J Cryptol (2023)36:39
https://doi.org/10.1007/s00145-023-09473-3

Journal of
CRYPTOLOGY

Correction

Topology-Hiding Communication from Minimal Assumptions*

Marshall Ball
Columbia University, New York City, USA
marshall@cs.columbia.edu

Elette Boyle
Reichman University, Herzliya, Israel
elette.boyle@runi.ac.il

Ran Cohen
Northeastern University, Boston, USA
rancohen@ccs.neu.edu

Lisa Kohl
Cryptology Group, CWI Amsterdam, Amsterdam, The Netherlands
lisa.kohl@cwi.nl

Tal Malkin
Columbia University, New York City, USA
tal@cs.columbia.edu

Pierre Meyer
Reichman University, Herzliya, Israel
École Normale Supérieure de Lyon, Lyon, France
pierre.meyer@ens-lyon.fr

Tal Moran
Spacemesh, New York, USA
Northeastern University, Boston, USA
Reichman University, Herzliya, Israel
talm@runi.ac.il

Communicated by Amit Sahai

Received 29 March 2021 / Revised 12 June 2023 / Accepted 14 June 2023
Online publication 12 September 2023

Abstract. *Topology-hiding broadcast* (THB) enables parties communicating over an incomplete network to broadcast messages while hiding the topology from within a given class of graphs. THB is a central tool underlying general *topology-hiding secure computation* (THC) (Moran et al. TCC'15). Although broadcast is a privacy-free task, it was recently shown that THB for certain graph classes necessitates computational assumptions, even in the semi-honest setting, and even given a single corrupted party. In

*A preliminary version of this work appeared at *TCC 2020* [5]

this work, we investigate the minimal assumptions required for topology-hiding communication: both *Broadcast* or *Anonymous Broadcast* (where the broadcaster's identity is hidden). We develop new techniques that yield a variety of necessary and sufficient conditions for the feasibility of THB/THAB in different cryptographic settings: information theoretic, given existence of key agreement, and given existence of oblivious transfer. Our results show that feasibility can depend on various properties of the graph class, such as *connectivity*, and highlight the role of different properties of topology when kept hidden, including *direction*, *distance*, and/or *distance-of-neighbors* to the broadcaster. An interesting corollary of our results is a dichotomy for THC with a public number of at least three parties, secure against one corruption: information-theoretic feasibility if all graphs are 2-connected; necessity and sufficiency of key agreement otherwise.

Keywords. Foundations, Secure multiparty computation, Topology-hiding computation.

1. Introduction

Reliable communication between a set of mutually distrustful parties lies at the core of virtually any distributed protocol, ranging from consensus tasks [25,30] to secure multiparty computation [8,13,19,33]. Classical protocols from the '80 s considered complete communication graphs between the parties, where each pair of parties is connected by a communication channel. However, in many real-life scenarios the parties are not pairwise connected; this raises the need for distributed interactive computations, and in particular communication protocols, over an incomplete graph. Often, the *network topology itself* may be sensitive information that should not be revealed by the protocol.

Topology-hiding broadcast With this motivation, Moran et al. [29] formalized the concept of *topology-hiding computation (THC)*. Here, the goal is to allow parties who see only their immediate neighborhood (and possibly know that the graph belongs to some class), to securely compute arbitrary functions without revealing any additional information about the graph topology other than the output. (Computations on the graphs, e.g., establishing routing tables, are also supported.) THC is of theoretical interest, but is also motivated by real-world settings where it is desired to keep the underlying communication graph private. These include social networks, ISP networks, ad hoc (or mesh) networks, vehicle-to-vehicle communications, and possible approaches for contact tracing.

Given the existence of general MPC protocols, achieving THC for arbitrary functions hinges on communicating in a topology-hiding way, rather than on keeping inputs private. In particular, a core bottleneck for achieving general THC is the special case of *topology-hiding broadcast (THB)*, where a designated party (the broadcaster) reliably sends its message to all other parties. Indeed, given an MPC protocol for a function f defined in the broadcast model (where *all* communication is sent via a broadcast channel, possibly encrypted),[1] the parties can replace the broadcast channel by a THB protocol to obtain a THC protocol for the function f.

[1] Such protocols exist in the honest-majority setting assuming key agreement, and thus under this assumption, THB implies THC. In the information-theoretic setting, THC can be strictly stronger, as we will see.

Although broadcast is a privacy-free task, realizing THB turns out to be challenging, even in the semi-honest setting where all parties follow the protocol. This is in stark contrast to standard (topology-revealing) broadcast, which is trivially achievable in the semi-honest setting, e.g., simply "flooding" the network, forwarding received messages. For general semi-honest corruptions, the best THB constructions follow from a series of works [1,2,21,26,29], culminating in THB (as well as THC) protocols for all graphs [2]. However, even for THB, all known protocols require structured public-key cryptographic assumptions, such as QR, DDH, or LWE.[2] The use of strong assumptions was justified by Ball et al. [3] who showed that without an honest majority, even THB implies oblivious transfer (OT).[3]

A central paradigm in standard (topology-revealing) secure computation is to exchange *cryptographic* assumptions with an *honest-majority* assumption [8,13,31]. A recent work of Ball et al. [4] asked whether such a paradigm can be applied in the topology-hiding realm. The results of [4] demonstrated that answering this question is more subtle than meets the eye, even when considering the basic case of *one semi-honest corruption*. On the one hand, they showed that information-theoretic THB (IT-THB) can be achieved for the graph class of *cycles*, where the protocol hides the ordering of parties within the cycle. On the other hand, they identified that THB for *paths* of $n \geq 4$ nodes (again hiding ordering) implies key agreement.

This work In a sense, [4] unveiled the tip of the iceberg, revealing a range of questions: Which aspects of the topology can be hidden information theoretically, and which require cryptographic hardness? Is key agreement sufficient for 1-corruption THB, or are there graph classes that require stronger assumptions?

In this paper, we study the cryptographic power of THB. The main question that we ask is:

> *What are the minimal cryptographic assumptions required for THB for a given class of graphs?*

We focus on a minimal setting, with a small number of parties and a single, or few, semi-honest corruptions, which we denote by t-THB for t corruptions. This makes our lower bounds stronger; and, as we demonstrate, even this simple setting offers a rich multi-layered terrain, and provides insights and implications for more general settings (including THC).

