph class from [BBMM18], where a 2-secure THB implies OT.

A more recent research direction aimed to understand the minimal assumptions needed for THB and THC for restricted graph classes, seeking constructions from assumptions that do not imply OT, and even potentially information-theoretic (IT) results. The main focus in this thread of research was on a *single* semi-honest corruption, as several results showed that even in this case, and even for the privacy-free task of broadcast, cryptography is necessary for simple graph classes [BBC+19, BBC+20, BBC24]. The state of the art can be summarized as follows:

- Generic THC over 1-connected graphs requires KA [BBC+19]. Further, generic THC over certain variable-size graphs requires infinitely often OT [BBC+20]. The former was shown for 1-secure anonymous broadcast for 3 parties on a line; the latter for 1-secure anonymous broadcast over either 2 parties in a line or 3 parties in a line, see Figure 2(a).

  THC from KA is only known for 2-connected graphs of a known size $n \geq 3$ and for $t = 1$ [BBC+20]. Information-theoretic THC (IT-THC) with *perfect* security is only known for cycles [BBC+19] (see Figure 2(b)) and with *statistical* security for 2-connected graphs [BBC+20]; again, both results hold for connected graphs of a known size $n \geq 3$ and for $t = 1$.

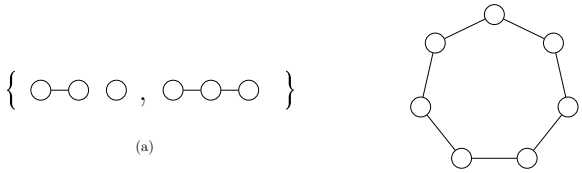


Figure 2: (a) A graph class of variable size where 1-secure THC implies infinitely often OT. (b) A 7-cycle graph, where IT-THC is possible for $t = 1$, and THB for $t = 2$ implies KA.

- The feasibility landscape of THB is more involved: while 1-secure THB of a 4-party line (which is 1-connected and of diameter 3, see Figure 3(a)) requires KA [BBC+19], 1-secure IT-THB can be achieved over 2-connected graphs [BBC+20] and over certain 1-connected graphs of diameter 2 [BBC+20, BBC24] (namely, certain subgraphs of the wheel graph which are *star-embedded*, i.e., where a single center node is connected to all others, see Figure 3(b)); further, certain graphs of variable size can also be supported.

  Recently, [BBC24] presented the first feasibility of IT-THB beyond a single corruption (in fact for any $t < n$), for the class of *friendship* graphs (which are graphs of diameter 2, consisting

[n], and all labels are distinct; see, e.g., Figure 1. In case we do not write down the labels, e.g., Figure 2, we consider the graph class to consist of *all* $n!$ graphs corresponding to all possible assignments of the labels.

of multiples triangles "glued" together at a central node [ERS66], see Figure 3(c)); however, their result does not extend to IT-THC, as this class is 1-connected.

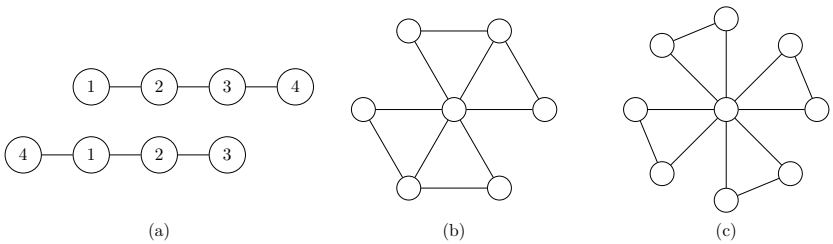


Figure 3: (a) A line of length 4, where 1-secure THB implies KA. (b) A star-embedded graph class, admitting 1-secure IT-THB. (c) A friendship graph class, admitting t-secure IT-THB for any $t < n$.

Main question. When considering more than a single corruption, the feasibility landscape of THB may suggest toward a zero-one law: for each graph class, either information-theoretic security is known to be possible, or feasibility is only known given assumptions implying OT. Further, all known (non-degenerate) THC constructions rely on assumptions implying OT.

Presently, only one approach has been shown for exploiting the power of KA toward THC (the "dead-end channels" technique from [BBC+20]). However, this technique strongly relies on the promise of a *single* corrupted party, and completely breaks once the adversary obtains two points of view in the graph. Indeed, the single-corruption setting greatly simplifies the design challenge in a fashion amenable to the guarantees offered by KA, where two parties can privately communicate over a third party, and an honest majority is guaranteed in every such triplet. It is not clear whether KA alone can suffice to support security against a multi-corruption scenario.

Our main question is therefore:

> *When stepping beyond a single corruption, are there graph classes for which THB exists from KA, but cannot be achieved information theoretically?*

Any affirmative answer to this question would readily imply THC under the same assumptions, and as long as the number of nodes is fixed and known, and that a majority of the nodes are honest.

1.1 Our Results

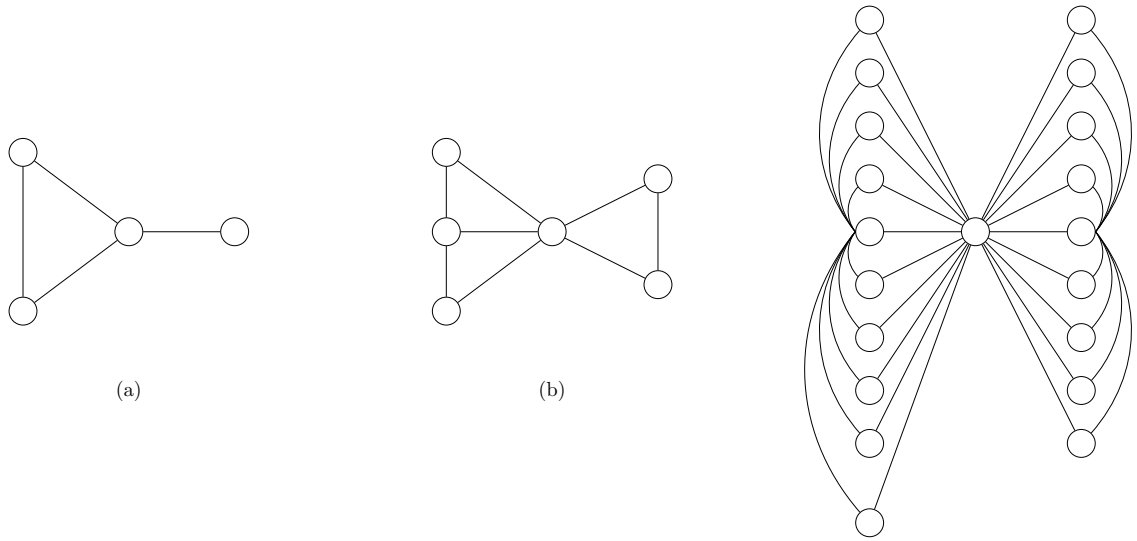
We conduct a systematic study of the feasibility of THB and THC over *diameter*-2 graphs classes from KA, facing more than one corruption. Ideally, we aim to capture THB for $t < n$ and THC for $t < n/2$ (recall that dishonest-majority MPC requires OT even when it is topology revealing). Diameter-2 graphs are interesting as they capture many of the candidate topologies for feasibility/impossibility in prior work [BBC+20, BBC24]), and are even more relevant in our context by the impossibility result of [BBMM18], showing that 2-secure THB in simple graphs of diameter-3 requires OT.

Necessity of KA. First, to confirm that our results are non-trivial in the sense of [Kil91] (i.e., do not hold information theoretically), we present a family of diameter-two graph classes for which a t-secure THB implies KA.

For $t \in \mathbb{N}$ and $n = 2(t+1)$ we denote by $\mathcal{G}_{\text{join-stars}}(n)$ the graph class consisting of a center node that is connected to every node in the disjoint union of two stars, one with t nodes and the other with $t+1$, see Figure 4(c).[2] Each graph class consists of all possible labeling for the n nodes.

Theorem 1.1 (necessity of KA, informal)**.** *Let $t \in \mathbb{N}$ and let $n = 2(t+1)$. A t-secure THB for $\mathcal{G}_{\text{join-stars}}(n)$ implies KA.*

Note that by construction, the graphs in $\mathcal{G}_{\text{join-stars}}(n)$ are star embedded, i.e., admit a single center node that is connected to all others. For the special case of $t = 1$, Theorem 1.1 thus implies the result from [BBC+20], showing that 1-secure THB for the 4-nodes "paw graph" implies KA, see Figure 4(a). Before focusing on the case $t = 2$, note that [BBC24] defined an *admissible* graph to be a subgraph of a wheel graph that is star-embedded graph and in which the degree of every non-center node is 2 or 3 (i.e., there are no "tails" that are connected only to the center), and showed that such admissible graphs admit 1-secure IT-THB. It follows that for $t = 2$, $\mathcal{G}_{\text{join-stars}}(6)$ consists of admissible 6-nodes graphs that admit 1-secure IT-THB and for which 2-secure THB requires KA, see Figure 4(b).



(a) (b) (c)

Figure 4: (a) For $t = 1$ the graph class $\mathcal{G}_{\text{join-stars}}(4)$ corresponds to the paw graph, where 1-secure THB implies KA. (b) For $t = 2$ the graph class $\mathcal{G}_{\text{join-stars}}(6)$ admits 1-secure IT-THB but 2-secure THB implies KA. (c) For $t \in \mathbb{N}$, t-secure THB for $\mathcal{G}_{\text{join-stars}}(2t+2)$ implies KA. Assuming KA, these graph classes admit t-secure THB for any $t < n$.

We note that our positive results below (Theorem 1.3) show that every such graph class *admits* THB assuming KA for any number of corruptions, showing tightness of this lower bound. We refer the reader to Section 1.2.2 for an overview of the techniques in the case $t = 2$, and to Section 5 for the general case.

[2] That is, the *join* operation of a single node with the disjoint union of two stars.

Sufficiency of KA. We proceed to present our feasibility results. We show that for a large range of diameter-2 graph classes, it is possible to construct THB from KA for $t < n$. Namely, for graphs in which for every pair of non-neighboring parties P_i and P_j, the number of common neighbors is *fixed and known*, or potentially *unknown but odd*. Given a graph G, denote by $\mathcal{N}_G(i)$ the neighbor-set of P_i, and by $\mathsf{CommonNbr}_G(i,j) := \mathcal{N}_G(i) \cap \mathcal{N}_G(j)$ the set of common neighbors of P_i and P_j.

Theorem 1.2 (THB/THC classes from KA: common-neighbor condition, informal)**.** *Let $\mathcal{G}$ be an n-node graph class of diameter 2. Suppose that exactly one of the following conditions holds:*

- *either there exists a known constant $d \in \mathbb{N}$ such that for every pair of parties P_i and P_j and for every $G \in \mathcal{G}$, if P_i and P_j are non-neighbors in G, then $|\mathsf{CommonNbr}_G(i,j)| = d$;*
- *or for every pair of parties P_i and P_j and for every $G \in \mathcal{G}$, if P_i and P_j are non-neighbors in G, then $|\mathsf{CommonNbr}_G(i,j)|$ is an odd number.*

Then, there exists t-secure THB for $t < n$ assuming KA. Further, there exists t-secure THC for $t < n/2$ assuming KA.

Theorem 1.2 covers many interesting graphs classes, e.g., star-embedded graphs in which $|\mathsf{CommonNbr}_G(i,j)| = 1$ for every non-neighboring P_i and P_j (see Figure 5(a)), or graphs with a fixed number of "centers" (see Figure 5(b) for a graph with two centers). Figure 5(c) illustrates a graph class where every pair of non-neighboring parties has either 1 or 3 common neighbors.

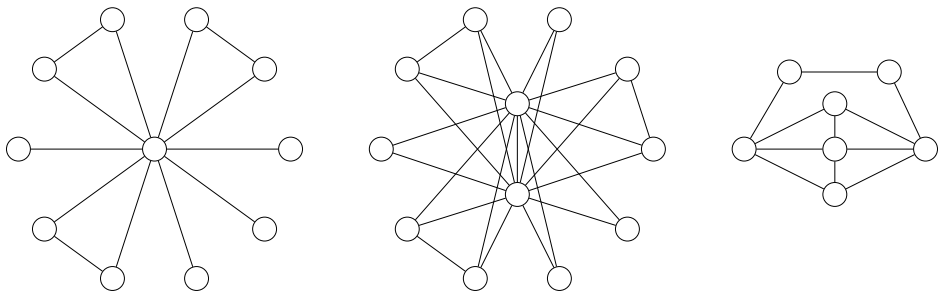


Figure 5: (a) A star-embedded graph class where every pair of non-neighboring parties have exactly one common neighbor. (b) Here every pair of non-neighboring parties have exactly two common neighbors. (c) Unknown-but-odd amount of common neighbors between every non-neighboring parties. These graph classes admit THB assuming KA for $t < n$.

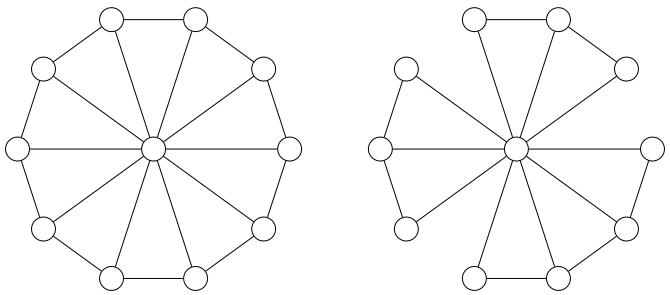
In fact, we can generalize Theorem 1.2 to the case where the number of common neighbors may be unknown, but there are d common neighbors that can self-validate themselves as "special," or if the number of self-validating "special" parties is unknown-but-odd. This generalization captures, for example, the case of a star-embedded graph in which there are nodes of degree 3 (see Figure 6). In such a case there may be non-neighboring nodes in that have a single common neighbor (the center), and there may be non-neighboring nodes that have two common neighbors; however, in the latter case, the center node can self-validate itself as the center and play differently than the

non-center node. In particular, this captures the graph class from Theorem 1.1, showing that KA is sufficient and necessary for this class.

Theorem 1.3 (THB/THC classes from KA: self-validating condition, informal)**.** *Let $\mathcal{G}$ be an n-node graph class of diameter 2. Suppose that exactly one of the following conditions holds:*

- *either there exists a known constant $d \in \mathbb{N}$ such that for every pair of parties P_i and P_j and for every $G \in \mathcal{G}$, if P_i and P_j are non-neighbors in G, then there are exactly d parties in $\mathsf{CommonNbr}_G(i,j)$ that can self-validate as special;*
- *or for every pair of parties P_i and P_j and for every $G \in \mathcal{G}$, if P_i and P_j are non-neighbors in G, then there is an odd number of parties in $\mathsf{CommonNbr}_G(i,j)$ that can self-validate as special.*

Then, there exists t-secure THB for $t < n$ assuming KA. Further, there exists t-secure THC for $t < n/2$ assuming KA.



(a) (b)

Figure 6: (a) A wheel graph W_7. (b) An admissible graph with some degree-3 nodes. Although the number of common neighbors is not fixed-and-known or unknown-but-odd, these graph classes admit THB assuming KA for $t < n$.

However, Theorem 1.3 leaves open the setting where for some pairs of non-neighboring parties, the number of common neighbors (or self-validating special parties) is unknown and potentially even. Our next result shows feasibility as long as such parties have sufficiently many honest neighbors; see Figure 7 for examples.

Theorem 1.4 (THB/THC classes from KA: degree condition, informal)**.** *Let $\mathcal{G}$ be an n-node graph class of diameter 2, and let d_{max} mark the maximal degree and d_{min} the minimal degree in the graph class. If it holds that $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$, then there exists t-secure THB assuming KA. Further, as in this case it must hold that $t < n/2$, there exists t-secure THC assuming KA.*

Theorems 1.3 and 1.4 capture a large fraction of diameter-2 graph classes, leaving open the case where for some pairs of non-neighboring parties, the number of common neighbors is unknown and potentially even, and there is no guarantee on the size of their neighbor sets or on the existence of an honest majority among them. For completeness, we summarize our results compared with prior works for THC in Table 1, and for THB in Table 2.