Before proceeding to state our results, we note that prior THB protocols actually achieved the stronger property of *topology-hiding anonymous broadcast* (THAB), where the identity of the broadcaster remains hidden [11,12]. From the definitions of these primitives, we have that[4]

$$\mathsf{THC} \implies \mathsf{THAB} \implies \mathsf{THB}.$$

[2] That is, the *Quadratic Residuosity* assumption, the *Decisional Diffie–Hellman* assumption, and the *Learning With Errors* assumption, respectively.

[3] The lower bound of [3] holds for 4-party 2-secure THB with respect to a small class of 4-node graphs, namely a square, and a square with any of its edges removed.

[4] To see that THC $\Rightarrow$ THAB, observe that semi-honest anonymous broadcast can be realized using a secure sum, where the broadcaster inputs the message to be broadcast anonymously, and all other parties input 0.

Thus, all lower bounds for THB (such as the one from [4] and our own results) apply also for THAB and THC. As we will show, there are classes of graphs where THB is possible information theoretically, but THAB, and thus THC, requires strong cryptographic assumptions. Understanding for which topologies the reverse implications hold is addressed here in part, but the full answer remains an interesting open question.

1.1. *Our Results*

This work makes significant strides in mapping the landscape of THB, THAB, and THC in minimal settings, in the process developing new techniques that may be useful to achieve a full understanding of its complexities. As standard in the THC literature, we consider a synchronous setting, where the protocol proceeds in rounds.[5]

New Lower Bounds and Techniques

- *THB.* We explore which properties of graph topology are "hard" to hide, in the sense of requiring cryptographic assumptions to do so. We show that hiding any one of the properties of *direction*, *distance*, and/or *distance-of-neighbors* to the broadcaster is hard—while revealing all three but nothing else (in fact, only revealing distance-of-neighbors) can always be achieved information theocratically, using the trivial flooding protocol.
- *THAB.* We observe that t-THAB for any graph class containing a graph that is not $(t+1)$-connected[6] implies *key agreement*. We further show that hiding the *number of participants* in certain graph classes implies *infinitely often oblivious transfer*, even for 1-THAB.

Unconditional and KA-Based Upper Bounds

- *Unconditional.* We provide a construction of 1-THAB for *all 2-connected graphs*, whose complexity grows with the number of potential graphs in the class (in particular, it is efficient for constant-size graphs), which achieves *statistical* information-theoretic security.
- *Key Agreement.* Assuming the existence of *key agreement*, we achieve 1-THB for *all graphs*, and 1-THAB for all graphs of ≥ 3 nodes.

Corollaries and Conclusions

- *Dichotomy for* 1-THC *with* ≥ 3 *parties.* An interesting corollary of our results is a dichotomy for 1-THC with a fixed and known set of at least three parties[7] (i.e., where all graphs share the same vertex set): If all graphs in a class are 2-connected, the class supports information-theoretic 1-THC; otherwise, key agreement is necessary and sufficient for 1-THC.
- *Dichotomy for* 1-THAB *with* ≥ 3 *parties.* A similar result holds for 1-THAB for a dynamic set of parties (i.e., the vertex set of every graph is a *subset* of $[n]$) as long as each graph contains at least three nodes: If all graphs in a class

[5] LaVigne et al. [27] recently studied THC in a non-synchronous setting, demonstrating many barriers.

[6] A graph is k-connected if and only if every pair of nodes is connected by k *vertex-disjoint* paths.

[7] If the class of graphs contains a 2-path, then oblivious transfer is necessary for secure computation [24].

are 2-connected, the class supports information-theoretic 1-THAB; otherwise, key agreement is necessary and sufficient for 1-THAB.
- *Characterization of* 1-THB *for small graphs.* Our results introduce several new constructions and analysis techniques; as a demonstration of their wider applicability, we provide a characterization of the more complex case of 1-THB for all graph classes on four nodes or fewer. Note that the feasibility boundaries of 1-THB are more complex than 1-THAB since, as we show, certain lower bounds for 1-THAB do not apply to 1-THB.
- *THB without OT.* Our upper bounds constitute the first protocols using machinery "below" oblivious transfer,[8] aside from the specific graph class of cycles of fixed length (that was shown in [4]).

We next describe these results in more detail.

1.1.1. *Lower Bounds*

We begin by investigating the conditions under which THB and THAB for a graph class $\mathcal{G}$ *necessitate* cryptographic assumptions.

THB: **Hiding direction, distance, or distance-of-neighbors** Recall that restricting attention to a class of graphs $\mathcal{G}$ captures that a THB protocol hides *partial* information about a graph. For example, if all graphs in $\mathcal{G}$ have property P, then the THB protocol need not hide whether P is satisfied when providing indistinguishability within this class. Our question thus becomes: For which properties of a graph topology is it the case that hiding necessitates cryptography?

Consider as a baseline the trivial "flooding" protocol, which in general is *not* topology hiding. Parties flood the network: On receiving the broadcast message, a party forwards it to all neighbors from which it was not previously received. Indeed, this protocol reveals information; e.g., the round number in which a party first receives the message corresponds directly to its distance from the broadcaster. However, even for this simple protocol, the amount of information revealed is limited. The leakage can be quantified precisely: Each party learns exactly the distance from the broadcaster of each of its neighbors,[9] or *"distance-of-neighbors."* In particular, this includes the information of (a) *direction* of the broadcaster (i.e., which neighbors are on a shortest path to the broadcaster), and (b) *distance* to the broadcaster. Since the flooding protocol can be executed unconditionally for any graph class $\mathcal{G}$, it can only be some combination of this leaked distance-of-neighbors information for which hiding requires cryptography.

Examining the lower bound of [4], we observe that it constitutes an example where hiding the *direction* of the broadcaster from a given party necessitates key agreement (KA). This is embodied via the class of two graphs $\mathcal{G}_{\text{4-path}} = \{①\text{-}②\text{-}\underline{③}\text{-}④,\ ②\text{-}\underline{③}\text{-}④\text{-}①\}$ on a path, where party ③ is unaware whether the broadcasting party ① lies to its left

[8] Note that OT is strictly stronger than KA in terms of black-box reductions, since OT implies KA in a black-box way, but the converse does not hold [18].

[9] If the neighbor sends the message in the first round that the party learns it, then its distance is one less of the party's distance. If the neighbor sends after the party learned it, then its distance equals the party's distance. If the neighbor does not send, then its distance is one more than the party's distance.

or right. Indeed, broadcaster direction is central to their lower bound, where KA agents Alice and Bob emulate the THB parties ① and ④, respectively, and jointly emulate ③. Each flips a (private) coin to decide whether to also emulate ① on their corresponding side. The two parties can detect cases where both (or neither) party decided to emulate ①. In the remaining cases, both parties agree on which side the broadcaster appears: This will serve as the secret common key bit.