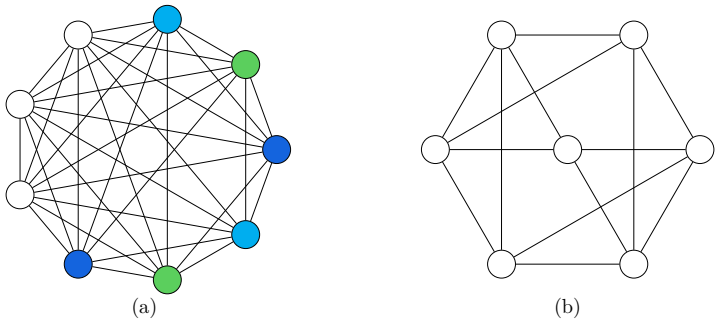


Figure 7: (a) This 9-nodes graph class consists of nearly complete graphs with three missing edges (note that same-color nodes are not connected). Here, $d_{\mathsf{min}} = 7$ and $d_{\mathsf{max}} = 8$, and there are parties for which the number of common neighbors is unknown and is potentially even (either 7 or 8). THB and THC exist assuming KA for $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2 = 3$. (b) In this 7-nodes graph class $d_{\mathsf{min}} = d_{\mathsf{max}} = 4$, and between every two non-neighboring nodes there are 3 or 4 common neighbors. THB and THC exist assuming KA for $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2 = 2$.

	assumption	**diameter**	**graph class**	**threshold**	
UB	implying OT	any	poly-size	$t < n$	[MOR15, ALM17, HMTZ16] [LZM+18, BBKM23]
	KA	**2**	**common-neighbor condition**	$\boldsymbol{t < n/2}$	**Thm 1.3**
	KA	**2**	**bounded-degree condition**	$\boldsymbol{t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2}$	**Thm 1.4**
	KA	any	poly-size	$t = 1$	[BBC+20]
	IT (statistical)	any	2-connected	$t = 1$	[BBC+20]
	IT (perfect)	any	cycles, e.g., $\{\hexagon\}$	$t = 1$	[BBC+19]
LB	OT	3	$\{\square, \sqsubset\}$	$n = 4, t = 2$	[BBMM18]
	ioOT	2	$\{\urcorner, \ulcorner\}$	$n \leq 3, t = 1$	[BBC+20]
	KA	2	$\{\ulcorner\}$	$n = 3, t = 1$	[BBC+20]

Table 1: Assumption needed for THC. For upper bounds (UB) the listed assumptions are sufficient for general THC whereas for lower bounds (LB) the listed assumptions are necessary.

1.2 Technical Overview

We proceed to describe our techniques: in Section 1.2.1 we explain the new feasibility results and in Section 1.2.2 the impossibility results.

1.2.1 Feasibility Results

Our starting point is the reduction used in all OT-based THC protocols, of initially constructing a THB protocol and then execute a OT-based MPC protocol designed for the broadcast model. As we rely on the existence of a key-agreement protocol, we can establish pairwise secure channels on top of broadcast, and execute generic honest-majority MPC designed for secure point-to-point

	assumption	diameter	graph class	threshold	
UB	implying OT	any	poly-size	$t < n$	[MOR15, ALM17, HMTZ16] [LZM+18, BBKM23]
	KA	**2**	**common-neighbor condition**	$\boldsymbol{t < n}$	**Thm 1.3**
	KA	**2**	**bounded-degree condition**	$\boldsymbol{t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2}$	**Thm 1.4**
	KA	any	poly-size	$t = 1$	[BBC+20]
	IT (statistical)	any	2-connected	$t = 1$	[BBC+20]
	IT (perfect)	any	cycles, e.g., {⬠}	$t = 1$	[BBC+19]
	IT (perfect)	2	admissible, e.g., {⋈}	$t = 1$	[BBC24]
	IT (perfect)	2	friendship, e.g., {⛢}	$t < n$	[BBC24]
LB	OT	3	{□, ⊏}	$n = 4, t = 2$	[BBMM18]
	KA	3	{⊏}	$n = 4, t = 1$	[BBC+19]
	KA	3	{⬠}	$n = 7, t = 2$	[BBC+19]
	KA	2	{⊤◸}	$n = 4, t = 1$	[BBC+20]
	KA	2	{◸, ⊓}	$n = 3, t = 1$	[BBC+20]
	KA	2	{⋈, ×}	$n = 5, t = 1$	[BBC24]
	KA	**2**	{⋈}	$\boldsymbol{n = 6, t = 2}$	**Thm 1.1**

Table 2: Assumption needed for THB. For upper bounds (UB) the listed assumptions are sufficient whereas for lower bounds (LB) the listed assumptions are necessary.

channels such as [BGW88, CCD88].

Second, as implicitly used in several IT-THB protocols, we reduce the task of THB to the task of *topology-hiding reliable message transmission* (THRMT). Here, there is a known sender P_S and a known receiver P_R, and the goal is to deliver a message from the sender to the receiver in a topology-hiding manner. Given a THRMT protocol, one can obtain a THB protocol via $n-1$ executions of the THRMT protocol with the same sender sending the same message to every potential receiver.

In this work we focus on constructing THRMT protocols from KA over diameter-2 graphs. (Recall that certain diameter-3 graphs require OT for this task.) In such graphs it is guaranteed that every pair of parties are either neighbors or have a common neighbor; however, a given party (or a subset of the parties) may not be aware whether other parties are neighbors or not.[3] We proceed to explain the ideas underlying our protocols in a gradual way.

Star-embedded graphs with degrees 1 and 2. We begin with a simple graph class, consisting of a subgraph of the n-nodes wheel graph, i.e., star-embedded graphs with a single center and where each non-center node has degree 1 or 2; see Figure 8 for an example (also Figure 5(a)). Here, in case there is a tail (i.e., a node of degree 1), KA is known to be necessary for THB even for a single corruption [BBC24] and sufficient for THC facing a single corruption [BBC+20] (using the

[3] In this work, for simplicity, we focus on connected graphs. THB over non-connected graphs can be achieved using similar techniques as in [BBC24].

"dead-end channels" technique, which is inherently tied to $t = 1$); assuming OT, t-secure THC is known for any $t < n$ [MOR15, BBKM23].

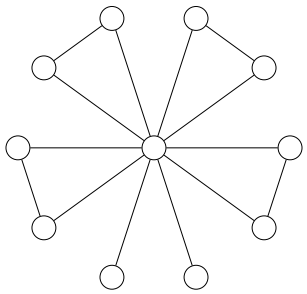


Figure 8: A star-embedded graph where non-center nodes have degree one or two.

We establish THB for any $t < n$ and THC for any $t < n/2$ by constructing t-secure THRMT for $t < n$. Consider first the setting where the sender P_S and receiver P_R are non-neighbors, in which the structure of the graph implies that neither of them is the center node. Our protocol begins by P_S sending its message $m \in \{0, 1\}$ to all of its neighbors. Now, it is guaranteed that exactly one of P_R's neighbors knows m, but if the receiver has two neighbors it must not know which one it is.

We accomplish this by utilizing the KA protocol. Say that P_R has two neighbors, then these neighbors start by executing the KA protocol to establish a secure communication channel between themselves. Although they are neighbors (since one of them is the center), the center must not know who the second neighbor is. Hence, they communicate and execute the KA protocol via the receiver. Further, in case the receiver has only one neighbor (which is the center), the information of whether the receiver has 1 or 2 neighbors cannot be revealed to the center; therefore, in this case P_R will emulate in its head a virtual second neighbor.

Given this secure channel, P_R and its two neighbors (one of them might be virtual) participate together in an MPC protocol in which the center enters the message m, and the receiver and the non-center neighbor enter 0. The function sums up all inputs and gives output only to P_R. As this is a linear function, we can use either the semi-honest protocol from [BGW88] or from [GMW87] which are secure against any number of corruptions. (Note that in the case P_R emulates an additional virtual party in the MPC it already knows its unique neighbor is the center, so no hiding is required for P_R from the MPC.)

The case where P_S and P_R are neighbors, may seem even simpler, as they can communicate directly and transmit the message; however, care must be taken not to reveal this information to the rest of the system. Focusing on the case where the receiver is the center, it must hide this from any degree-2 neighbor it may have. We resolve this by establishing a secure channel between every pair of parties using the KA protocol via the receiver (who is the center) and having all parties participate in the solitary-output MPC protocol for the sum, in which parties that are not neighbors of the sender enter 0.

To ensure that receiver's neighbors cannot distinguish between the case where the receiver is the center or not, we adjust the case where the receiver is *not* the center. Instead of establishing a single secure channel, the receiver will emulate in its head every non-neighboring party and establish n^2 secure channels, one between every pair of real receiver neighbors and virtual non-neighbors. The receiver will set the input of each emulated party to be 0, so the result of the computation does not change.

However, this change introduces a new challenge: In case the receiver is the center and the sender has degree 2, then there is one common neighbor between the sender and the receiver. The sender, in turn, does not know which of its two neighbors is the center. If both the sender and the common neighbor enter the bit to the computation of the sum, the result may not be correct anymore, and the receiver will learn that the degree of the sender is 2. To overcome this challenge, we make use of the observation that once the sender and receiver are neighbors, they do not need the MPC computation for transmitting the bit, but just to hide the fact that they are neighbors. Therefore, they can jointly ensure that the MPC computation will abort without anyone else realizing it. In particular, using [BGW88] or [GMW87], this can be done by using n-out-of-n secret sharing, and having the sender not sending the output share to the receiver. This way, the receiver obtains no information about the output of the computation, and can directly output the bit received beforehand from the sender. We abstract this into the function computed by the MPC by allowing the sender to provide either a bit or abort: if all inputs are bits the output is their sum, and otherwise the output is $\perp$.

Star-embedded graphs with degrees up to 3. In the previous case, the graph class ensured that every pair of non-neighboring sender and receiver have exactly one common neighbor which is the center, and this common neighbor is the only party to enter the sender's message into the MPC. We can generalize this approach to subgraphs of n-nodes wheel that are star-embedded and where non-center nodes may have degree 1, 2, or 3. Here, pair of non-neighboring sender and receiver may have one common neighbor or two (where the common neighbor who is not the center has degree 3); see Figure 9 (also Figure 6).

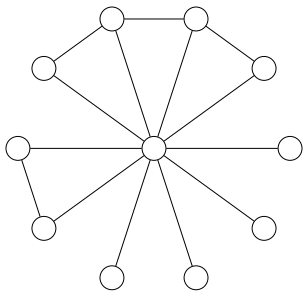


Figure 9: A star-embedded graph where non-center nodes have degree up to 3.

The main observation here is that although there may be two common neighbors between P_S and P_R, each one of them can self-validate whether it is the center or not. To ensure that exactly one of them provides the input to the computation, we have the non-center party set its input to the MPC of the summation to be 0. Looking ahead, we abstract this idea to more involved graph classes by defining a predicate Pred (having a high degree in the current example) that enables a common neighbor to determine whether it should provide the sender's message or 0 as its input to the MPC.

Known number of common neighbors. We next describe how to extend this approach from star-embedded graphs that have a unique party that is a neighbor of everyone else, to more involved graphs in which every pair of non-neighboring parties has a known number d of common neighbors: these common neighbors may be the same for all pairs (like multiple centers of stars, see Figure 5(b))

or different for each pair, as illustrated in Figure 10. As in the previous case, we can relax this requirement and enable more common neighbors, as long as exactly d common neighbors satisfy some predicate.

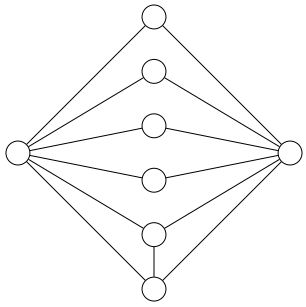


Figure 10: In this 8-node graph class the number of common neighbors is unknown as some non-neighboring parties have two common neighbors and some have six. Nonetheless, we can specify the predicate to return 1 if the party has degree at least 3. This way, each pair of non-neighboring parties has exactly two common neighbors that self-validate, and so we can establish THB for $t < 8$, and THC for $t < 4$, assuming KA.

In this case, when there are d common neighbors, we need to adjust the function to be computed by the MPC. Since there is more than one party that may enter the sender's message, we no longer compute the sum, but instead we compute the Boolean OR of the input bits. As we must still allow the sender to abort the computation, we call this function $f_{\mathsf{or\text{-}abort}}$. We compute this function facing $t < n$ corruptions by providing a linear circuit. Indeed, if it is guaranteed that exactly d common neighbor will enter sender's bit m, then the sum of the inputs will compute $\sum x_i = d \cdot m$. We can recover m back by dividing by d. Therefore, we use the BGW protocol [BGW88] over a field $\mathbb{F}_q$ with a prime $q > n$; this ensures that $d \in \mathbb{F}_q$ is invertible, and we can compute the circuit $\frac{1}{d}\sum x_i$. Further, since d is known, multiplying by the inverse of d in $\mathbb{F}_q$ is a multiplication by a scalar, i.e., a linear operation. This proves the first condition in Theorems 1.2 and 1.3.

Unknown but odd. In case the number of common neighbors (or number of self-validated common neighbors) is unknown, we can no longer multiply by the inverse of this value as a linear operation. Nonetheless, in case this unknown number of such common neighbors is guaranteed to be *odd*, we can adjust the protocol. See Figure 5(c) for an illustration. The idea is to work over a field of characteristic 2, such that summing up all the input bits would result with the original input. That is, we work over the field $\mathbb{F}_q$ with $q = 2^\ell$ and $q > n$, and compute the circuit $\sum x_i$. This proves the second condition in Theorems 1.2 and 1.3.

Bounded degree. The techniques discussed so far covered graph classes and established optimal resiliency, of $t < n$ for THB and $t < n/2$ for THC, by relying on BGW for linear circuits. We next focus on cases where the number of (self-validated) common neighbors is not known and potentially even, as illustrated in Figure 7 and captured in Theorem 1.4. Recall that when the number of common neighbors d is fixed and known, the MPC function is $\frac{1}{d}\sum x_i$. In case this number is not known and may be even, we adjust the circuit to first compute the parameter d. This is done by having each party enter two bits (x_i, α_i) as input: either $(m, 1)$ where m is the

sender's message, or $(0,0)$. The circuit first sums up the second coordinate of the inputs to obtain the number of input providers d and the first coordinate to obtain $d \cdot m$. Next, the circuit inverts d and multiplies

$$\left(\sum \alpha_i\right)^{-1} \cdot \left(\sum x_i\right) = \frac{1}{d} \cdot (d \cdot m) = m.$$

As this is a non-linear circuit and we do not assume OT, the protocol must rely on an honest majority. In particular, the sender cannot abort the computation in case it is a neighbor of the receiver, as the receiver can overcome this missing share and obtain the output. Therefore, we must have the sender participate in the MPC when the receiver is a neighbor, and the receiver always obtains the output m. This means that the requirement we make on the graph must hold also when the sender and receiver are neighbors.

We require the degree of each node to be bounded: not too small and not too large. Observe that by denoting d_{max} the maximal degree and by d_{min} the minimal degree, the number of parties executing the MPC is $d_{\mathsf{max}} + 1$, and the receiver P_R emulates $d_{\mathsf{max}} - \deg(\mathsf{P}_R)$ parties (and also participates). This party emulation takes place to hide the true degree of P_R:

- In the case that P_R is honest, then the number of possible corrupted neighbors t must be a minority of the $d_{\mathsf{max}} + 1$ participating parties. That is, $t < \frac{1}{2} \cdot (d_{\mathsf{max}} + 1)$.
- In the case that P_R is corrupted, then it could emulate as many as $(d_{\mathsf{max}} - d_{\mathsf{min}})$ of these parties, as well as collude with $(t-1)$ additional corrupted neighbors; thus, we must have this total number of effective corruptions to be a minority of the $d_{\mathsf{max}} + 1$ participating parties. This amounts to the constraint $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$.

Further, note that the condition $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$ together with $d_{\mathsf{min}} \leq d_{\mathsf{max}} < n$ implies $t < n/2$. Therefore, THC also exists under the KA assumption.

1.2.2 Impossibility Results

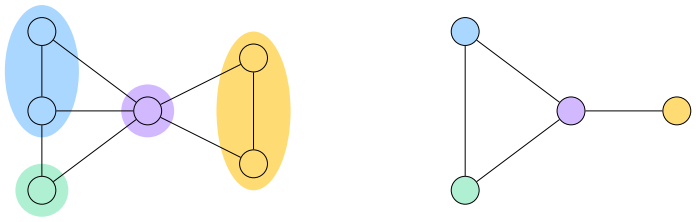
We proceed to explain the lower bound, stated in Theorem 1.1. We focus on the 6-node case where $t = 2$, as illustrated in Figure 4(b). This class admits 1-secure $\mathsf{IT\text{-}THB}$ [BBC24], yet we show that 2-secure THB necessitates KA. By Theorem 1.3, there exists a t-secure THB for any $t < n$ assuming KA; combined, this shows that KA is sufficient and necessary for this class.

The graph class $\mathcal{G}_{\mathsf{join\text{-}stars}}(6)$ consists of a center node connected to two components (their "star" structure is degenerate in this case): one consists of two nodes (each of degree 2) and the other a path of three nodes (one of degree 3 and two of degree 2), see Figure 4(b).

Subtleties with the player simulation technique. Given a 2-secure THB protocol π for this class, a natural approach to prove the necessity of KA is via player simulation, by a constructing a 1-secure THB protocol π' for to the 4-node "paw" graph (where KA is known to be necessary [BBC+20]), see Figure 4(a). In such a reduction: (1) the "tail" node in the paw would simulate the smaller component in $\mathcal{G}_{\mathsf{join\text{-}stars}}(6)$ consisting of the two neighbors with degree-2; (2) one of nodes of degree 2 in the paw would simulate two neighbors in the larger component; (3) the second node of degree 2 in the paw would simulate the remaining party in the larger component; and (4) the center in the paw would simulate the center in $\mathcal{G}_{\mathsf{join\text{-}stars}}(6)$. See Figure 11 for an illustration.

However, player simulation in the topology-hiding setting is more subtle than in standard MPC. In topology-revealing MPC, we can have a party in π' simply run the code of multiple parties in π.

In the topology-hiding case, we must ensure this does not inadvertently leak the topology to the center node. Since the center in $\mathcal{G}_{\text{join-stars}}(6)$ has five neighbors, it expects five communication channels. In a naïve reduction to the 4-node paw, the two simulated neighbors would have their communication "multiplexed" over the single channel between the tail and the center. This reveals to the center that two virtual nodes are connected, thus leaking information about the simulated topology.



(a) (b)

Figure 11: The problem with the player-simulation technique. On the left is the 6-node graph, where a 2-secure THB would presumably imply a 1-secure THB for the paw graph on the right. However, although in the 6-party graph the sender does not know which parties are neighbors, this information is leaked during the reduction to the paw.

The phantom jump technique. To overcome this obstacle, we "open the box" and extend the lower bound of the paw [BBC+20], which was based on the "phantom jump" technique. The high-level idea is given a 2-secure THB protocol π for $\mathcal{G}_{\text{join-stars}}(6)$ to construct a KA protocol that is run between two parties, Alice and Bob. Alice begins by choosing two long random strings m_1 and m_2 and sending them to Bob in the clear. Next, Alice and Bob continue in phases as follows:

- In each phase Alice and Bob locally toss coins c_A and c_B, respectively.
- They proceed to run two executions of π, as illustrated in Figure 12, in which Alice always emulates both ① and ② and Bob always emulates ③.
  - If $c_A = 0$ then in the first execution Alice emulates ④, ⑤, and ⑥ as neighbors of ①, who acts as the center, where ④ acts as the broadcaster with message m_1.
    Otherwise, if $c_A = 1$, then in the second execution Alice emulates ④, ⑤, and ⑥ as neighbors of ①, who acts as the center, where ④ acts as the broadcaster with message m_2.
  - Similarly, if $c_B = 1$ then in the first execution Bob emulates ④, ⑤, and ⑥ as neighbors of ③, who acts as the center, where ④ acts as the broadcaster with message m_1.
    Otherwise, if $c_B = 0$, then in the second execution Bob emulates ④, ⑤, and ⑥ as neighbors of ③, who acts as the center, where ④ acts as the broadcaster with message m_2.
- If parties ①, ② and ③ output m_1 in the first run and m_2 in the second, Alice and Bob output their bits c_A and c_B, respectively; otherwise, they execute another phase.

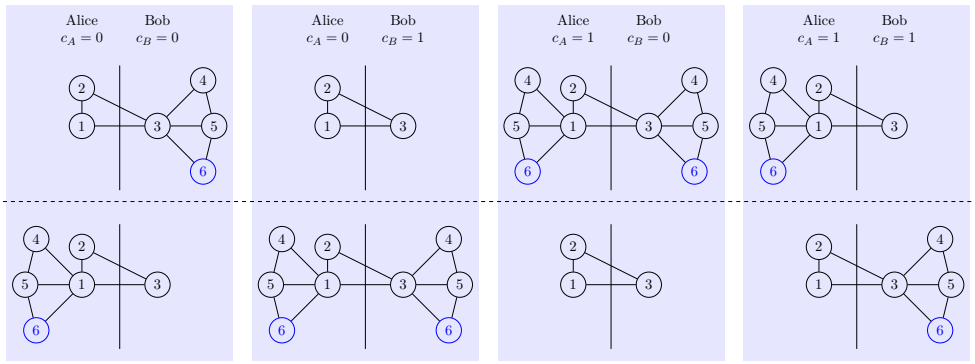


Figure 12: The four possible pairs of executions in a phase of the KA protocol constructed from a THB protocol for $\mathcal{G}_{\text{join-stars}}(6)$. Above the dashed line is the first execution and below the dashed line is the second. If $c_A \neq c_B$ there is no sender in one of the executions so Alice and Bob will execute another phase; if $c_A = c_B$ an eavesdropper cannot identify whether Alice or Bob simulate the broadcaster in each execution.

In case $c_A = c_B$ in some phase, then Alice and Bob will output the same coin, and by the assumed security of π, the eavesdropper Eve will not be able to learn who emulated ④, ⑤, and ⑥ in the first execution and who in the second, i.e., whether $c_A = c_B = 0$ or $c_A = c_B = 1$. In case $c_A \neq c_B$, then in at least one of the executions nobody emulates the broadcaster ④, so with overwhelming probability Alice and Bob will detect this case and execute another iteration.

In more detail, when $c_A = c_B$ the view of Eve consists of the two communication channels between ① and ③ and between ② and ③, as depicted in Figure 13. The leftmost case corresponds to the first execution when $c_A = c_B = 1$ and the second execution when $c_A = c_B = 0$, and the rightmost case to the first execution when $c_A = c_B = 0$ and the second execution when $c_A = c_B = 1$. We prove indistinguishability using four hybrid steps.

1. By the security of the THB protocol, when ③ acts as the center it cannot distinguish which nodes are connected to which; in particular, the distribution of the messages on the channels between ① and ③ and ② and ③ is indistinguishable when ① and ② are neighbors and when they are not (i.e., they have another common neighbor, say ⑥). This justifies the first hybrid.

2. Again, by the security of the THB protocol, when neither ① and ② is the center, and they have two common neighbors (③ and ⑥), they cannot know which of their common neighbors is the center, so they cannot distinguish between the center being ③ or ⑥. In particular, the distribution of the messages on the channels between ① and ③ and between ② and ③ is indistinguishable in both cases. This justifies the second hybrid.

3. Similarly, when ③ is not the center, it cannot distinguish between the center being ① or ⑥. In particular, the distribution of the messages on the channels between ① and ③ and between ② and ③ is indistinguishable in both cases. This justifies the third hybrid.

4. Finally, when ① acts as the center, then together with ②, they cannot distinguish whether ③ is connected to ⑥ or not; in particular, the distribution of the messages on the channel between ① and ② and between ② and ③ is indistinguishable in both cases. This justifies the fourth hybrid.

We conclude that the messages between nodes ① and ② and node ③ are indistinguishable when communicating in the topology of the leftmost case and the rightmost case; thus, the distinguishing advantage of Eve is negligible.

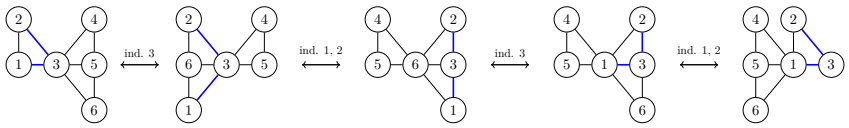


Figure 13: Hybrid steps in the phantom jump for $\mathcal{G}_{\mathsf{join\text{-}stars}}(6)$.

2 Preliminaries

Notations. For $n \in \mathbb{N}$ let $[n] = \{1, \ldots, n\}$. Looking ahead, we denote by n the exact number of participating parties and by t an upper bound on the number of corrupted parties. The security parameter is denoted by κ. We denote variables in italic single-letter, e.g., x, and Turing machines in sans-serif, e.g., P. We denote by $\mathbb{F}_q$ the finite field with q elements.

Let poly denote the set of all positive polynomials. A function $\mu\colon \mathbb{N} \to [0,1]$ is *negligible*, denoted $\mu(\kappa) = \mathsf{negl}(\kappa)$, if $\mu(\kappa) < 1/p(\kappa)$ for every $p \in \mathsf{poly}$ and large enough κ. Two distribution ensembles $X = \{X(a,\kappa)\}_{a\in\{0,1\}^*,\kappa\in\mathbb{N}}$ and $Y = \{Y(a,\kappa)\}_{a\in\{0,1\}^*,\kappa\in\mathbb{N}}$ are computationally indistinguishable (denoted $X \approx_c Y$) if for every non-uniform polynomial-time distinguisher Dist there exists a function $\mu(\kappa) = \mathsf{negl}(\kappa)$, such that for every $a \in \{0,1\}^*$ and every κ,

$$|\Pr[\mathsf{Dist}(X(a,\kappa), 1^\kappa) = 1] - \Pr[\mathsf{Dist}(Y(a,\kappa), 1^\kappa) = 1]| \leq \mu(\kappa).$$

Graph notations and properties. A graph $G = (V, E)$ is a set V of vertices and a set E of edges, each of which is an unordered pair of distinct vertices. The *(open) neighborhood* of a vertex v in an undirected graph G, denoted $\mathcal{N}_G(v)$, is the set of vertices sharing an edge with v in G. The *closed neighborhood* of v in G is in turn defined by $\mathcal{N}_G[v] := \mathcal{N}_G(v) \cup \{v\}$.

Isomorphically closed graph class. Following [BBC+20], in this work we will consider *isomorphically closed* graph classes.

Definition 2.1 (isomorphically closed)**.** *Let $n \in \mathbb{N}$ and let $\mathcal{G}$ be a graph class comprised of graphs with n vertices and labeled by $\{1, \ldots, n\}$. The graph class $\mathcal{G}$ is* isomorphically closed *if for every graph $G = (V, E) \in \mathcal{G}$ and for every bijection $\tau : V \to \{1, \ldots, n\}$, the graph $H = (\tau(V), \tau(E))$ is also in $\mathcal{G}$, where $\tau(E)$ is defined by $\tau(E) = \{(\tau(u), \tau(v)) \mid (u, v) \in E\}$.*

In other words, if a graph is in an isomorphically closed class, then so are all the graphs obtained by relabeling its vertices using $\{1, \ldots, n\}$. Furthermore, given an unlabeled graph G with n nodes, define *the graph class associated with G* as the graph class containing the labeled graphs $H = (\tau(V), \tau(E))$ where $\tau : V \to \{1, \ldots, n\}$ is a bijection and $\tau(E)$ is defined as above. By definition, the graph class associated with G is isomorphically closed.

2.1 Key-Agreement Protocols

We follow the definition from [KL14]. Let $\ell = \mathsf{poly}(\kappa)$. Let π be a two-party protocol where Alice and Bob begin with the security parameter 1^κ as input, and where Alice outputs $k_A \in \{0,1\}^\ell$

and Bob outputs $k_B \in \{0,1\}^\ell$. The protocol π is a key-agreement protocol for keys of length ℓ if there exists a negligible function μ such that $\Pr[k_A \neq k_B] \leq \mu(\kappa)$, where the probability is over the randomness of the parties. The security of π is defined as follows:

Definition 2.2 (security of key agreement)**.** *A key-agreement protocol π for keys of length ℓ is secure in the presence of an eavesdropper if for every PPT adversary* Adv *there is a negligible function μ such that*

$$\Pr\left[\mathsf{ExpKA}^{\mathsf{eav}}_{\mathsf{Adv},\pi}(\kappa) = 1\right] \leq \frac{1}{2} + \mu(\kappa)$$

where $\mathsf{ExpKA}^{\mathsf{eav}}_{\mathsf{Adv},\pi}(\kappa)$ is as defined in Figure 14 and the probability is taken over the random coins of the experiment and of the adversary.

Experiment $\mathsf{ExpKA}^{\mathsf{eav}}_{\mathsf{Adv},\pi}(\kappa)$

1. The challenger executes the protocol π between Alice and Bob on input 1^κ. Denote by trans the transcript of this execution and by k the common key (in case $k_A \neq k_B$, abort).
2. The challenger flips a random bit $b \leftarrow \{0,1\}$, if $b = 0$ set $k' := k$, and if $b = 1$ sample $k' \leftarrow \{0,1\}^\ell$ uniformly at random.
3. The challenger gives trans and k' to Adv, that returns a bit b'.
4. The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise.

Figure 14: The KA experiment

2.2 Topology-Hiding Computation (THC)

Following [MOR15] we consider two definitions of topology-hiding computation. Our positive results (protocol constructions) are defined with respect to the stronger simulation-based definition while our lower bounds are given with respect to the weaker indistinguishability-based definition.

UC framework. The simulation-based definition is defined in the UC framework of [Can01]; we present an informal overview of the model in Section A. Unless stated otherwise, we will consider static, semi-honest, PPT adversaries and environments.

2.2.1 Simulation-Based THC

We recall the definition of simulation-based topology-hiding computation from [MOR15, BBC+19]. The real-world protocol is defined in a model where all communication is transmitted via the functionality $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ (described in Figure 15). The functionality is parametrized by a family of graphs $\mathcal{G}$, representing all possible network topologies (aka communication graphs) that the protocol supports. We implicitly assume that every node in a graph is associated with a specific *party identifier*, pid. To simplify the notation, we will consider that P_v in the protocol is associated with node communication v in the graph.

Initially, before the protocol begins, $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ receives the network communication graph G from a special graph party $\mathsf{P}_{\mathsf{graph}}$, makes sure that $G \in \mathcal{G}$, and provides to each party P_v with $v \in V$ its local neighbor-set. Next, during the protocol's execution, whenever party P_v wishes to send a

message m to party P_w, it sends (v, w, m) to the functionality; the functionality verifies that the edge (v, w) is indeed in the graph, and if so delivers (v, w, m) to P_w.

Given a class of graphs $\mathcal{G}$ with n parties, we define a protocol π with respect to $\mathcal{G}$ as a set of n PPT interactive Turing machines (ITMs) $(\mathsf{P}_1, \ldots, \mathsf{P}_n)$ (the parties). Each party is initially activated with a (potentially empty) input. All the parties participate in the protocol, send messages to one another (via $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$), and produce output.

Functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$

The functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ is parametrized by a family of n-node graphs $\mathcal{G}$. The functionality proceeds with a special graph party $\mathsf{P}_{\mathsf{graph}}$ and with the parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ as follows.

Initialization Phase:

Input: $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ waits to receive the graph $G = (V, E)$ from $\mathsf{P}_{\mathsf{graph}}$. If $G \notin \mathcal{G}$, abort.

Output: Upon receiving an initialization message from P_v, verify that $v \in V$, and if so send $\mathcal{N}_G(v)$ to P_v.

Communication Phase:

Input: $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ receives from a party P_v a destination/data pair (w, m) where $w \in \mathcal{N}_G(v)$ and m is the message P_v wants to send to P_w. (If $v, w \notin V$, or if w is not a neighbor of v, $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ ignores this input.)