At a high level, the security of this KA protocol relies on the fact that the eavesdropper's view is essentially that of party ③—who, by topology hiding, cannot distinguish the relative direction of ①. Thus, one may naturally ask whether hiding the *direction* to the broadcaster captures the essence of the cryptographic power of THB.

Our first result shows that the direction to the broadcaster is not the complete answer. We present a class of graphs $\mathcal{G}_{\text{oriented-5-path}}$ for which any constant-round 1-secure THB implies *infinitely often key agreement*,[10] but for which the direction to the broadcaster is always known. Specifically, we consider the class of 5-path graphs where the broadcaster ① is always on the left,[11] i.e.,

$$\mathcal{G}_{\text{oriented-5-path}} = \left\{ ①\text{-}②\text{-}③\text{-}④\text{-}⑤,\ ①\text{-}⑤\text{-}②\text{-}③\text{-}④,\ ①\text{-}④\text{-}⑤\text{-}②\text{-}③,\ ①\text{-}③\text{-}④\text{-}⑤\text{-}② \right\}.$$

Because of this structure, the lower bound techniques of Ball et al. [4] do not apply. Proving a key-agreement implication for $\mathcal{G}_{\text{oriented-5-path}}$ requires a new, more subtle approach, which we discuss in Sect. 1.2. In particular, unlike [4], we must leverage the fact that topology hiding holds for *any* choice of corrupted party. For example, party ③ cannot distinguish between ①-②-③-④-⑤ and ①-⑤-②-③-④, and party ② cannot distinguish between ①-⑤-②-③-④ and ①-④-⑤-②-③.

Taking a broader view of this example, we observe that while the *direction* of the broadcaster is public for $\mathcal{G}_{\text{oriented-5-path}}$, the information to be hidden corresponds directly to the *distance* of the given parties to the broadcaster. One may thus once again wonder whether revealing both the *direction and distance* to the broadcaster dictates unconditional THB feasibility.

Our second result reveals that the answer is even more intricate. We demonstrate a class of graphs for which each party publicly knows *both its direction and distance* to the broadcaster, but for which 1-THB still implies key agreement.

Specifically, we consider the class $\mathcal{G}_{\text{triangle}}$ consisting of a triangle, with possibly one of its edges missing (see Fig. Fig. 1). Interestingly, this is a very basic communication pattern: If a party has two neighbors, it does not know if its neighbors are directly connected or not, but a party with one neighbor knows the entire topology. Notably, direction and distance from the broadcaster are both clearly identifiable to each party given just its neighbor set; the only information hidden from a party is *its neighbor's* distance to the broadcaster. We show that this is enough to imply KA (see Sect. 1.2 for details).

[10] An infinitely often key agreement guarantees correctness and security for infinitely many $\lambda \in \mathbb{N}$ (where λ stands for the security parameter).

[11] In particular, the "left/right" orientation can be deduced locally from each node's neighbor set.

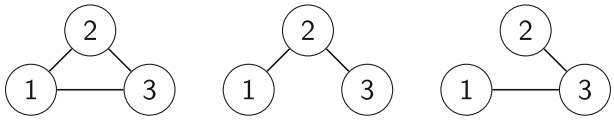


Fig. 1. The class $\mathcal{G}_{\mathsf{triangle}}$.

To summarize, for each strict subset of the properties that are leaked by the flooding protocol (namely *direction* and/or *distance* to the broadcaster) we demonstrate a graph class for which hiding only these properties implies public-key cryptographic assumptions. Complementarily, if all three properties (essentially, just the distance-of-neighbors) are known then one can use the flooding protocol to obtain THB information theoretically.

Theorem 1.1. *(THB* lower bounds, informal) We consider THB with one semi-honest corruption.

- 1-secure THB for the graph class $\mathcal{G}_{\mathsf{oriented\text{-}5\text{-}path}}$ of 5-path graphs for which the broadcasting party is always in the left-most direction (see above) implies infinitely often key agreement.
- 1-secure THB for the graph class $\mathcal{G}_{\mathsf{triangle}}$ (Fig. 1), for which the broadcasting party is always at a known distance *and* direction, implies key agreement.

In contrast, for any class $\mathcal{G}$ such that for every party the distance of each of its neighbors to the broadcaster is fixed and known across all graphs, there exists an unconditionally 1-secure THB protocol.

THAB: **Key Agreement and Beyond** We next turn to topology-hiding *anonymous* broadcast (THAB). As mentioned above, any lower bound for THB is also a lower bound for THAB; however, we show even stronger results for THAB.

The connection between anonymous communication and cryptographic hardness was previously studied by Ishai et al. [22]. They showed that in a communication network that provides *sender-anonymity* (under relatively strong adversarial observation), key agreement exists unconditionally; i.e., each pair of parties within the system can agree on a secret key. Our setting is slightly different, however, using the lower bound technique from [4] a similar observation can be made: Sender-anonymous communication over a path of three nodes implies the existence of standard Alice–Bob key agreement, where the eavesdropper can see which party sends which message.

This clear-cut impossibility of information-theoretic 1-THAB (in fact, 1-secure anonymous broadcast) on arbitrary incomplete networks stands in contrast to 1-THB, where the determination of when a graph class yields an implication to key agreement was demonstrably complex. Concretely, consider the following (singleton) class $\mathcal{G}_{\{1\text{-}2\text{-}3\}}$:

$$\mathcal{G}_{\{1\text{-}2\text{-}3\}} = \{①\text{-}②\text{-}③\}\,.$$

THB for this class is glaringly trivial (indeed, there is no information to hide because the topology is fixed); however, as discussed, 1-THAB on this class implies key agreement.

For completeness, in Sect. 5.1 we prove this implication as a direct corollary of the key-agreement lower bound of Ball et al. [4], where the "direction" of the broadcaster (either ① or ③) in this case is hidden from the intermediate party ② by anonymity.

At this moment, the reader may pause, ensnared in the underwhelming nature of the above class $\mathcal{G}_{\{1\text{-}2\text{-}3\}}$. However, by a standard player-partitioning argument ("projecting" a larger graph down onto the 3-path), the above result yields a much broader statement.

Proposition 1.2. (THAB lower bound 1, informal, [4,22]) *Let $\mathcal{G}$ be a class of graphs that contains a graph with at least $(t+2)$ nodes that is not $(t+1)$-connected. Then, t-secure THAB for $\mathcal{G}$ implies KA.*

In our final lower bound result, we demonstrate an even more extreme form of separation between THB and THAB. We consider the graph class $\mathcal{G}_{2\text{-vs-}3}$ that consists of all possible 2-path and 3-path graphs over three parties, i.e.,

$$\mathcal{G}_{2\text{-vs-}3} = \left\{①\text{-}②,\ ①\text{-}③,\ ②\text{-}③,\ ①\text{-}②\text{-}③,\ ②\text{-}③\text{-}①,\ ③\text{-}①\text{-}②\right\}.$$

In this class, for example, if ① is connected only to ②, then it does not know whether ② has a second neighbor (necessarily ③) or not. It is easy to see that 1-secure THB exists unconditionally for this class (by the flooding protocol); however, we show that 1-secure THAB implies *infinitely often oblivious transfer*.[12] We emphasize that as opposed to other classes of graphs discussed thus far, the "hardness" of the class $\mathcal{G}_{2\text{-vs-}3}$ is based on hiding the *number of nodes* participating in the protocol. We refer the reader to Sects. 1.2 and 5.2 for further details on the lower bound.

Overall, we obtain the following theorem.

Theorem 1.3. (THAB lower bound 2, informal) 1-*secure THAB for $\mathcal{G}_{2\text{-vs-}3}$ implies infinitely often OT.*

We remark that these results separate THB from THAB for very simple graph classes, where THAB requires computational assumptions whereas unconditional THB exists via the trivial flooding protocol. Later, in Sect. 1.1.2 we will show a more interesting separation via the "butterfly" graph, where the existence of information-theoretic THB itself is non-trivial.

1.1.2. *Upper Bounds*

Before stating our results, we recall the state of the art for semi-honest THB and THAB with one corruption. Assuming oblivious transfer (OT), 1-THAB can be obtained for all graphs following the construction approach of Moran et al. [29].[13] Without assum-

[12] An infinitely often OT protocol guarantees correctness and security for infinitely many $\lambda \in \mathbb{N}$ (where λ stands for the security parameter).

[13] The result of [29] was limited to graphs of small diameter to allow an arbitrary number of corruptions. With a single corruption, the same construction can support all graphs.

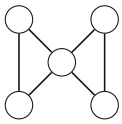


Fig. 2. The butterfly graph.

ing OT, the only previously known non-trivial[14] construction of THB or THAB is the information-theoretic 1-THAB for the specific graph class of cycles on a known number of nodes in [4].

We consider three settings of upper bounds: (1) with *information-theoretic* security, (2) assuming only *key agreement*, and (3) converting generically from THB to THAB.

Information-theoretic security First, we consider protocols for achieving 1-THAB (and THB) in the *information-theoretic* setting, without cryptographic assumptions. Recall that the lower bound in Proposition 1.2 rules out the possibility of 1-THAB for any graph class containing a graph that is not 2-connected. We show that conversely, if a class of graphs $\mathcal{G}$ contains only 2-connected graphs, then 1-THAB for $\mathcal{G}$ *is* feasible.

The protocol's communication grows polynomially in the *size* of the class $\mathcal{G}$, and its computation grows polynomially in the size of $\mathcal{G}$ and exponentially in the *maximal degree* of any $G \in \mathcal{G}$. However, our results are meaningful despite this caveat: First, the protocol is efficient when considering a constant number of parties (or appropriate graph classes of polynomial size). Second, since the protocol remains secure against computationally unbounded adversaries, it is still meaningful to consider protocols that are inefficient in the class.

Theorem 1.4. (1-IT-THAB for 2-connected, informal) *Let $\mathcal{G}$ be a class containing only 2-connected graphs. Then, there exists a statistical information-theoretic 1-THAB for $\mathcal{G}$ whose communication complexity is polynomial in the size of $\mathcal{G}$, and whose computation complexity is polynomial in the size of $\mathcal{G}$ and exponential in the maximal degree of $\mathcal{G}$.*

Combining Proposition 1.2 and Theorem 1.4 gives a characterization for information-theoretic 1-THAB: Namely, a protocol exists if and only if all graphs in the class are 2-connected (with the exception of the trivial class containing only the 2-path). For the case of 1-THB, such dichotomy does not hold and, as we show, there exist graph classes with 1-connected graphs that still admit information-theoretic 1-THB protocols.

Remark 1.5. (1-IT-THB for $\mathcal{G}_{\text{butterfly}}$) Consider the 5-node, 1-connected butterfly graph (Fig. 2), and let $\mathcal{G}_{\text{butterfly}}$ contain all permutations of the nodes on the graph (where parties' positions are permuted). In Sect. 6.2, we show that although the simple flooding protocol does *not* directly hide topology, there exists a (perfectly secure) information-theoretic 1-THB protocol for $\mathcal{G}_{\text{butterfly}}$.

[14] THB exists trivially for any graph class in which each party's neighborhood uniquely identifies the graph topology.

Upper bounds from KA Recall that from the lower bounds presented above (see Sect. 1.1.1), key agreement is a necessary assumption for 1-THB and 1-THAB for many classes of graphs. This begs the question of when key agreement is also a *sufficient* assumption for 1-THB and 1-THAB. We show that assuming key agreement there exist 1-secure THB for all graphs, and 1-secure THAB for all graphs containing at least 3 nodes.

Theorem 1.6. (1-THAB and 1-THB from KA, informal)

- *Let $\mathcal{G}$ be a class consisting of graphs with at least three nodes. Assuming key agreement, there exists 1-THAB for $\mathcal{G}$.*
- *Let $\mathcal{G}$ be a class of graphs. Assuming key agreement, there exists 1-THB for $\mathcal{G}$.*

We note that in the first item of Theorem 1.6, removing the restriction of at least three nodes would require bypassing black-box separation results, due to Theorem 1.3 that asserts the necessity of (infinitely often) OT for the class $\mathcal{G}_{\text{2-vs-3}}$. On the other hand, by [29], assuming OT there exists 1-THAB for all graphs, essentially closing the gap in this regime.