Output: $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ gives output (v, m) to P_w indicating that P_v sent the message m to P_w.

Figure 15: The communication graph functionality

An ideal-model computation of a functionality $\mathcal{F}$ is augmented to provide the corrupted parties with the information that is leaked about the graph; namely, every corrupted (dummy) party should learn its neighbor-set. Note that the functionality $\mathcal{F}$ can be completely agnostic about the actual graph that is used, and even about the family $\mathcal{G}$. To augment $\mathcal{F}$ in a generic way, we define the wrapper-functionality $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F})$, that runs internally a copy of the functionality $\mathcal{F}$. The wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\cdot)$ acts as a shell that is responsible to provide the relevant leakage to the corrupted parties; the original functionality $\mathcal{F}$ is the core that is responsible for the actual ideal computation.

More specifically, the wrapper receives the graph $G = (V, E)$ from the graph party $\mathsf{P}_{\mathsf{graph}}$, makes sure that $G \in \mathcal{G}$, and sends a special initialization message containing G to $\mathcal{F}$. (If the functionality $\mathcal{F}$ does not depend on the communication graph, it can ignore this message.) The wrapper then proceeds to process messages as follows: Upon receiving an initialization message from a party P_v responds with its neighbor set $\mathcal{N}_G(v)$ (just like $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$). All other input messages from a party P_v are forwarded to $\mathcal{F}$ and every message from $\mathcal{F}$ to a party P_v is delivered to its recipient.

Note that formally, the set of all possible parties V^* is fixed in advance. To represent a graph $G' = (V', E')$ where $V' \subseteq V^*$ is a subset of the parties, we use the graph $G = (V^*, E')$, where all vertices $v \in V^* \setminus V'$ have degree 0.

Definition 2.3 (Topology-hiding computation)**.** *We say that a protocol π securely realizes a functionality $\mathcal{F}$ in a topology-hiding manner with respect to $\mathcal{G}$ tolerating a semi-honest PPT adversary*

corrupting t parties if π securely realizes $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F})$ in the $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$-hybrid model tolerating a semi-honest adversary corrupting t parties.

Below we describe a few important functionalities that we use in this work.

Topology-hiding SFE. In the *secure function evaluation* (SFE) functionality for a function f, denoted $\mathcal{F}^{f}_{\mathsf{sfe}}$ (see Figure 16), each party P_i provides an input x_i, the functionality computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, and sends y_i to P_i. We next present a *topology-hiding* variant (THSFE).

Definition 2.4 (THSFE)**.** *Let $n \in \mathbb{N}$, let $\mathcal{G}$ be a class of n-node graphs, let t be an integer, and let $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-party function. A protocol π is a t-secure THSFE protocol for f with respect to $\mathcal{G}$ if π securely realizes $\mathcal{F}^{f}_{\mathsf{sfe}}$ in a topology-hiding manner with respect to $\mathcal{G}$ tolerating a semi-honest adversary corrupting t parties.*

Functionality $\mathcal{F}^{f}_{\mathsf{sfe}}$

The SFE functionality $\mathcal{F}^{f}_{\mathsf{sfe}}$ is parameterized by an n-party function $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ and proceeds as follows.

Initialization: The functionality receives the communication graph G from the wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}$.

Input: Upon receiving the input $x_i \in \{0,1\}^*$ from P_i, record it and forward $|x_i|$ to the adversary.

Output: Compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ and send the output y_i to every party P_i.

Figure 16: The secure function evaluation functionality

Standard SFE. We also consider the standard notion of *secure function evaluation* (SFE) (i.e., not topology hiding), which is captured by the SFE functionality in Figure 16, without the initialization phase. In our case we will consider SFE protocols over a star network where only the center node obtains the output.

Definition 2.5 (SFE)**.** *Let $n \in \mathbb{N}$, let t be an integer, and let $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-party function. A protocol π is a t-secure SFE protocol for f if π securely realizes $\mathcal{F}^{f}_{\mathsf{sfe}}$, tolerating a semi-honest adversary corrupting t parties.*

Topology-hiding Broadcast. In the *broadcast* functionality (see Figure 17), where a designated and publicly known party, named *the broadcaster*, starts with an input value m. Our broadcast functionality guarantees that every party receives the message m as output. We denote the broadcast functionality where the broadcaster is P_v by $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$.

Definition 2.6 (THB)**.** *Let $n \in \mathbb{N}$, let $\mathcal{G}$ be a class of n-node graphs, and let t be an integer. A protocol π is a t-secure THB protocol with respect to $\mathcal{G}$ if $\pi(\mathsf{P}_v)$ securely realizes $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$ in a topology-hiding manner with respect to $\mathcal{G}$, for every P_v, tolerating a semi-honest adversary corrupting t parties.*

Functionality $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_i)$

The broadcast functionality $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_i)$ is parametrized by the broadcaster P_i and proceeds as follows.

Initialization: The functionality receives the communication graph G from the wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}$.

Input: Upon receiving the input message $m \in \{0,1\}$ from the sender P_i, record it and forward m to the adversary.

Output: Send the output m to every party that is in the same connected component as P_i in G. For every other party in G, the output delivered to that party is supplied by the adversary.

Figure 17: The broadcast functionality

Topology-hiding Reliable message transmission (THRMT). Our THB construction we be decomposed to a topology-hiding computation of the *RMT* functionality (see Figure 18), where a designated and publicly known sender P_S, transmits its input value m to a designated and publicly known receiver P_R. We denote the RMT functionality with sender P_S and receiver P_R by $\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R)$.

Definition 2.7 (THRMT)**.** *Let $n \in \mathbb{N}$, let $\mathcal{G}$ be a class of n-node graphs, and let t be an integer. A protocol π is a t-secure THRMT protocol with respect to $\mathcal{G}$ if $\pi(\mathsf{P}_S, \mathsf{P}_R)$ securely realizes $\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R)$ in a topology-hiding manner with respect to $\mathcal{G}$, for every P_S and P_R, tolerating a semi-honest adversary corrupting t parties.*

Functionality $\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R)$

The RMT functionality $\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R)$ is parametrized by the sender P_S, the receiver P_R and proceeds as follows.

Initialization: The functionality receives the communication graph G from the wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}$.

Input: Upon receiving the input message $m \in \{0,1\}$ from the sender P_S, record it and forward m to the adversary.

Output: Send the output m to party P_R.

Figure 18: The reliable message transmission functionality

2.2.2 Indistinguishability-Based THC

Moran et al. [MOR15] gave a weaker definition of topology-hiding computation: IND-CTA security (indistinguishability under Chosen Topology Attack). We will next provide the explicit definitions for broadcast protocols.

Definition 2.8 (IND-CTA THB)**.** *Let $n \in \mathbb{N}$, let $\mathcal{G}$ be a class of n-node graphs, and let t be an integer. A broadcast protocol π is* indistinguishable under chosen topology attack against t semi-honest corruptions (t-IND-CTA secure) *with respect to $\mathcal{G}$, if for any PPT adversary* Adv *there exists a negligible function* negl*, such that for every $\kappa \in \mathbb{N}$ it holds that*

$$\Pr\left[\mathsf{ExpTHB}^{\mathtt{ind\text{-}cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa) = 1\right] \leq 1/2 + \mathsf{negl}(\kappa),$$

where $\mathsf{ExpTHB}^{\texttt{ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa)$ *is as defined in Figure 19 and the probability is taken over the random coins of the experiment and of the adversary.*

Experiment $\mathsf{ExpTHB}^{\texttt{ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa)$

Choice phase. The challenger invokes Adv on input $(\mathcal{G}, 1^\kappa)$. Adv chooses two graphs $G_0, G_1 \in \mathcal{G}$, a broadcaster node u, a message $m \in \{0,1\}$, and a set $\mathcal{I} \subseteq [n]$ of size $|\mathcal{I}| \leq t$ such that $\mathcal{N}_{G_0}(v) = \mathcal{N}_{G_1}(v)$ for every $v \in \mathcal{I}$. Next, the adversary returns $(G_0, G_1, u, m, \mathcal{I})$. If Adv's output is not of the required form, the experiment aborts.

Challenge phase. The challenger flips a random bit $b \leftarrow \{0,1\}$ and runs $\pi(1^\kappa)$ with node u broadcasting message m over graph G_b, where the adversary controls every P_v with $v \in \mathcal{I}$.

Output phase. At the conclusion of the execution of π, Adv outputs b'. If $b = b'$, the experiment outputs 1; otherwise 0.

Figure 19: The 1-IND-CTA broadcast experiment

In our lower bounds (Section 5) we will require basic sequential composition guarantees from t-secure IND-CTA broadcast protocols. Namely, we will compose κ executions of 1-bit broadcast protocols over the same graph to transmit longer messages, and we will run two instances of κ-bit broadcast over two different graphs within the class $\mathcal{G}$. Definition 2.8 remain secure under sequential composition via a simple hybrid argument, as we consider *independent* executions and a *semi-honest* adversary.

Definition 2.9 (sequential composition)**.** *A broadcast protocol π is t-secure IND-CTA under q sequential composition with respect to a graph class $\mathcal{G}$, if for every PPT adversary* Adv *there exists a negligible function* negl*, such that for every $\kappa \in \mathbb{N}$ it holds that*

$$\Pr\left[\mathsf{ExpTHB}^{\texttt{ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(q,\kappa) = 1\right] \leq 1/2 + \mathsf{negl}(\kappa),$$

where $\mathsf{ExpTHB}^{\texttt{ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(q,\kappa)$ *is as defined in a similar way to* $\mathsf{ExpTHB}^{\texttt{ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa)$ *with the exception that the adversary chooses q tuples (G_0^i, G_1^i, u^i, m^i) and the challenger sequentially runs q executions of π where the ith execution is with graph G_b^i and sender u^i broadcasts message m^i.*

The following lemma is proven in [BBC+20].

Lemma 2.10. *Let $q \in \mathsf{poly}(\kappa)$. If π is a t-secure IND-CTA broadcast protocol with respect to $\mathcal{G}$ then π is t-secure under q sequential composition.*

3 Topology-Hiding RMT from Linear MPC

In this section, we present our THRMT protocols from KA for certain diameter-2 graphs, that is achieved from t-secure semi-honest MPC for $t < n$; such MPC is known from KA if the function can be represented by a linear circuit. We begin in Section 3.1 by constructing THRMT protocols in the SFE-hybrid model (for the function $f_{\mathsf{or\text{-}abort}}$ defined below). Next, in Sections 3.2 and 3.3 we present different instantiations of the SFE functionality for different graphs classes. Our constructions in this section are for diameter-two graphs; we begin by defining this graph class.

Definition 3.1 (diameter-two graph)**.** *A graph $G = (V, E)$ is a* diameter-two graph *if for every pair of non-adjacent vertices $u, v \in V$ (i.e., $(u, v) \notin E$), it holds that $\mathcal{N}(u) \cap \mathcal{N}(v) \neq \emptyset$.*

Denote by $\mathcal{G}_{\mathsf{diam\text{-}two}}(n)$ the graph class consisting of all diameter-two graphs with n vertices.

3.1 RMT Protocol in SFE-Hybrid Model

The main idea is to leverage the graph's structure by having the sender forward the message to all its neighbors. Since the graph has diameter two, at least one of the receiver's neighbors must have received the message. Therefore, the challenge is to securely forward the message from these intermediate nodes to the receiver.

Initially, we abstract the last step using a secure function evaluation of the solitary-output function $f_{\mathsf{or\text{-}abort}}(x_1, \ldots, x_n)$ where $x_i \in \{0, 1\} \cup \{\mathsf{abort}\}$ for every $i \in [n]$. In case one the inputs is a special input abort, the output is $\perp$; otherwise, the output is $x_1 \vee \ldots \vee x_n$. We emphasize that this is *not* topology-hiding SFE, since we operate in a star-based network, in which the center is the only party learning any output. Every other party only communicates with the receiver during this protocol.

Protocol $\pi_{\mathsf{rmt}}(n, \mathsf{P}_S, \mathsf{P}_R)$

Auxiliary input: An n parties diameter-two graph-class $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$ and a predicate Pred.

Input: The sender P_S, with $S \in [n]$, holds an input $m \in \{0, 1\}$.

Hybrid model: The protocol is defined in the $(\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}, \mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}})$-hybrid model.

The protocol:

1. *Initialization.*
   Each party P_i sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ and receives its neighbor-set $\mathcal{N}(i)$.
2. *Sender sends the message to its neighbors.*
   The sender P_S sends its input message m to each of its neighbors, P_i with $i \in \mathcal{N}(S)$.
3. *Receiver obtains the message from its neighbors.*
   (a) Every neighbor of the receiver, P_i with $i \in \mathcal{N}(R)$, invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ as follows:
      i. If P_i is the sender, with the message abort.
      ii. Otherwise, if $i \in \mathcal{N}(S)$ and $\mathsf{Pred}(i) = 1$ with the message m (received from P_S).
      iii. Otherwise, with 0.
   (b) The receiver invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ by emulating every P_i with $i \notin \mathcal{N}(R)$ sending 0 as input.
   (c) The receiver obtains an output $m^* \in \{0, 1\} \cup \{\perp\}$ from $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$.
4. *Output.* If P_R is a neighbor of P_S, then P_R outputs m received from P_S in Step 2; otherwise, P_R outputs m^* received from $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ in Step 3.

Figure 20: THRMT protocol for $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}$ in $(\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}, \mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}})$-hybrid model.

Lemma 3.2. *Let $n \in \mathbb{N}$, let $t < n$, let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$, and let Pred be a predicate. Suppose that for every $S, R \in [n]$ it holds that for every $G \in \mathcal{G}$ either $R \in \mathcal{N}_G(S)$ or there exists $v \in \mathcal{N}_G(S) \cap \mathcal{N}_G(R)$ for which $\mathsf{Pred}(v) = 1$.*

Then, π_{rmt} is a t-secure THRMT with respect to $\mathcal{G}$ in the $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$-hybrid model.

Proof. Since the graph has diameter two, correctness follows directly from the protocol structure, and the correctness of the underlying secure function evaluation. The receiver P_R obtains the correct message in one of the following cases:

- If the receiver is a neighbor of the sender, then it gets the message directly from the sender in Step 2, and outputs it.
- Otherwise, there is at least one common neighbor v between the sender and the receiver such that and $\mathsf{Pred}(v) = 1$. It holds that P_v receives the message m from the sender in Step 2 and invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ with m; further, the sender does not send abort (as it is not a neighbor of P_R) and all parties that are neighbors of the receiver but not neighbors of the sender invoke $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ with 0. By definition of $f_{\mathsf{or\text{-}abort}}$ the output in this case is the Boolean OR of all input bits, which equals the message m.

Hence, the receiver obtains the correct output in both cases.

For security, we will construct a simulator Sim that given the neighbor-set of the set of corrupted parties, indexed by $\mathcal{I} \subseteq [n]$, generates a simulated view for the corrupted parties that is computationally indistinguishable from their view in a real execution of the protocol. The simulator Sim runs every P_i with $i \in \mathcal{I}$ in its head, and does the following:

- To simulate the initialization phase, in Step 1 each party P_i with $i \in \mathcal{I}$ sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ and expects to receive its neighbor-set $\mathcal{N}(i)$. The simulator forwards these messages to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$ and sends the response $\mathcal{N}(i)$ to each P_i.

  In the input phase, in case the sender is corrupted, i.e., $S \in \mathcal{I}$, the simulator sends its input m to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$; regardless, Sim receives the message m as the output from $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$ (formally, Sim sends the "empty input" for a non-sender corrupted party).
- To simulate Step 2, if the sender is corrupted i.e., $S \in \mathcal{I}$, Sim receives m on behalf of every honest party P_i for which $i \in \mathcal{N}(S)$. If the sender is honest i.e., $S \notin \mathcal{I}$, then Sim sends m to every corrupted party P_i for which $S \in \mathcal{N}(i)$.
- To simulate Step 3, Sim simulates $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ towards the adversary as follows:
  1. If the receiver is corrupted, i.e., $R \in \mathcal{I}$:
     (a) Sim receives a value for each corrupted party P_v with $v \in \mathcal{I} \cap \mathcal{N}(R)$.
     (b) Sim receives a value from P_R for each party that P_R emulates in its head, and in addition a value from P_R itself.
     (c) If the sender is a neighbor of the receiver, i.e., $S \in \mathcal{N}(R)$, then Sim sets the output of the functionality to be $\perp$; otherwise, Sim sets the output to be the message m.
     (d) Eventually, Sim provides the output to P_R.
  2. If the receiver is honest, i.e., $R \notin \mathcal{I}$:

(a) Sim receives a value for each corrupted party P_i with $i \in \mathcal{I} \cap \mathcal{N}(v)$. (Recall that only the receiver obtains the output of $f_{\mathsf{or\text{-}abort}}$, so there is nothing else to simulate when the receiver is honest.)