THC dichotomy Upon closer inspection, we observe that our upper bounds—both the information-theoretic protocols for 2-connected graphs, as well as the results from KA above—give something even stronger than 1-THAB: They give topology-hiding *secure message transmission*, i.e., emulating pairwise secure point-to-point channels. In this case, assuming that the number of parties is fixed and known across all graphs, we can run the semi-honest "BGW" protocol [8], which only requires pairwise secure channels and works for an honest majority. Thus, together with our lower bounds, we arrive at the following dichotomy for 1-THC:

Corollary 1.7. (1-secure THC dichotomy, informal) *Consider a class of graphs $\mathcal{G}$ on $n \geq 3$ nodes. Then, the following hold regarding existence of THC for $\mathcal{G}$ secure against 1 semi-honest corruption:*

- *If all graphs $G \in \mathcal{G}$ are 2-connected, then there exists a statistically information-theocratically secure, 1-THC protocol for $\mathcal{G}$, whose communication is polynomial in the size of $\mathcal{G}$ and whose computation is polynomial in the size of $\mathcal{G}$ and exponential in the maximal degree of $\mathcal{G}$.*
- *If there exists $G \in \mathcal{G}$ that is* not *2-connected, then KA is necessary and sufficient for 1-secure THC for $\mathcal{G}$.*

Generically converting THB **to** THAB Our results have demonstrated a number of non-trivial separations between THB and THAB, identifying classes of t-connected graphs and computational assumptions which admit t-THB protocols but provably cannot obtain t-THAB. This includes, for example, $\mathcal{G}_{\{1\text{-}2\text{-}3\}}$ and $\mathcal{G}_{\text{butterfly}}$ for information theoretic vs. key agreement, as well as $\mathcal{G}_{\text{2-vs-3}}$ for information theoretic vs. oblivious transfer.

Finally, we show that graph connectivity is, indeed, a critical property for determining the relation between THB and THAB on a class of graphs. Specifically, we show that $(t+1)$-connectivity is a sufficient condition for *equivalence* of the two notions against t corruptions.

Table 1. Summary of upper and lower bound results.

	1-THB		1-THAB	
	Sufficient	Necessary	Sufficient	Necessary
IT	$\mathcal{G}_{\text{cycle}}$ [4] $\mathcal{G}_{\text{butterfly}}$ (Remark 1.5) 2-connected (Thm 1.4)	–	2-connected (Thm 1.4)	–
KA	All graphs (Thm 1.6)	$\mathcal{G}_{\text{4-path}}$ [4] $\mathcal{G}_{\text{oriented-5-path}}$ (Thm 1.1) $\mathcal{G}_{\text{triangle}}$ (Thm 1.1)	All graphs ($\geq$ 3 nodes) (Thm 1.6)	Not 2-connected ($\geq$ 3 nodes) (Prop 1.2)
OT	All graphs [29]	–	All graphs[29]	$\mathcal{G}_{\text{2-vs-3}}$ (Thm 1.3)

Theorem 1.8. (t-THB $\Rightarrow$ t-THAB given $(t+1)$-connectivity, informal) *Let $n \in \mathbb{N}$, and let $\mathcal{G}$ be a class consisting of $(t+1)$-connected graphs over n nodes. If there exists t-THB for $\mathcal{G}$, then there exists t-THAB for $\mathcal{G}$.*

Our reduction builds upon the "Dining Cryptographers" approach for anonymous broadcast due to Chaum [12]. Recall in THAB there exists a unique broadcaster who wishes to convey its input bit $x \in \{0, 1\}$ to all parties without revealing its identity (or the topology). To do so, each party first additively secret shares its input—defined to be 0 for any non-broadcaster—across its neighbors, locally sums all received shares to $s_i \in \{0, 1\}$, and then acts as broadcaster within the underlying (non-anonymous) THB with input value s_i. After this phase, all parties receive the vector of shares $(s_1, \ldots, s_n)$, which can be summed to yield the original input x. It was shown by [12] that if the graph is $(t+1)$-vertex connected (so as to ensure that the adversary cannot corrupt a vertex cut), then the protocol is anonymous. We observe that the protocol further preserves the *topology hiding* of the underlying THB protocol. Indeed, given $(t+1)$-connectivity, the vector of broadcasted shares $(s_1, \ldots, s_n)$ will be *uniform* conditioned on the necessary sum, independent of the graph structure.

1.1.3. *Summary and Characterization of Graphs with up to Four Nodes*

We summarize our combined contributions in Table 1, together with relevant prior results.

In addition, and as a demonstration of the power and applicability of the techniques developed, in Sect. 8.2 we provide a characterization of the feasibility of 1-THB and 1-THAB for all graph classes on up to 4 parties. The characterization uses a partition of the 4-node graphs into multiple classes, each of which can be handled by a separate technique.

1.2. *Technical Overview*

We next highlight a selection of our new analysis and protocol-construction techniques, described in Sects. 1.2.1 to 1.2.4. We will describe two analysis techniques that are used in our lower bounds: "*phantom jump*" and "*artificial over-extension*." In addition, we will describe two protocol-design techniques that are used in our upper bounds: "*censored brute force*" and "*dead-end channels*."

1.2.1. *Analysis Technique: "Phantom Jump"*

The "phantom jump" technique is a means for proving indistinguishability of the transcript of messages sent across a given edge ①-② in THB executions on two different graphs, via a sequence of intermediate indistinguishability steps, each appealing to THB security for a different graph pair. In applications, the initial and final graphs will have a party "jump" from one side of the graph to the other, which will be used within the key-agreement implication analysis.

This technique is used within some of our key-agreement lower bounds. We focus here on a specific example for the class $\mathcal{G}_{\mathsf{triangle}}$ (of a triangle graph with a potential edge missing). We point the reader to more elaborate examples on 4-node graph classes in Sect. 8.2.

We start by recalling how a 1-THB protocol π for $\mathcal{G}_{4\text{-path}} = \{$①-②-③-④, ②-③-④-①$\}$ was used to construct key agreement in [4]. The idea is for Alice to choose two long random strings m_1 and m_2 and send them to Bob in the clear. Next, Alice and Bob continue in phases as follows:

- In each phase, Alice and Bob locally toss coins A and B, respectively.
- They proceed to run two executions of π in which Alice always emulates ② and ③ and Bob emulates ④. In addition, if $A = 0$, then Alice emulates ① (as a neighbor of ②) broadcasting m_1 in the first run; otherwise, she emulates ① broadcasting m_2 in the second run. Similarly, if $B = 1$, then Bob emulates ① (as a neighbor of ④) broadcasting m_1 in the first run; otherwise, he emulates ① broadcasting m_2 in the second run.
- If parties ② and ④ output m_1 in the first run and m_2 in the second, Alice and Bob output their bits A and B, respectively; otherwise, they execute another phase.

Clearly, if $A = B$ in some iteration, 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 ① in the first run and who in the second. If $A \neq B$, then in at least one of the runs nobody emulates the broadcaster ①, so with overwhelming probability Alice and Bob will detect this case.

We proceed to adjust this argument to $\mathcal{G}_{\mathsf{triangle}}$. Constructing the KA protocol is rather similar, where Alice always emulates ② and Bob always emulates ③, and each party emulates the broadcaster ① based on their local coins A and B (see Fig. 3). Proving correctness follows exactly as in the argument from [4]; however, proving security is more involved. Indeed, in $\mathcal{G}_{4\text{-path}}$ the view of Eve corresponds to a partial view of the intermediate node ③ who is never a neighbor of ① and, so by the security of π, never

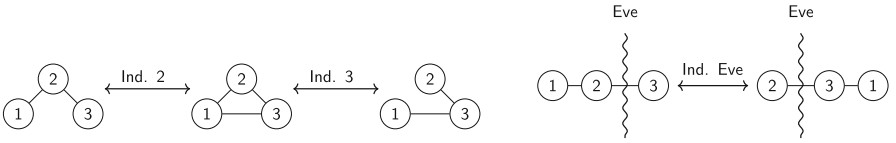


Fig. 3. 1-THB on $\mathcal{G}_{\text{triangle}}$ implies KA.

learns its direction to ①. When considering $\mathcal{G}_{\text{triangle}}$, the view of Eve consists of the communication between ② and ③, and one of them must be a neighbor of ①.

This is where the new phantom-jump technique comes into play. As opposed to [4], we do not construct a reduction from Eve to the security of the THB protocol; rather, we use a direct indistinguishability argument. Notice that the KA construction required the use of only two graphs ①-②-③ and ②-③-①. The third graph (the triangle) is needed for the proof.

As depicted in Fig. 3, the view of Eve consists of the communication between ② and ③. By THB security ② cannot distinguish between the 3-path ①-②-③ and the complete triangle; in particular, the distribution of the messages on the channel between ② and ③ is indistinguishable in both cases. Similarly, by THB security ③ cannot distinguish between the 3-path ②-③-① and the complete triangle; in particular, the distribution of the messages on the channel between ② and ③ is indistinguishable in both cases. By a simple hybrid argument, it follows that the messages between ② and ③ are indistinguishable when communicating in ①-②-③ and when communicating in ②-③-①. It follows that the distinguishing advantage of Eve is negligible.

1.2.2. *Analysis Technique: "Artificial Over-Extension"*

The artificial over-extension technique is used for proving two of our lower bounds: first, Theorem 1.1 where 1-THB for $\mathcal{G}_{\text{oriented-5-path}}$ is used to construct infinitely often KA (see also Sect. 4.1) and second, Theorem 1.3 where 1-THAB for $\mathcal{G}_{\text{2-vs-3}}$ is used to construct infinitely often OT (see Sect. 5.2). In the following, we focus on the latter.

Recall that in the class $\mathcal{G}_{\text{2-vs-3}}$ a party (say ①) that has a single neighbor (say ②) does not know whether ② has another neighbor, ③, or not. This uncertainty is the source of the cryptographic hardness we present; indeed, if the parties know that an honest majority cannot be assumed (i.e., there are only two parties), then 1-THAB is trivial, whereas if an honest majority can be assumed (i.e., there are three parties), then 1-THAB exists assuming KA (by Theorem 1.6). We also note that without anonymity, 1-THB trivially exists in $\mathcal{G}_{\text{2-vs-3}}$ (via the flooding protocol).

We start with an intermediate goal, that of constructing oblivious transfer from a *two-round* 1-THAB protocol π for the graph class $\mathcal{G}_{\text{2-vs-3}}$,[15] and later explain how the novel "artificial over-extension" technique allows us to extend this construction to arbitrary constant-round protocols. Note that using this technique we can only construct *infinitely*

[15] In fact, for this step we will only need for the subclass {②-③, ①-②-③, ②-③-①} $\subseteq \mathcal{G}_{\text{2-vs-3}}$.

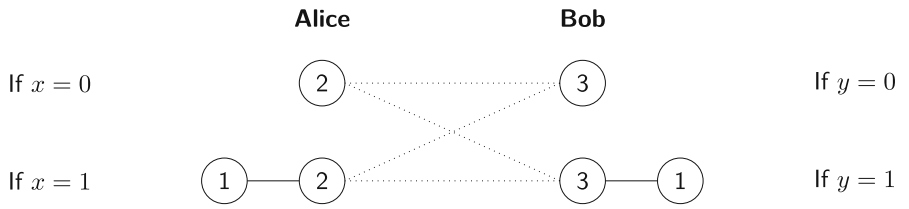


Fig. 4. Boolean AND from two-round 1-THAB for {②-③, ①-②-③, ②-③-①}.

often OT, and extending the implication to a full-blown OT is left as an interesting open question.

OT from two-round 1-THAB Given a two-round 1-THAB protocol π, we construct a secure two-party protocol for Boolean AND (which in turn implies OT, by Kilian [24]).

In the protocol, Alice and Bob will emulate an execution of the 1-THAB protocol on a path, where each extends the length of the path (by emulating an extra party) if their input is 1. More concretely, Alice simulates a single node ② if her input is 0, and two nodes ①-② if her input is 1. Similarly, Bob simulates a single node ③ if his input is 0 and two nodes ③-① if his input is 1 (see Fig. 4). Next, Alice chooses a random message $m \leftarrow \{0,1\}^\lambda$, sends it to Bob in the clear, and initiates an execution of π on message m on the graph with her left-most node (either ② or ①) as broadcaster. At the conclusion of π, Bob identifies whether his right-most emulated party (either ③ or ①) correctly outputs m. If so, then Bob outputs 0; if not, he outputs 1.

We show that this protocol securely computes AND of Alice and Bob's inputs.