It is immediate to verify that the simulation is perfect, as the only difference between the view of the adversary in the real world and in the simulation is the way $f_{\mathsf{or\text{-}abort}}$ is computed; however, both computations induce the same distribution. □

3.2 Instantiating for d-Validated Graphs

We proceed to realize $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ with t-security for $t < n$ over graphs in which every pair of non-neighboring parties have exactly d common neighbors which are validated by Pred.

Definition 3.3 (d-validated graph)**.** *Let $d \in \mathbb{N}$. Given a graph $G = (V, E)$ of size n and a predicate Pred, the pair (G, Pred) is a* d-validated graph *if for every pair of non-adjacent vertices $u, v \in V$ (i.e., $(u, v) \notin E$), it holds that*

$$|\{w \in \mathcal{N}(u) \cap \mathcal{N}(v) \mid \mathsf{Pred}(w) = 1\}| = d.$$

Definition 3.4 (d-validated graph-class)**.** *Let $n \in \mathbb{N}$ and $d < n$, let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$, and let Pred be a predicate. The pair $(\mathcal{G}, \mathsf{Pred})$ is a* d-validated graph-class *if for every $G \in \mathcal{G}$ it holds that (G, Pred) is a d-validated graph.*

Protocol $\pi_{\mathsf{or\text{-}abort\text{-}d}}$

The n-party protocol is executed between the receiver P_R and its neighbors in the graph; the receiver emulates in its head every non-neighbor party P_i with $i \notin \mathcal{N}_G[R]$ to form an n-party protocol.

Input: Every P_i with $i \in [n]$ has an input $x_i \in \{0,1\} \cup \{\mathsf{abort}\}$

Auxiliary input: A field $\mathbb{F}_q$ with a prime $q > n$, a linear circuit C specified as $\frac{1}{d}\sum x_i$, and a key-agreement protocol π_{ka} for keys of length $\ell = \lceil \log q \rceil$.

The protocol:

1. *Establishing secure channels between the neighbors of the receiver.*
   (a) The receiver samples uniformly at random a permutation $\sigma : [n] \to [n]$.
   (b) The receiver emulates in its head every non-neighboring party P_i with $i \notin \mathcal{N}(R)$.
   (c) The receiver sends to each of its neighbors P_i with $i \in \mathcal{N}(R)$ and to the emulated parties P_i with $i \notin \mathcal{N}(R)$, the value $\sigma(i)$ (where i is the party's id).
   (d) Every ordered pair of parties P_i and P_j (where each is either a neighbor of P_R or emulated by P_R) execute the key-agreement protocol π_{ka} where the receiver is acting as a relay between them. Denote by $k_{i \to j}$ the resulting key.
   (e) At the end of this phase there are n^2 secure channels between all n parties: real neighbors of the receiver and non-neighbors of the receiver that are emulated by the receiver.
2. *Execute BGW for C.*
   The n parties execute the BGW protocol for the linear circuit C over $\mathbb{F}_q$.
   (a) Every party P_i uses its input x_i; if a party's input is abort, it proceeds with input 0.

(b) Each party P_i secret shares its input x_i using an n-out-of-n Shamir secret sharing over $\mathbb{F}_q$ as $(s_{i\to 1}, \ldots, s_{i\to n})$. Next, for every $j \in [n]$, party P_i encrypts the jth share as $c_{i\to j} = s_{i\to j} \oplus k_{i\to j}$ and sends $c_{i\to j}$ to P_j via the receiver.

(c) Upon receiving n encrypted shares $(c_{1\to i}, \ldots, c_{n\to i})$, party P_i decrypts each ciphertext as $s_{j\to i} = c_{j\to i} \oplus k_{j\to i}$ to obtain $(s_{1\to i}, \ldots, s_{n\to i})$. Next, P_i evaluates the circuit C over these shares to compute the output share τ_i.

(d) Every P_i sends the resulting share τ_i to the receiver; if the party's input is abort, it does not send this message.

(e) If the receiver receives n shares, it reconstructs and outputs the bit m^*; otherwise, it outputs $\perp$.

Figure 21: SFE protocol for $f_{\mathsf{or\text{-}abort}}$ over d-validated graphs

Lemma 3.5. *Let $n \in \mathbb{N}$ be polynomial in κ, let $d, t < n$, let $q > n$ be a prime, and let π_{ka} be a key-agreement protocol for keys of length $\ell = \lceil \log q \rceil$ secure in the presence of an eavesdropper. Let $(\mathcal{G}, \mathsf{Pred})$ be a d-validated graph-class. Then, for every $S, R \in [n]$, it holds that $\pi_{\mathsf{or\text{-}abort\text{-}d}}(q, \pi_{\mathsf{ka}})$ (defined in Figure 21) realizes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ in the presence of PPT semi-honest t-adversaries.*

Proof. We will prove *correctness* and *security* separately.

Correctness. We need to show that the SFE protocol $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ correctly realizes the $f_{\mathsf{or\text{-}abort}}$ functionality. If for any $i \in [n]$ the input is $x_i = \mathsf{abort}$ then in Step 2a party P_i uses 0 as input. In Step 2d party P_i will not send the final share, thus the reconstruction fails and the output is $\perp$. Otherwise, all parties have inputs in $\{0, 1\}$. Since the protocol is executed on a d-validated graph-class, there are exactly d parties P_i that $\mathsf{Pred}(i) = 1$ for those parties $x_i = m$, all other parties have $x_i = 0$. By the correctness of π_{ka}, each ordered pair of parties agree on a key except for negligible probability, in which case the computation of the BGW protocol for the linear circuit C is correct. Therefore, except for negligible probability, the output of the receiver is:

$$\frac{1}{d} \sum_{i=1}^{n} x_i = \frac{1}{d} \cdot d \cdot m = m.$$

This equals the output of $f_{\mathsf{or\text{-}abort}}$ which is m.

Security. In high level, security follows from the security of the BGW protocol for $t < n$ over secure channels together with the security of the KA protocol. Let Adv be a semi-honest adversary and denote by $\mathcal{I}$ the set of indices of corrupted parties. Denote by $\mathsf{Sim}_{\mathsf{BGW}}$ the simulator corresponding to Adv for the BGW protocol for linear functions, when executed over a complete graph of secure channels [BGW88, AL17, ACS22]. We will construct a simulator Sim by considering two cases:

We begin with assuming that for every pair of honest parties P_i and P_j, the key $k_{i\to j}$ used by P_i to encrypt the message to P_j is sampled uniformly at random from $\{0, 1\}^\ell$. In this case, P_i and P_j send one-time pad encryptions via the receiver, so they are effectively connected via a perfectly secure channel, and we can use the simulator for the BGW protocol for linear functions [BGW88, AL17, ACS22]. We will construct a simulator Sim by considering two cases:

Case 1: honest receiver. Sim starts by honestly simulating the receiver choosing a random permutation and sending $\sigma(i)$ to every $i \in \mathcal{N}(R) \cap \mathcal{I}$. Next, Sim simulates the KA instances in

Step 1 between every honest party (and simulated party by P_R) and every corrupted party, where all simulated communication is between the receiver and each corrupted party. In addition, the simulated receiver honestly routes corrupted-to-corrupted messages for the KA instances.

Next, in Step 2, the simulator encrypts the simulated BGW messages generated by $\mathsf{Sim}_{\mathsf{BGW}}$ towards the corrupted parties using the corresponding keys established by the KA executions. In addition, the simulated receiver honestly routes corrupted-to-corrupted encrypted messages for the BGW protocol.

Claim 3.6. *In case the receiver is honest, for every PPT environment, the output of the real computation of $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ with the semi-honest adversary Adv is computationally indistinguishable from the ideal computation of $f_{\mathsf{or\text{-}abort}}$ with Sim.*

Proof. Conditioned on the correctness of the KA instances (i.e., when every pair of parties generates a common key), which occurs with overwhelming probability, the claim follows by the perfect security of BGW and since every corrupted party only sees encryptions of messages sent to corrupted parties or received by corrupted parties. Further, as the receiver is honest, no corrupted party obtains any output. □

Case 2: corrupted receiver. Again, Sim starts by executing the KA in Step 1 instances between every honest party and every corrupted party, where now all simulated communication is between the receiver and each honest party.

Next, in Step 2 and as before, the simulator encrypts the simulated messages towards the corrupted parties using the corresponding keys established by the KA executions. However, differently from the previous case, for every pair of honest parties P_i and P_j, the simulator Sim samples a random key $k_{i\to j} \in \{0,1\}^\ell$ and simulates the messages from P_i to P_j via the corrupted receiver by encrypting zeros, i.e., by sending $k_{i\to j}$.

Claim 3.7. *In case the receiver is corrupt, for every PPT environment, the output of the real computation of $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ with the semi-honest adversary Adv is computationally indistinguishable from the ideal computation of $f_{\mathsf{or\text{-}abort}}$ with Sim.*

Proof. The proof follows via a standard hybrid argument. For $h \in \{0,\ldots,n^2\}$, define the hybrid game HYB^h by modifying the real computation of $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ as follows. In Step 1, after executing the KA instances, for every $i,j \in [n]$, if P_i and P_j are honest and if $(i-1)\cdot n + j \le h$, then the key $k_{i\to j}$ used by P_i to encrypt the message to P_j, is sampled uniformly at random from $\{0,1\}^\ell$ (instead of using the key computed by π_{ka}). Clearly, HYB^0 corresponds to the real execution of $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ with Adv, whereas the HYB^{n^2} corresponds to ideal computation of $f_{\mathsf{or\text{-}abort}}$ with Sim.

Now, assume toward contradiction that $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ does not securely realizes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$, i.e., there is a PPT environment Env that can distinguish with non-negligible probability. We will construct an attacker for π_{ka}.

Since HYB^0 and HYB^{n^2} can be distinguished with non-negligible probability, there must exist $h^* \in \{1,\ldots,n^2\}$ such that HYB^{h^*-1} and HYB^{h^*} can be distinguished with non-negligible probability. The attacker receives a transcript trans and a key k and proceeds to guess h^* (which is a successful guess with probability $1/(n^2+1)$). Next, invoke Env to obtain the inputs, and proceed to simulate the execution towards Adv (controlling the corrupted receiver) as in HYB^{h^*-1}, except that for honest parties P_i and P_j such that $(i-1)n + j = h^*$, use the challenge transcript trans and the challenge

key k. Finally, the attacker outputs whatever Env outputs, i.e., $b' = 0$ if Env outputs 0 and $b' = 1$ if Env outputs 1.

It holds that if k is the key corresponding to trans, then the attacker identically simulates HYB^{h^*-1}, whereas if k is uniformly sampled, then the attacker identically simulates HYB^{h^*}. Therefore, conditioned on guessing h^* correctly, the distinguishing probability of the attacker is the same as that of Env. We conclude that the attacker wins the KA experiment (Figure 14) with non-negligible probability. □

This concludes the proof of Lemma 3.5. □

Using the UC composition theorem, we obtain the following corollary.

Corollary 3.8 (THRMT for d-validated graphs)**.** *Let $n \in \mathbb{N}$ be polynomial in κ, let $d, t < n$, assume the existence of a key-agreement protocol. Let $(\mathcal{G}, \mathsf{Pred})$ be a d-validated graph class. Then, π_{rmt} instantiated with $\pi_{\mathsf{or\text{-}abort\text{-}d}}$ is a t-secure THRMT with respect to $\mathcal{G}$.*

As we consider semi-honest adversaries, t-secure THRMT readily yields t-secure THB for any $t < n$, by having the broadcaster send its message one at a time to every recipient. Further, for $t < n/2$ and assuming KA, the BGW protocol can be executed over the established broadcast protocol. Hence, we obtain the following corollaries.

Corollary 3.9 (THB and THSFE for d-validated graphs)**.** *Let $n \in \mathbb{N}$ be polynomial in κ, let $d < n$, assume the existence of a key-agreement protocol. Let $(\mathcal{G}, \mathsf{Pred})$ be a d-validated graph class. Then:*

1. *Let $t < n$. There exists t-secure THB with respect to $\mathcal{G}$.*
2. *Let $t < n/2$. For every efficiently computable n-party function f, there exists t-secure THSFE for f with respect to $\mathcal{G}$.*

3.3 Instantiating for Odd-Validated Graphs

We now turn to realize $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ with t-security for $t < n$ over graphs in which every pair of non-neighboring parties have an unknown-but-odd common neighbors which are validated by Pred.

Definition 3.10 (odd-validated graph)**.** *Given a graph $G = (V, E)$ of size n and a predicate Pred, we say the tuple (G, Pred) is an* odd-validated graph *if for every pair of non-adjacent vertices $u, v \in V$ (i.e., $(u, v) \notin E$), it holds that*

$$|\{w \in \mathcal{N}(u) \cap \mathcal{N}(v) \mid \mathsf{Pred}(w) = 1\}| \text{ is odd.}$$

Definition 3.11 (odd-validated graph-class)**.** *Let $n \in \mathbb{N}$, let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$, and let Pred be a predicate. The pair $(\mathcal{G}, \mathsf{Pred})$ is an* odd-validated graph-class *if for every $G \in \mathcal{G}$ it holds that (G, Pred) is a d-validated graph.*

Protocol $\pi_{\mathsf{or\text{-}abort\text{-}odd}}$

The n-party protocol is executed between the receiver P_R and its neighbors in the graph; the receiver emulates in its head every non-neighbor party P_i with $i \notin \mathcal{N}_G[R]$ to form an n-party protocol.

Input: Every P_i with $i \in [n]$ has an input $x_i \in \{0, 1\} \cup \{\mathsf{abort}\}$

Auxiliary input: A field $\mathbb{F}_q$ with $q = 2^\ell$ such that $q > n$, a linear circuit C specified as $\sum x_i$, and a

key-agreement protocol π_{ka} for keys of length ℓ.

The protocol:

1. *Establishing secure channels between receiver's neighbors.*
   (a) The receiver samples uniformly at random a permutation $\sigma : [n] \to [n]$.
   (b) The receiver emulates in its head every non-neighboring party $i \notin \mathcal{N}(R)$.
   (c) The receiver sends to each of its neighbors $i \in \mathcal{N}(R)$ and to the emulated parties $i \notin \mathcal{N}(R)$, the value $\sigma(i)$, (where i is the party's id).
   (d) Every ordered pair of parties P_i and P_j (where each is either a neighbor of P_R or emulated by P_R) execute the key-agreement protocol π_{ka} where the receiver is acting as a relay between them. Denote by $k_{i\to j}$ the resulting key.
   (e) At the end of this phase there are n^2 secure channels between all n parties: real receiver's neighbors and receiver's non-neighbors emulated by the receiver.
2. *Execute BGW for C.*
   The n parties execute the BGW protocol for the linear circuit C for $t < n$ over $\mathbb{F}_q$.
   (a) Every party P_i uses its input x_i; if the sender's input is abort, it proceeds with input 0.
   (b) Each party P_i secret shares its input x_i using an n-out-of-n Shamir secret sharing over $\mathbb{F}_q$ as $(s_{i\to 1}, \ldots, s_{i\to n})$. Next, for every $j \in [n]$, party P_i encrypts the jth share as $c_{i\to j} = s_{i\to j} \oplus k_{i\to j}$ and sends $c_{i\to j}$ to P_j via the receiver.
   (c) Upon receiving n encrypted shares $(c_{1\to i}, \ldots, c_{n\to i})$, party P_i decrypts each ciphertext as $s_{j\to i} = c_{j\to i} \oplus k_{j\to i}$ to obtain $(s_{1\to i}, \ldots, s_{n\to i})$. Next, P_i evaluates the circuit C over these shares to compute the output share τ_i.
   (d) Every P_i sends the resulting share τ_i to the receiver; if the party's input is abort, it does not send this message.
   (e) If the receiver receives n shares, it reconstructs and outputs the bit m^*; otherwise, it outputs $\perp$.

Figure 22: SFE protocol for $f_{\mathsf{or\text{-}abort}}$ over odd-validated graphs

Lemma 3.12. *Let $n \in \mathbb{N}$ be polynomial in κ, let $t < n$, and let π_{ka} be a key-agreement protocol. Let $(\mathcal{G}, \mathsf{Pred})$ be odd-validated graph-class. Then, for every $S, R \in [n]$, it holds that $\pi_{\mathsf{or\text{-}abort\text{-}odd}}$ (defined in Figure 22) realizes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or\text{-}abort}}}$ in the presence of PPT semi-honest t-adversaries.*

Proof. Security follows identically as in the proof of Lemma 3.5, so we are left with the proof of correctness.

Correctness. We need to show that the SFE protocol $\pi_{\mathsf{or\text{-}abort\text{-}odd}}$ correctly realizes the $f_{\mathsf{or\text{-}abort}}$ functionality.

If for any $i \in [n]$ the input is $x_i = \mathsf{abort}$ then in Step 2a party P_i uses 0 as input. In Step 2d party P_i will not send the final share, thus the reconstruction fails and the output is set to $\perp$. Otherwise, all parties have inputs in $\{0, 1\}$. Since the protocol is executed on an odd-common neighbor graph-class $\mathcal{G}_{\mathsf{odd\text{-}input}}$, then there is an odd amount of parties P_i such that $i \in \mathcal{N}(S) \cap \mathcal{N}(R)$, and so party P_i input is $x_i = m$, all other parties have $x_i = 0$. By the correctness of π_{ka}, each ordered pair of parties agree on a key except for negligible probability, in which case the computation of the BGW protocol for the linear circuit C is correct. Therefore, and as we operate over a binary field, the

circuit computation is:

$$\sum_{i=1}^{n} x_i = \sum_{\text{odd times}} m = m$$

This equals the output of $f_{\mathsf{or\text{-}abort}}$ which is m. □

Using the UC composition theorem, we obtain the following corollary.

Corollary 3.13 (THRMT for odd-validated graphs)**.** *Let n be polynomial in κ and let $t < n$, assume the existence of a key-agreement protocol. Let $(\mathcal{G}, \mathsf{Pred})$ be a odd-validated graph-class. Then, π_{rmt} instantiated with $\pi_{\mathsf{or\text{-}abort\text{-}odd}}$ is a t-secure THRMT with respect to $\mathcal{G}$.*

As we consider semi-honest adversaries, t-secure THRMT readily yields t-secure THB for any $t < n$, by having the broadcaster send its message one at a time to every recipient. Hence, we obtain the following corollaries.

Corollary 3.14 (THB and THSFE for odd-validated graphs)**.** *Let $n \in \mathbb{N}$ be polynomial in κ, let $d < n$, assume the existence of a key-agreement protocol. Let $(\mathcal{G}, \mathsf{Pred})$ be a odd-validated graph class. Then:*

1. *Let $t < n$. There exists t-secure THB with respect to $\mathcal{G}$.*
2. *Let $t < n/2$. For every efficiently computable n-party function f, there exists t-secure THSFE for f with respect to $\mathcal{G}$.*

4 Topology-Hiding RMT from Non-Linear MPC

In the general case of diameter-two graphs, there is no guarantee on the number of common neighbors between any two vertices, or whether its odd or even. In this section, we extend the technique from Section 3 that relied on BGW for linear circuits, to support honest-majority BGW for non-linear circuits; this enables achieving THB and THC for several graph classes not captured in Section 3. In Section 4.1 we present an RMT protocol in the SFE-hybrid model. Later, in Section 4.2 we instantiate the functionality using BGW and key agreement.

4.1 RMT Protocol in SFE-Hybrid Model with an Honest Majority

In this section, we denote by d_{max} the maximal degree of every graph in the graph class and by d_{min} the minimal degree. We define the protocol $\pi_{\mathsf{rmt\text{-}nl}}$ (for non-linear circuits) in the SFE-hybrid model, for computing the $(d_{\mathsf{max}} + 1)$-party solitary-output functionality $f_{\mathsf{or}}((x_{j_1}, \alpha_{j_1}), \ldots, (x_{j_{d_{\mathsf{max}}+1}}, \alpha_{j_{d_{\mathsf{max}}+1}})) = x_{j_1} \vee \ldots \vee x_{j_{d_{\mathsf{max}}+1}}$. Note that the second coordinate in each input is not taken into account in the computation, and here there is no special input for aborting the computation.

Protocol $\pi_{\mathsf{rmt\text{-}nl}}$

Auxiliary input: An n parties diameter-two graph-class $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$ and an integer $d_{\mathsf{max}} \in \mathbb{N}$ denoting the maximal degree in every graph $G \in \mathcal{G}$.

Input: The sender P_S, with $S \in [n]$, holds an input $m \in \{0, 1\}$.

Hybrid model: The protocol is defined in the $(\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}, \mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}})$-hybrid model.

The protocol:

1. *Initialization.*
   Each party P_i sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ and receives its neighbor-set $\mathcal{N}(i)$.
2. *Sender sends the message to its neighbors.*
   The sender P_S sends to each of its neighbors, P_i with $i \in \mathcal{N}(S)$, its input message m.
3. *Receiver obtains the message from its neighbors.*
   (a) Every neighbor of the receiver, P_i with $i \in \mathcal{N}(R)$, invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ as follows:
      i. If $i \in \mathcal{N}[S]$ with the message $(m, 1)$ (received from P_S).
      ii. Otherwise, with $(0, 0)$.
   (b) The receiver invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ by emulating $d_{\mathsf{max}} - \deg(R)$ parties sending $(0, 0)$ as input ($\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ is invoked using $d_{\mathsf{max}} + 1$ parties in total).
   (c) The receiver obtains an output $m^* \in \{0, 1\}$ from $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$.
4. *Output.* If P_R is a neighbor of P_S, then it outputs m received from P_S in Step 2; otherwise, P_R outputs m^* received from $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ in Step 3.

Figure 23: THRMT protocol for $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}$ in $(\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}, \mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}})$-hybrid model

Lemma 4.1. *Let $n \in \mathbb{N}$, let $t < n$, and let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$. Then, $\pi_{\mathsf{rmt\text{-}nl}}$ is a t-secure THRMT with respect to $\mathcal{G}$ in the $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$-hybrid model.*

Proof. Since the graph has diameter two, correctness follows directly from the protocol structure, and the correctness of the underlying secure function evaluation. The receiver P_R obtains the correct message in one of the following cases:

- If the receiver is a neighbor of the sender, then it gets the message directly from the sender in Step 2, and outputs it.
- Otherwise, there is at least one common neighbor v between the sender and the receiver. It holds that P_v receives the message m from the sender in Step 2 and invokes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ with m. All parties that are neighbors of the receiver but not neighbors of the sender invoke $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ with 0. By definition of f_{or} the output in this case is the Boolean OR of all input bits, which equals the message m.

Hence, the receiver obtains the correct output in any case.

For security, we will construct a simulator Sim that given the neighbor-set of the corrupted parties, indexed by $\mathcal{I} \subseteq [n]$, generates a simulated view for the corrupted parties that is computationally indistinguishable from their view in a real execution of the protocol. The simulator Sim runs every P_i with $i \in \mathcal{I}$ in its head, and does the following:

- To simulate the initialization phase, in Step 1 each corrupted party P_i with $i \in \mathcal{I}$ sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ and receives its neighbor-set $\mathcal{N}(i)$. The simulator forwards these messages to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$ and sends the response $\mathcal{N}(i)$ to each P_i.

In the input phase, in case the sender is corrupted, i.e., $S \in \mathcal{I}$, the simulator sends its input m to $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$; regardless, Sim receives the message m as the output from $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{rmt}}(\mathsf{P}_S, \mathsf{P}_R))$ (formally, Sim sends the "empty input" for a non-sender corrupted party).

- To simulate Step 2, if the sender is corrupted, i.e., $S \in \mathcal{I}$, Sim simulates receiving m on behalf of every honest party P_i for which $v \in \mathcal{N}(i)$. If the sender is honest, i.e., $S \notin \mathcal{I}$, then Sim sends m to every corrupted party P_i for which $S \in \mathcal{N}(i)$.
- To simulate Step 3, Sim simulates $\mathcal{F}^{f_{\mathsf{or}}}_{\mathsf{sfe}}$ towards the adversary as follows:
  1. If the receiver is corrupted, i.e., $R \in \mathcal{I}$:
     (a) Sim receives a value from each corrupted party P_i with $i \in \mathcal{I} \cap \mathcal{N}(R)$.
     (b) Sim receives a value from P_R for each emulated party, and in addition a value from P_R itself.
     (c) Sim sets the output to be the message m.
     (d) Eventually, Sim provides the output to P_R.
  2. If the receiver is honest, i.e., $R \notin \mathcal{I}$:
     (a) Sim receives a value for each corrupted party P_i with $i \in \mathcal{I} \cap \mathcal{N}(v)$. (Recall that only the receiver obtains the output of f_{or}, so there is nothing else to simulate when the receiver is honest.)

It is immediate to verify that the simulation is perfect, as the only difference between the view of the adversary in the real world and in the simulation is the way f_{or} is computed; however, both computations induce the same distribution. □

4.2 Instantiating for Degree-Bounded Graphs

We proceed to realize $\mathcal{F}^{f_{\mathsf{or}}}_{\mathsf{sfe}}$ for degree bounded graphs.

Protocol π_{or}

The $(d_{\mathsf{max}}+1)$-party protocol is executed between the receiver P_R and its neighbors in the graph; the receiver emulates in its head $d_{\mathsf{max}} - \deg(R)$ parties to form an $(d_{\mathsf{max}}+1)$-party protocol. For simplicity and by abuse of notation, denote the set of participating parties by $\mathsf{P}_1, \ldots, \mathsf{P}_{d_{\mathsf{max}}+1}$.

Input: Every P_i with $i \in [d_{\mathsf{max}}+1]$ has an input $(x_i, \alpha_i) \in \{0,1\} \times \{0,1\}$.

Auxiliary input: A field $\mathbb{F}_q$ with a prime $q > d_{\mathsf{max}}$, a circuit C specified as $(\sum \alpha_i)^{-1} \cdot \sum x_i$, and a key-agreement protocol π_{ka} for keys of length ℓ, where ℓ is the total number of bits sent by a party in the BGW protocol for C.

The protocol:

- *Establishing secure channels between receiver's neighbors.*
  1. The receiver samples uniformly at random a permutation $\sigma : [d_{\mathsf{max}}+1] \to [d_{\mathsf{max}}+1]$.
  2. The receiver emulates in its head $d_{\mathsf{max}} - \deg(R)$ parties, P_i with $i \notin \mathcal{N}(R)$.
  3. The receiver sends to each of its neighbors, P_i with $i \in \mathcal{N}(R)$, and to the emulated parties, P_i with $i \notin \mathcal{N}(R)$, the value $\sigma(i)$.

4. Every ordered pair of parties P_i and P_j (where each is either a neighbor of P_R or emulated by P_R) execute the key-agreement protocol π_{ka} where the receiver is acting as a relay between them. Denote by $k_{i\to j}$ the resulting key.
5. At the end of this phase there are $(d_{\mathsf{max}}+1)^2$ secure channels between all $(d_{\mathsf{max}}+1)$ parties: real receiver's neighbors and receiver's non-neighbors emulated by the receiver.

- *Execute BGW for C.*

  The $(d_{\mathsf{max}}+1)$ parties execute the BGW protocol for the circuit C for threshold $t < (d_{\mathsf{max}}+1)/2$ over $\mathbb{F}_q$.

  1. Every party P_i uses its input (x_i, α_i).
  2. Each party secret shares (separately) x_i and α_i using a $(t+1)$-out-of-$(d_{\mathsf{max}}+1)$ Shamir secret sharing over $\mathbb{F}_q$, and sends the jth shares to P_j for every $j \in [d_{\mathsf{max}}+1]$, encrypted using one-time pad with the relevant part of the key $k_{i\to j}$.
  3. The parties proceed to execute the BGW protocol the circuit C (with $2(d_{\mathsf{max}}+1)$ input wires) to compute the output share τ_i for each P_i. Every message from a party P_i to a party P_j is encrypted by P_i using the relevant part of $k_{i\to j}$, and decrypted by P_j accordingly.
  4. Each party P_i sends the resulting share to the receiver.
  5. Upon receiving $d_{\mathsf{max}}+1$ shares, the receiver reconstructs and outputs the bit m^*.

Figure 24: An SFE protocol for the function f_{or} assuming an honest majority

Definition 4.2 (degree-bounded graph-class)**.** *Let $n, d_{\mathsf{min}}, d_{\mathsf{max}} \in \mathbb{N}$ where $d_{\mathsf{min}} \le d_{\mathsf{max}} < n$ and let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$. $\mathcal{G}$ is called* $(d_{\mathsf{min}}, d_{\mathsf{max}})$-degree-bounded graph-class *if it holds that for every $G \in \mathcal{G}$ has minimal degree at least d_{min}, and maximal degree at most d_{max}.*

Lemma 4.3. *Let n be polynomial in κ, let $d_{\mathsf{max}}, d_{\mathsf{min}} \in \mathbb{N}$, let $t \le d_{\mathsf{min}} - d_{\mathsf{max}}/2$, and let π_{ka} be a key-agreement protocol for keys of length ℓ, where ℓ is the total number of bits sent by a party in the BGW protocol for C. Let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{diam\text{-}two}}(n)$ be a $(d_{\mathsf{min}}, d_{\mathsf{max}})$-degree-bounded graph-class.*

Then, for every $S, R \in [n]$, it holds that π_{or} (defined in Figure 20) realizes $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{or}}}$ in the presence of PPT semi-honest t-adversaries.

Proof. We will prove *correctness* and *security* separately.

Correctness. We need to show that the SFE protocol π_{or} correctly realizes the f_{or} functionality. Every party P_i has an input $(x_i, \alpha_i) \in \{0,1\} \times \{0,1\}$. As the protocol is executed by the neighborhood of a party in a diameter two graph-class, it is guaranteed that there exists at least one party P_i that is a neighbor of the sender and provides input $(m, 1)$. Every party P_j, with $j \notin \mathcal{N}(S) \cap \mathcal{N}(R)$, uses the input $(0,0)$. By the correctness of π_{ka}, each ordered pair of parties agree on a key except for negligible probability, in which case the computation of the BGW protocol for the linear circuit C is correct. Let $c \coloneqq |\mathcal{N}(S) \cap \mathcal{N}(R)|$. The circuit computes:

$$\Big(\sum_{i=1}^{d_{\mathsf{max}}+1} \alpha_i\Big)^{-1} \cdot \sum_{i=1}^{d_{\mathsf{max}}+1} x_i = c^{-1} \cdot c \cdot m = m.$$

This equals $\bigvee_{i=1}^{n} x_i$, which is the output of f_{or}.

Security. The proof follows similarly to the proof of Lemma 3.5, where the main difference is the instantiation of the SFE function. The security follows from the security of the BGW protocol in the honest-majority setting and from the security of the key-agreement protocol. Note that the requirement $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$ guarantees an honest majority in both cases:

- If the receiver is honest, then the $(d_{\mathsf{max}} - \deg(R))$ parties are honestly emulated, and the adversary can control t neighbors of the receiver, i.e., at most $t \leq \deg(R) \leq d_{\mathsf{min}}$ parties. As $d_{\mathsf{min}} \leq d_{\mathsf{max}}$ it holds that $d_{\mathsf{min}} - d_{\mathsf{max}}/2 \leq d_{\mathsf{max}}/2$, and from the requirement $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$ we obtain that
$$t \leq d_{\mathsf{min}} - \frac{d_{\mathsf{max}}}{2} \leq \frac{d_{\mathsf{max}}}{2} < \frac{d_{\mathsf{max}}+1}{2}.$$
- If the receiver is corrupted, then it can emulate $d_{\mathsf{max}} - \deg(R)$ parties in its head, and collude with additional $t-1$ neighbors. so the number of corrupted parties is
$$d_{\mathsf{max}} - \deg(R) + t \leq d_{\mathsf{max}} - \deg(R) + d_{\mathsf{min}} - \frac{d_{\mathsf{max}}}{2} \leq \frac{d_{\mathsf{max}}}{2} < \frac{d_{\mathsf{max}}+1}{2}.$$

This concludes the proof of Lemma 4.3. □

Using the UC composition theorem, we obtain the following corollary.