- For *security*, we exploit the fact that the only case where there is something to hide (namely if a party holds input 0) is where the respective party has control over just a *single* node in π. Security therefore follows from the fact that π is a THAB protocol with security against one corruption. For example, the views of a corrupt Alice emulating ② within executions over graphs ②-③ (Bob has input 0) and ②-③-① (Bob has input 1) are indistinguishable.
  Note here that for security it is crucial that π is an *anonymous* broadcast protocol, because in case $x = 0$, Alice broadcasts from node ② and in case $x = 1$ from node ①. In fact, as noted above, 1-THB can be achieved trivially on $\mathcal{G}_{\text{2-vs-3}}$.
- For *correctness*, first note that when at least one party has input 0, the corresponding graph is an element of $\{②\text{-}③, ①\text{-}②\text{-}③, ②\text{-}③\text{-}①\} \subseteq \mathcal{G}_{\text{2-vs-3}}$, in which case proper delivery of m to Bob's right-most node is guaranteed by correctness of π. On the other hand, when $x \wedge y = 1$ (i.e., both Alice and Bob emulate node ①), the parties effectively emulate π over an "invalid" length-4 path ①-②-③-①. While behavior of π within such execution is unclear, since π runs in only 2 rounds, the message m simply cannot reach the right-most node emulated by Bob at distance 3. Thus, Bob will correctly output 1 with overwhelming probability.

Infinitely often OT from constant-round 1-THAB Note that correctness of the construction above crucially relies on efficiently detecting an execution of π on the graph ①-②-③-①, leveraging its insufficient round complexity. However, this argument is

no longer guaranteed when π completes in more than two rounds. This is where the "artificial over-extension" technique comes into play.

The insight is that *either* an execution of π on graph ①-②-③-① can indeed be efficiently detected, in which case the protocol above extends (and we are done), *or* π actually provides a stronger form of topology hiding that we can further leverage. Namely, if neither Alice nor Bob can identify when π is executed on ①-②-③-① as opposed to a legal graph, then in particular π provides 1-THAB for the larger graph class $\mathcal{G}'_{\text{2-vs-3}} := \mathcal{G}_{\text{2-vs-3}} \cup \{①\text{-}②\text{-}③\text{-}①\}$.

In the latter case, we can take a similar approach to above, but with the graph class $\{①\text{-}②, ③\text{-}①\text{-}②, ①\text{-}②\text{-}③\text{-}①\} \subseteq \mathcal{G}'_{\text{2-vs-3}}$, with Alice emulating ① or ③-①, and Bob emulating ② or ②-③-①, and hope that π identifiably breaks down on the "over-extended" path ③-①-②-③-① of length 5. If not, this argument repeats, until—via this artificial over-extension technique—ultimately we reach a graph class $\mathcal{G}$ for which:

- π *is* 1-THAB on $\mathcal{G}$, including $\{(u_2^*)\text{-}(u_3^*), (u_1^*)\text{-}(u_2^*)\text{-}(u_3^*), (u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*\} \subseteq \mathcal{G}$
- π is *not* 1-THAB on $\mathcal{G} \cup \{(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*\}$

where $(u_1^*), (u_2^*), (u_3^*) \in \{①, ②, ③\}$, and $\boldsymbol{P}^*$ is a path of length upper bounded by the round complexity of π. Once we do, then the original secure-AND protocol approach will succeed, modulo some differences described below, with Alice emulating (u_2^*) or $(u_1^*)\text{-}(u_2^*)$, and Bob emulating (u_3^*) or $(u_3^*)\text{-}\boldsymbol{P}^*$.

To argue that eventually we find a path $(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*$ for which π identifiably breaks down, we again appeal to its bounded round complexity, i.e., π must fail identifiably (with probability 1) once the length of the path exceeds the round complexity. The limitation of constant rounds is a subtle side effect of the corresponding hybrid argument, to argue that there must be some step where we jump sufficiently from indistinguishable to efficiently identifiable.

Consider the resulting secure-AND protocol, once an appropriate $(u_1^*), (u_2^*), (u_3^*), \boldsymbol{P}^*$ are found. The only modification from the simpler two-round version is how to detect the (over-extended) case $x \wedge y = 1$. When π was two rounds, identifying this event was immediate: Bob's right-most party simply will not receive the delivered message. Here, this is not necessarily the case, as the identifiable "breakdown" of π may occur before the length of $(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*$ exceeds π's round complexity. Thus, instead, the parties will run the distinguisher that—roughly speaking—exists from the fact that π is not 1-THAB on $\mathcal{G} \cup \{(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*\}$. This is the reason why our final protocol guarantees correctness only for infinitely many $\lambda \in \mathbb{N}$: All we can say is that *either* the protocol π is 1-THAB on $\mathcal{G} \cup \{(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*\}$ and we can continue with the extension argument, *or* π is not 1-THAB, i.e., there exists a distinguisher that efficiently detects the "too-long" path $(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*$ with noticeable advantage for *infinitely many* $\lambda \in \mathbb{N}$.

Finally, in order to boost correctness toward negligible correctness error (for infinitely many λ), Alice and Bob simply run the protocol π and the distinguisher sufficiently many times, each time on input of a fresh message m, and take a corresponding majority vote.

1.2.3. *Protocol Design: "Censored Brute Force"*

The Censored Brute Force technique enables constructing unconditionally secure pairwise channels between each pair of parties which further guarantees sender anonymity. Such anonymous and private channels are used for proving Theorem 1.4 and Corollary 1.7, by constructing 1-THAB and 1-THC with information-theoretic security for any class $\mathcal{G}$ for which all graphs are 2-connected (see Sect. 6.1 for more details). Recall that the communication complexity of the resulting protocols is polynomial in the size of $\mathcal{G}$ (i.e., the number of graphs in the class, which could be superpolynomial in the number of parties) and the computation complexity is polynomial in the size of $\mathcal{G}$ and exponential in the maximal degree of $\mathcal{G}$.

The high-level idea is twofold: For any *single* 2-connected graph G, we show how to unconditionally perform sender-anonymous point-to-point communication on G with an ability for any party to (anonymously) "censor" the communication, i.e., yielding delivery of random garbage instead of the intended message. Then, for a given class of 2-connected graphs $\mathcal{G}$, the parties will simultaneously execute (in parallel) a separate anonymous-communication protocol for *every graph* $G \in \mathcal{G}$; for each such G-execution, a party will *censor* the execution if its true neighborhood is inconsistent with its neighborhood in G. As such, the only protocol execution that remains uncensored will be the one corresponding to the *correct* execution graph G (and the identity of which G this corresponds to can be made hidden to the receiving party). We elaborate on these two aims below.

Communicating anonymously in a 2-connected graph More concretely, suppose we have a *single* 2-connected graph G on vertex set $[n]$, and fix some designated source and target nodes $\sigma \neq \tau \in [n]$. Let $H_{\sigma\tau}$ denote an arbitrary $\sigma\tau$-*orientation* of G,[16] i.e., a directed acyclic graph with unique sink τ and unique source σ formed by assigning a direction to each edge in G. Moreover, label all nodes $1, 2, \ldots, n$ according to a topologically consistent ordering of $H_{\sigma\tau}$ (beginning with σ and ending at τ). We consider the numbering/orientation of any graph G to be a public parameter, computed according to some deterministic procedure (see Proposition 6.3).