Corollary 4.4 (THRMT for degree-bounded graphs)**.** *Let $d \in \mathbb{N}$ and $d_{\mathsf{max}} \in \mathbb{N}$, Let n be polynomial in κ, let $d_{\mathsf{min}} \leq d_{\mathsf{max}} < n$ be integers, let $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$, and assume the existence of a key-agreement protocol. Let $\mathcal{G}$ be a $(d_{\mathsf{min}}, d_{\mathsf{max}})$-degree-bounded graph class. Then, $\pi_{\mathsf{rmt\text{-}nl}}$ instantiated with π_{or} is a t-secure THRMT with respect to $\mathcal{G}$.*

As we consider semi-honest adversaries, t-secure THRMT readily yields t-secure THB for any $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$, by having the broadcaster send its message one at a time to every recipient. Further, as $d_{\mathsf{min}} \leq d_{\mathsf{max}}$ and $d_{\mathsf{max}} < n$, it holds that $t < n/2$ in this case. Hence, we obtain the following corollaries.

Corollary 4.5 (THB and THSFE for degree-bounded graphs)**.** *Let $n \in \mathbb{N}$ be polynomial in κ, let $d_{\mathsf{min}} \leq d_{\mathsf{max}} < n$, let $t \leq d_{\mathsf{min}} - d_{\mathsf{max}}/2$, and assume the existence of a key-agreement protocol. Let $\mathcal{G}$ be a $(d_{\mathsf{min}}, d_{\mathsf{max}})$-degree-bounded graph class. Then:*

1. *There exists t-secure THB with respect to $\mathcal{G}$.*
2. *For every efficiently computable n-party function f, there exists t-secure THSFE for f with respect to $\mathcal{G}$.*

5 Lower Bound

We proceed to prove the lower bound, Theorem 1.1, showing that a certain graph-class containing a join of star graphs, necessitates key agreement for t-secure THB.

Definition 5.1 (join of graphs)**.** *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be graphs. The join of G_1 and G_2 is defined to be the graph $G = (V, E)$ where $V = V_1 \cup V_2$ (i.e., disjoint union) and*
$$E = E_1 \cup E_2 \cup \{(u, v) \mid u \in V_1, v \in V_2\}.$$

Definition 5.2 (star graph)**.** *A star graph of size n is an n-node graph $G = (V, E)$ such that there exists a node u of degree $\deg(u) = n - 1$ and the degree of every node $v \in V \setminus \{u\}$ is $\deg(v) = 1$.*

We proceed to define the graph classes for the lower bound, specified as a join of a single node (the center) with the disjoint union of a t-star and a $(t+1)$-star; see Figure 4 for examples.

Definition 5.3 (graph class of joined stars)**.** *Let $t \in \mathbb{N}$, let G_1 be a star graph of size t, and let G_2 be a star graph of size $t+1$. The graph $G_{\mathsf{join\text{-}stars}}(t)$ that is defined as the join of a vertex v with the disjoint union of G_1 and G_2 is said to be a join of two stars.*

The graph class $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$ is defined to be the isomorphically closed graph class associated with $G_{\mathsf{join\text{-}stars}}(t)$.

Theorem 5.4. *Let $t \in \mathbb{N}$. If there exists a t-secure IND-CTA broadcast protocol for $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$, then there exists a key-agreement protocols for 1-bit keys that is secure against an eavesdropper.*

Proof. Let π be a t-secure IND-CTA broadcast protocol for $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$, where node $\boxed{t+2}$ is the broadcaster. Via sequential composition, we may assume π is a κ-bit broadcast protocol. As part of the KA construction we will use two specific labeled graphs from $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$, denoted G_0 and G_1, that are defined as follows (see Figure 25 for an illustration):

- In G_0 the center node is $\boxed{1}$. The star of size t consists of $\boxed{2}, \ldots, \boxed{t+1}$ and its center is $\boxed{2}$. The star of size $t+1$ consists of $\boxed{t+2}, \ldots, \boxed{2t+2}$ and its center is $\boxed{t+2}$.
- In G_1 the center is $\boxed{3}$. The star of size t consists of $\boxed{1}$, $\boxed{2}$, and $\boxed{4}, \ldots, \boxed{t+1}$ and its center is $\boxed{1}$. The star of size $t+1$ consists of $\boxed{t+2}, \ldots, \boxed{2t+2}$ and its center is $\boxed{t+2}$.

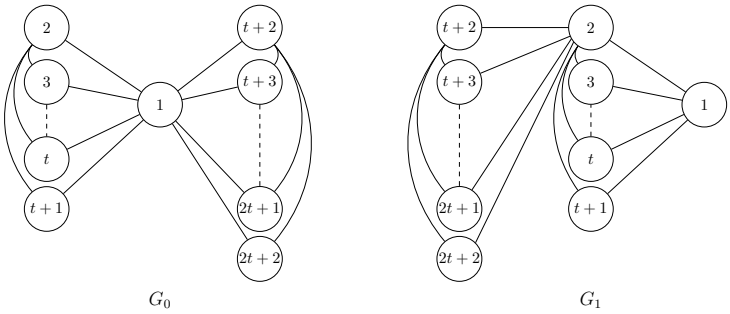


Figure 25: The graphs G_0 and G_1 used for the KA protocol.

We use the protocol π and the definition of the graphs G_0 and G_1 to construct the following key-agreement protocol for 1-bit keys.

Protocol Key-Agreement

Public Parameters: An integer $t \in \mathbb{N}$ and a t-secure broadcast protocol π for $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$.

The protocol:

- Alice samples two random strings $m_1, m_2 \leftarrow \{0,1\}^\kappa$ and sends them to Bob.
- Alice samples a random bit $c_A \leftarrow \{0,1\}$, and similarly, Bob samples a random bit $c_B \leftarrow \{0,1\}$. Next, they jointly simulate two instances of π, communicating in each instance the messages between

the party ① and the parties ②, …, $\boxed{t+1}$.

- If $c_A = 1$, Alice will simulate in the first execution parties ②, …, $\boxed{2t+2}$ (connected according to G_1), with $\boxed{t+2}$ broadcasting m_1, and in the second execution Alice will simulate parties ②, …, $\boxed{t+1}$ (connected according G_0).
  If $c_A = 0$, Alice will simulate in the first execution ②, …, $\boxed{t+1}$ (connected according to G_0), and in the second execution Alice will simulate ②, …, $\boxed{2t+2}$, where $\boxed{t+2}$ broadcasting m_2 (connected according G_1).
- If $c_B = 1$, Bob will simulate in the first execution ① (connected according to G_1), and in the second execution Bob will simulate ①, and $\boxed{t+2}$, …, $\boxed{2t+2}$ (connected according G_0), with $\boxed{t+2}$ broadcasting m_1.
  If $c_B = 0$, Bob will simulate in the first execution ①, and $\boxed{t+2}$, …, $\boxed{2t+2}$ (connected according to G_0), with $\boxed{t+2}$ broadcasting m_2, and in the second execution Bob will simulate ① (connected according G_1).

- *Output.*
  - If node ② outputs m_1 in the first instance of π and m_2 in the second, Alice outputs c_A; otherwise, Alice outputs $\perp$.
  - If node ① outputs m_1 in the first instance of π and m_2 in the second, Bob outputs c_B; otherwise, Bob outputs $\perp$.

Figure 26: Key agreement protocol from t-secure THB on $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$

We proceed to prove correctness and security of the KA protocol.

Lemma 5.5. *The protocol in Figure 26 is a key-agreement protocol for* 1*-bit keys.*

Proof. Consider the coin tosses of Alice and Bob and define the events that $c_A = c_B$ and that $c_A \neq c_B$ (each occurring with probability 1/2). We will show that in the first case both Alice and Bob output the same bit with overwhelming probability, and in the second case both Alice and Bob output $\perp$ with overwhelming probability. Via sequential repetition of upto κ phases, until outputting a bit (using the sequential composition lemma, Lemma 2.10), the parties will output the same bit with overwhelming probability.

The case of $c_A = c_B$. In this case, in each execution of π exactly one of Alice and Bob is simulating the broadcasting node $\boxed{t+2}$. By correctness of π, all parties (in particular ① and ②) will output the string m_1 in the first execution and m_2 in the second execution with overwhelming probability. Thus, Alice will output c_A and Bob will output c_B with overwhelming probability.

The case of $c_A \neq c_B$. If $c_A = 0$ and $c_B = 1$, then neither Alice nor Bob is simulating the broadcasting node $\boxed{t+2}$ in the first execution. In which case, all outputs of π in this simulation are independent of m_1. Thus, the probability that either ① or ② outputs m_1 in the execution is at most $2 \cdot 2^{-\kappa}$. Conversely, if $c_A = 1$ and $c_B = 0$, then neither party is simulating node $\boxed{t+2}$ in the second execution and all outputs are independent of m_2. Similarly, the probability that either ① or ② outputs m_2 in this case is at most $2 \cdot 2^{-\kappa}$.

This concludes the proof of Lemma 5.5. □

Lemma 5.6. *The key-agreement protocol in Figure 26 is secure in the presence of an eavesdropper.*

Proof. It suffices to show that conditioned on $c_A = c_B$, an eavesdropper Eve who has access to the communication between Alice and Bob has no advantage in determining the key-agreement output bit (whether $c_A = c_B = 0$ or $c_A = c_B = 1$). Recall that Eve has access to the communication transcript between Alice and Bob, that consists of the communication between party $\textcircled{1}$ (emulated by Bob) and parties $\textcircled{2}, \ldots, \boxed{t+1}$ (emulated by Alice), in both executions of π.

Recall that the graph G_0 corresponds to the first execution of π when $c_A = c_B = 0$ and to the second execution of π when $c_A = c_B = 1$. Similarly, the graph G_1 corresponds to the first execution of π when $c_A = c_B = 1$ and to the second execution of π when $c_A = c_B = 0$.

A reduction from two executions to one execution. We now use the reduction from [BBC+19, Claim 3.3][4] which shows that if Eve has a non-negligible advantage in distinguishing between the a uniform bit and the bit agreed by Alice and Bob, then there exists an efficient distinguisher between the transcripts of the communication between party $\textcircled{1}$ and parties $\textcircled{2}, \ldots, \boxed{t+1}$ when communicating over G_0 or over G_1 in a *single* run of π (whereas Eve has to distinguish between the transcript for G_0 followed by the transcript for G_1, from the transcript for G_1 followed by the transcript for G_0).

More specifically, given a graph $G \in \mathcal{G}_{\mathsf{join\text{-}stars}}(t)$ in which party $\textcircled{1}$ in a neighbor of each of the parties $\textcircled{2}, \ldots, \boxed{t+1}$, denote by $\textsc{tran}(G)$ the random variable representing the transcript of the communication between party $\textcircled{1}$ and parties $\textcircled{2}, \ldots, \boxed{t+1}$ when π is executed on G. In particular, for the graphs G_0 and G_1 it holds that

$$\mathsf{advantage}(\text{Eve}) \leq 2 \cdot \sup_{\text{PPT } \mathsf{Adv}} \left| \Pr\left[\mathsf{Adv}(\textsc{tran}(G_0)) = 1\right] - \Pr\left[\mathsf{Adv}(\textsc{tran}(G_1)) = 1\right] \right|$$

The reduction works by constructing an adversary Adv for π from Eve. Adv corrupts $\textcircled{1}$ and obtains the transcript $\textsc{tran}(G_i)$ for some unknown $i \in \{0,1\}$; next, Adv flips two coins $j, b \in \{0,1\}$. Adv then generates a transcript $\textsc{tran}(G_j)$ corresponding to an execution of π on G_j, and gives both transcripts $\textsc{tran}(G_i)$ and $\textsc{tran}(G_j)$ to Eve in the order determined by coin b. With probability $1/2$ it holds that $i = j$ and Eve is of no help, and with probability $1/2$ it holds that $i \neq j$ and Adv inherits Eve's distinguishing advantage. It can be shown that $\mathsf{advantage}(\mathsf{Adv}) \geq \mathsf{advantage}(\text{Eve})/2$.

Bounding the distinguishing probability of Eve. We now turn to prove that the distinguishing probability of any PPT Eve for the KA protocol is negligible. Toward this goal, we will use three more labeled graphs, other than G_0 and G_1, from the graph class $\mathcal{G}_{\mathsf{join\text{-}stars}}(t)$ (see Figure 27 for an illustration):

- In H_0 the center node is $\textcircled{1}$. The star of size t consists of $\boxed{t+3}, \ldots, \boxed{2t+2}$ and its center is $\boxed{t+3}$. The star of size $t+1$ consists of $\textcircled{2}, \ldots, \boxed{t+2}$ and its center is $\boxed{t+2}$.
- In H_1 the center node is $\boxed{t+2}$. The star of size t consists of $\boxed{t+3}, \ldots, \boxed{2t+2}$ and its center is $\boxed{t+3}$. The star of size $t+1$ consists of $\textcircled{1}, \textcircled{2}, \ldots, \boxed{t+2}$ and its center is $\textcircled{1}$.
- In H_2 the center node is $\textcircled{2}$. The star of size t consists of $\boxed{t+3}, \ldots, \boxed{2t+2}$ and its center is $\boxed{t+3}$. The star of size $t+1$ consists of $\textcircled{1}$ together with $\textcircled{3}, \ldots, \boxed{t+2}$ and its center is $\textcircled{1}$.

[4] The claim itself cannot be invoked as the graphs are different, but the same proof works in similar lines.

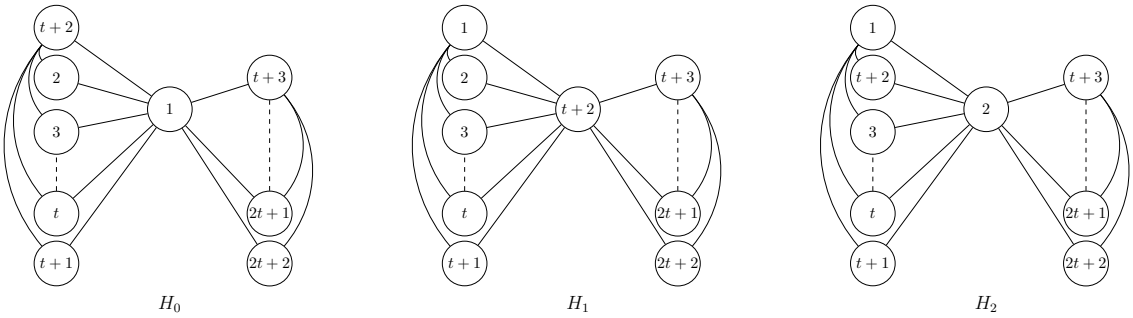


Figure 27: The graphs H_0, H_1, H_2 used for the security proof of the KA protocol.

Having defined the graphs G_0, G_1, H_0, H_1, and H_2, we prove a series of claims showing that the communication between party ① and parties ②,…,$\boxed{t+1}$ in an execution of π remains computationally indistinguishable over these graphs. We refer the reader to Figure 28 for an illustration of these claims. Given a graph $G \in \mathcal{G}_{\mathsf{join\text{-}stars}}(t)$ in which party ① in a neighbor of each of the parties ②, …, $\boxed{t+1}$, denote by $\text{VIEW}_1(G)$ the random variable representing the view of ① in an execution of π over G, and by $\text{VIEW}_{2,\ldots,t+1}(G)$ the joint view of parties ②,…,$\boxed{t+1}$. Let Dist be a PPT distinguisher.

Claim 5.7. $|\Pr[\mathsf{Dist}(\text{TRAN}(G_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_0)) = 1]| \leq \mathsf{negl}(\kappa)$.

Proof. By the security of π we obtain that

$$|\Pr[\mathsf{Dist}(\text{TRAN}(G_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_0)) = 1]| \leq$$
$$|\Pr[\mathsf{Dist}(\text{VIEW}_1(G_0)) = 1] - \Pr[\mathsf{Dist}(\text{VIEW}_1(H_0)) = 1]| \leq \mathsf{negl}(\kappa).$$

The first inequality holds since the transcript between party ① and parties ②,…,$\boxed{t+1}$ is fully contained in the view of party ①, and the second inequality follows from the security of π, since ① has the same neighborhood in both G_0 and H_0. □

Claim 5.8. $|\Pr[\mathsf{Dist}(\text{TRAN}(H_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1]| \leq \mathsf{negl}(\kappa)$.

Proof. By the security of π we obtain that

$$|\Pr[\mathsf{Dist}(\text{TRAN}(H_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1]| \leq$$
$$|\Pr[\mathsf{Dist}(\text{VIEW}_{2,3,\ldots,t+1}(H_0)) = 1] - \Pr[\mathsf{Dist}(\text{VIEW}_{2,3,\ldots,t+1}(H_1)) = 1]| \leq \mathsf{negl}(\kappa).$$

The first inequality holds since the transcript between party ① and parties ②,…,$\boxed{t+1}$ is fully contained in the joint view of parties ②,…,$\boxed{t+1}$, and the second inequality follows from the security of π, since ②,…,$\boxed{t+1}$ have the same neighborhood in both H_0 and H_1. □

Claim 5.9. $|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1]| \leq \mathsf{negl}(\kappa)$.

Proof. By the security of π we obtain that

$$|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1]| \leq$$
$$|\Pr[\mathsf{Dist}(\text{VIEW}_1(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{VIEW}_1(H_2)) = 1]| \leq \mathsf{negl}(\kappa).$$

The first inequality holds since the transcript between party ① and parties ②,…,$\boxed{t+1}$ is fully contained in the view of party ①, and the second inequality follows from the security of π, since ① has the same neighborhood in both H_1 and H_2. □

Claim 5.10. $|\Pr[\mathsf{Dist}(\textsc{tran}(H_2))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| \le \mathsf{negl}(\kappa)$.

Proof. By the security of π we obtain that

$$|\Pr[\mathsf{Dist}(\textsc{tran}(H_2))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| \le$$
$$|\Pr[\mathsf{Dist}(\textsc{view}_{2,3,\ldots,t+1}(H_2))=1]-\Pr[\mathsf{Dist}(\textsc{view}_{2,3,\ldots,t+1}(G_1))=1]| \le \mathsf{negl}(\kappa).$$

The first inequality holds since the transcript between party ① and parties ②,…,$\boxed{t+1}$ is fully contained in the joint view of parties ②,…,$\boxed{t+1}$, and the second inequality follows from the security of π, since ②,…,$\boxed{t+1}$ have the same neighborhood in both H_2 and G_1. □

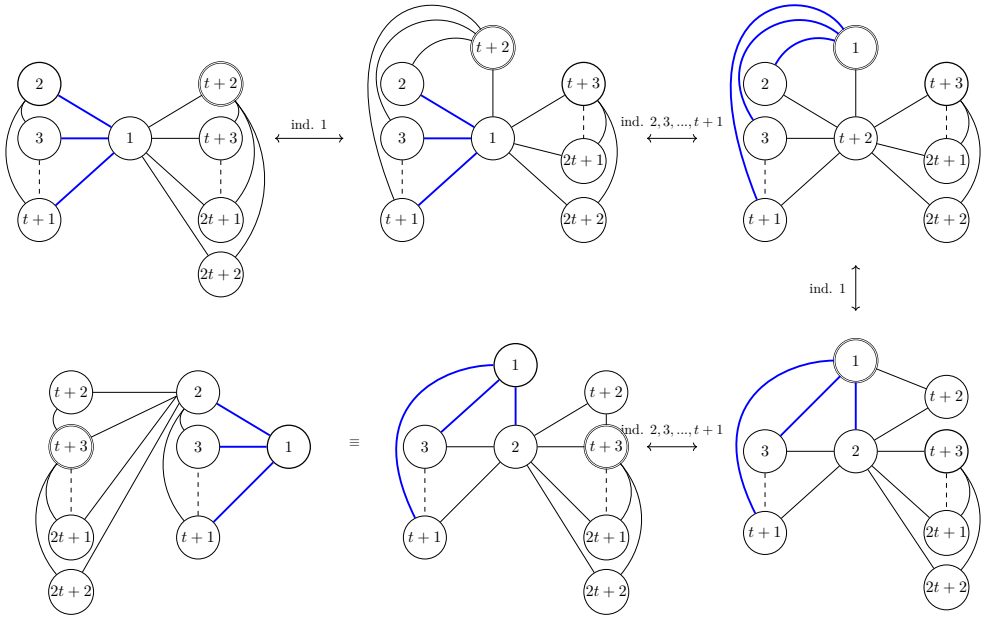


Figure 28: Hybrid steps in the phantom jump.

Finally, we combine the claims above to show that the distinguishing probability between $\textsc{tran}(G_0)$ and $\textsc{tran}(G_1)$ is negligible.

$$|\Pr[\mathsf{Dist}(\textsc{tran}(G_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| =$$
$$|\Pr[\mathsf{Dist}(\textsc{tran}(G_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]+\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| \le$$
$$|\Pr[\mathsf{Dist}(\textsc{tran}(G_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]|+|\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]|$$

By Claim 5.7, we obtain that $|\Pr[\mathsf{Dist}(\textsc{tran}(G_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]|$ (in blue) is negligible . Therefore, it is left to show that $|\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]|$ is negligible. Indeed

$$|\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| =$$
$$|\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(H_1))=1]+\Pr[\mathsf{Dist}(\textsc{tran}(H_1))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]| \le$$
$$|\Pr[\mathsf{Dist}(\textsc{tran}(H_0))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(H_1))=1]|+|\Pr[\mathsf{Dist}(\textsc{tran}(H_1))=1]-\Pr[\mathsf{Dist}(\textsc{tran}(G_1))=1]|$$

By Claim 5.8, it holds that $|\Pr[\mathsf{Dist}(\text{TRAN}(H_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1]|$ (in blue) is negligible . Therefore, it is left to show that $|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]|$ is negligible.

$|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]| =$
$|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1] + \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]| \leq$
$|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1]| + |\Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]|$

By Claim 5.9, it holds that $|\Pr[\mathsf{Dist}(\text{TRAN}(H_1)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1]|$ (in blue) is negligible and by Claim 5.10 it holds that $|\Pr[\mathsf{Dist}(\text{TRAN}(H_2)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]|$ is negligible. We therefore conclude that

$$|\Pr[\mathsf{Dist}(\text{TRAN}(G_0)) = 1] - \Pr[\mathsf{Dist}(\text{TRAN}(G_1)) = 1]| \leq \mathsf{negl}(\kappa).$$

This concludes the proof of Lemma 5.6. □

This concludes the proof of Theorem 5.4. □

Bibliography

[ACS22] Gilad Asharov, Ran Cohen, and Oren Shochat. Static vs. adaptive security in perfect MPC: A separation and the adaptive security of BGW. In *Proceedings of the 3rd Conference on Information-Theoretic Cryptography (ITC)*, volume 230, pages 15:1–15:16, 2022.

[AL17] Gilad Asharov and Yehuda Lindell. A full proof of the BGW protocol for perfectly secure multiparty computation. *Journal of Cryptology*, 30(1):58–151, 2017.

[ALM17] Adi Akavia, Rio LaVigne, and Tal Moran. Topology-hiding computation on all graphs. In *37th Annual International Cryptology Conference (CRYPTO), part I*, pages 447–467, 2017.

[AM17] Adi Akavia and Tal Moran. Topology-hiding computation beyond logarithmic diameter. In *36th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part III*, pages 609–637, 2017.

[BBC+19] Marshall Ball, Elette Boyle, Ran Cohen, Tal Malkin, and Tal Moran. Is information-theoretic topology-hiding computation possible? In *Proceedings of the 17th Theory of Cryptography Conference (TCC), part I*, pages 502–530, 2019.

[BBC+20] Marshall Ball, Elette Boyle, Ran Cohen, Lisa Kohl, Tal Malkin, Pierre Meyer, and Tal Moran. Topology-hiding communication from minimal assumptions. In *Proceedings of the 18th Theory of Cryptography Conference (TCC), part II*, pages 473–501, 2020.

[BBC24] D'or Banoun, Elette Boyle, and Ran Cohen. Information-theoretic topology-hiding broadcast: Wheels, stars, friendship, and beyond. In *5th Conference on Information-Theoretic Cryptography, ITC 2024*, volume 304 of *LIPIcs*, pages 1:1–1:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[BBKM23] Marshall Ball, Alexander Bienstock, Lisa Kohl, and Pierre Meyer. Towards topology-hiding computation from oblivious transfer. In *Proceedings of the 21st Theory of Cryptography Conference (TCC), part I*, pages 349–379, 2023.

[BBMM18] Marshall Ball, Elette Boyle, Tal Malkin, and Tal Moran. Exploring the boundaries of topology-hiding computation. In *37th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part III*, pages 294–325, 2018.

[BCM26] Elette Boyle, Ran Cohen, and Pierre Meyer. Maintaining sublinear locality over time: Adaptively secure MPC on a reusable hidden graph. Eurocrypt'26 (to appear), 2026.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–10, 1988.

[BMM99] Amos Beimel, Tal Malkin, and Silvio Micali. The all-or-nothing nature of two-party secure computation. In Michael J. Wiener, editor, *18th Annual International Cryptology Conference (CRYPTO)*, volume 1666 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 1999.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 136–145, 2001.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 11–19, 1988.

[ERS66] Paul Erdös, Alfréd Rényi, and Vera T. Sós. On a problem of graph theory. *Studia Sci. Math. Hungar.*, 1:215–235, 1966.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 218–229, 1987.

[HMTZ16] Martin Hirt, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Network-hiding communication and applications to multi-party protocols. In *36th Annual International Cryptology Conference (CRYPTO), part II*, pages 335–365, 2016.

[Kil91] Joe Kilian. A General Completeness Theorem for Two-Party Games. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 553–560, 1991.

[Kil00] Joe Kilian. More general completeness theorems for secure two-party computation. In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 316–324. ACM, 2000.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall, CRC Press, third edition, 2014.

[KMTZ13] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In *Proceedings of the 10th Theory of Cryptography Conference (TCC)*, pages 477–498, 2013.

[Li22] Shuaishuai Li. Towards practical topology-hiding computation. In Shweta Agrawal and Dongdai Lin, editors, *28th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), part I*, pages 588–617, 2022.

[LZM+18] Rio LaVigne, Chen-Da Liu Zhang, Ueli Maurer, Tal Moran, Marta Mularczyk, and Daniel Tschudi. Topology-hiding computation beyond semi-honest adversaries. In *Proceedings of the 16th Theory of Cryptography Conference (TCC), part II*, pages 3–35, 2018.

[LZM+20] Rio LaVigne, Chen-Da Liu Zhang, Ueli Maurer, Tal Moran, Marta Mularczyk, and Daniel Tschudi. Topology-hiding computation for networks with unknown delays. In *Proceedings of the 23rd International Conference on the Theory and Practice of Public-Key Cryptography (PKC), part II*, pages 215–245, 2020.

[MOR15] Tal Moran, Ilan Orlov, and Silas Richelson. Topology-hiding computation. In *Proceedings of the 12th Theory of Cryptography Conference (TCC), part I*, pages 159–181, 2015.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 162–167, 1986.

A UC Framework

We present a highly informal overview of the UC framework and refer the reader to [Can01] for further details. The framework is based on the real/ideal paradigm for arguing about the security of a protocol.

The real model. An execution of a protocol π in the real model consists of n PPT *interactive Turing machines* (ITMs) $\mathsf{P}_1, \ldots, \mathsf{P}_n$ representing the parties, along with two additional ITMs: an *adversary* Adv, describing the behavior of the corrupted parties and an *environment* Env, representing the external network environment in which the protocol operates. The environment gives inputs to the honest parties, receives their outputs, and can communicate with the adversary at any point during the execution. It is known that security against the *dummy* adversary (that forwards every message it sees to the environment and acts according to the environment's instructions) is sufficient to achieve security against arbitrary adversaries. Throughout, we consider *synchronous* protocols that proceeds in rounds (this can be formally modeled using the $\mathcal{F}_{\mathsf{sync}}$ functionality [Can01], or using the synchronous framework of [KMTZ13]) and *semi-honest (passive) security* (where corrupted parties continue following the protocol, but reveal their internal state to the adversary). We will consider both *static* corruptions (where Adv chooses the corrupted parties at the onset of the protocol) and *adaptive* corruptions (where Adv can dynamically corrupt parties based on information gathered during the computation), and will explicitly mention at any section which type of corruptions are considered. An t-adversary can corrupt up to t parties during the protocol.

The ideal model. A computation in the ideal model consists of n *dummy* parties $\tilde{\mathsf{P}}_1, \ldots, \tilde{\mathsf{P}}_n$, an *ideal-model adversary* (simulator) Sim, an *environment* Env, and an *ideal functionality* $\mathcal{F}$. As in the real model, the environment gives inputs to the honest (dummy) parties, receives their outputs, and can communicate with the ideal-model adversary at any point during the execution. The dummy parties act as channels between the environment and the ideal functionality, meaning that they send the inputs received from Env to $\mathcal{F}$ and vice-versa. The ideal functionality $\mathcal{F}$ defines the desired behaviour of the computation. $\mathcal{F}$ receives the inputs from the dummy parties, executes the desired computation and sends the output to the parties. The ideal-model adversary does not see the communication between the parties and the ideal functionality, however, Sim can corrupt dummy parties (statically or dynamically) and may communicate with $\mathcal{F}$ according to its specification.

Security definition. We present the definition for static and semi-honest adversaries.

We say that a protocol π $\mathsf{UC\text{-}realizes}$ (with computational security) an ideal functionality $\mathcal{F}$ in the presence of static semi-honest t-adversaries, if for any PPT static semi-honest t-adversary Adv and any PPT environment Env, there exists a PPT ideal-model t-adversary Sim such that

the output distribution of Env in the ideal-model computation of $\mathcal{F}$ with Sim is *computationally indistinguishable* from its output distribution in the real-model execution of π with Adv.

We say that a protocol π UC-realizes (with information-theoretic security) an ideal functionality $\mathcal{F}$ if the above holds even for computationally unbounded Adv, Env, and Sim. In that case the requirement is for the output distribution of Env in the ideal-model computation to be *statistically close* to its output distribution in the real-model execution. If the environment's outputs are identically distributed, we say that π UC-realizes $\mathcal{F}$ with *perfect* security.

The hybrid model. The *$\mathcal{F}$-hybrid model* is a combination of the real and ideal models, it extends the real model with an ideal functionality $\mathcal{F}$. The parties communicate with each other in exactly the same way as in the real model; however, they can also interact with $\mathcal{F}$ as in the ideal model. An important property of the UC framework is that the ideal functionality $\mathcal{F}$ in an $\mathcal{F}$-hybrid model can be replaced with a protocol that UC-realizes $\mathcal{F}$. The composition theorem of [Can01] states the following.

Theorem A.1 ([Can01], informal)**.** *Let ρ be a protocol that UC-realizes $\mathcal{F}$ in the presence of adaptive semi-honest t-adversaries, and let π be a protocol that UC-realizes $\mathcal{G}$ in the $\mathcal{F}$-hybrid model in the presence of adaptive semi-honest t-adversaries. Then, for any PPT adaptive semi-honest t-adversary* Adv *and any PPT environment* Env*, there exists a PPT adaptive semi-honest t-adversary* Sim *in the $\mathcal{F}$-hybrid model such that the output distribution of* Env *when interacting with the protocol π and* Sim *is computationally indistinguishable from its output distribution when interacting with the protocol π^ρ (where every call to $\mathcal{F}$ is replaced by an execution of ρ) and* Adv *in the real model.*