Now, suppose node u wishes to send a message m to the target node τ anonymously and securely on the graph G. In the first round, the source σ (i.e., the node labeled 1) prepares additive shares of 0 (or of m if $\sigma = u$) for each of its outgoing edges in $H_{\sigma\tau}$.

In round 2, the source σ sends the corresponding share to its neighbor node labeled 2, who then prepares secret shares of what it received ($+m$ if it is u) for each outgoing edge. More generally, in round $i < n$ all nodes with an edge to the i^{th} node send their shares to the i^{th} node. The i^{th} node, having received shares on all incoming edges, then sums up what it receives (adds m if it is u) and prepares additive shares of the result for each of its outgoing edges. In round n, all nodes with edges to τ (the target node) send their shares to τ and τ outputs their summation.

Correctness follows from the homomorphic properties of additive secret sharing. To see why this protocol is secure (namely that it hides u and m), note that the 2-connectivity of G implies that there are at least 2 vertex-disjoint $\sigma\tau$-paths in $H_{\sigma\tau}$. Thus, the messages

[16] The standard notation in the literature is st-orientation; to avoid confusion with the notation t that stands for the corruption threshold, we use $\sigma\tau$-orientation instead.

any intermediate party (corresponding to $2, \ldots, n-1$) receives are uniformly random because that node is in some sense always missing at least one share (corresponding to a disjoint $\sigma\tau$-path); the source σ does not receive anything at all, and the view of the target τ is simply a random sharing of its output m.

This protocol enjoys some other useful properties. Most notably, any non-sink node can covertly "censor" communication by simply preparing (and sending) shares of a uniformly random message, instead of preparing shares corresponding to what they received (as per the protocol). The view of every other party is identically distributed, with the exception of τ who now receives secret shares of a uniformly random message in the final round.

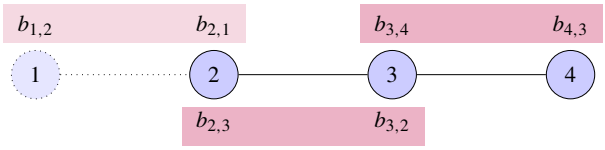
Compiling to hide topology Now, let $\mathcal{G}$ be a class of 2-connected graphs on vertex set $[n]$. Loosely speaking, the parties will simulate the above protocol for every possible graph in $\mathcal{G}$ simultaneously. Each node will covertly censor every protocol corresponding to a graph that is *not* locally consistent with their local neighborhood (sending random messages at the appropriate times). As a result, exactly one protocol (corresponding to the "real" graph) will give the correct output message and all others will give uniformly random output.

To be slightly more concrete, all nodes will execute the protocol above for each graph in the class in parallel. To keep track of which message is which, for every node but τ we will label the messages with the graph/protocol that the message corresponds to. If an edge is missing from the real graph, but present in a graph corresponding to one of the simulated protocols, the corresponding message cannot be sent. However, the receiving node knows not to expect a message either. From this and the uniformly random nature of non-terminal messages in the above protocol, nothing is leaked locally by labeling the simulations. However, sending labeled messages to τ would clearly identify the "real" topology. So instead, all parties will send all final protocol messages in randomly permuted order. To enable τ identifying the real output, the sender will append a long checksum to the message. The target τ will try all message combinations (this is the reason for the exponential dependency in the maximal degree) and output the unique one with a correct checksum (or abort if more than one message has a valid checksum).

1.2.4. *Protocol Design Technique: "Dead-End Channels"*

The Dead-End Channels technique is used for proving Theorem 1.6 (see also Sect. 7), i.e., to obtain 1-THAB for all graphs of at least three nodes (and 1-THB for all graphs), assuming existence of *key agreement*. Recall that before the present work, such results were only known assuming oblivious transfer [29].

The high-level idea of our 1-THAB protocol, as in Moran et al. [29], is to broadcast the message via *flooding*, but in a way that hides from the parties at which round they received the broadcast message. This can be achieved by passing the message between *virtual parties*, each consisting of two real parties that hold additive secret shares of the message (depicted, e.g., as pink bars for each neighboring pair of parties below). Only in the final round will the parties exchange their secret shares and recover the message.



The challenge thus becomes passing the messages between virtual parties. In [29], this is solved by using oblivious transfer (OT) to run an MPC protocol realizing the virtual party, and allowing every adjacent pair of virtual parties to securely compute the *OR* of their messages.

In our setting, we do not have the ability to perform secure computations pairwise between parties without OT. Instead, we leverage the fact that given at least three nodes we are guaranteed an honest majority and can therefore (once the parties establish secure channels using the key agreement protocol) build on techniques from information-theoretic secure computation to appropriately pass along the message.

However, this itself is not so straightforward. For example, in the image above, the neighboring parties ②-③-④ would wish to jointly emulate a three-party secure computation to perform the secure transfer from ②-③ to ③-④. But, the issue is that parties cannot reveal whether they truly have neighbors with which to jointly compute: For example, party ② above must then emulate a nonexistent neighbor ① to hide its true degree. Thus, grouping parties in three, including possibly a simulated neighbor, would allow the adversary to gain control over a majority. (On the other hand, building on secure computation including four our more neighbored parties, the same party could appear several times in the protocol and therefore potentially learn about the connectivity of its neighbors.)

Our approach builds on the following idea: We will give one party within each group of three the role of a *dealer* to deal *OT correlations*, which can be used to establish a secure OT channel between two other parties. This alone is not sufficient, as one of the parties could be simulated by the dealer (in the case that the dealer has degree one), and therefore allows the dealer to gain full control over the OT channel, and in particular learn the honest parties' inputs. To prevent this, we observe that—again using OT correlations—one can establish *dead-end channels* (i.e., information sent via such a channel cannot be read by anyone apart from the sender) if and only if the receiver is a simulated party. Therefore, even if the dealer simulates one of the parties, it does not learn anything about the honest parties' inputs. Note that it is crucial that dead-end channels are indistinguishable from secure channels from the view of the sender. Further, a key observation is that using OT correlations to establish dead-end channels does not leak anything about the topology, even if the dealer of the OT correlations has degree one. This is the case, because the only thing the dealer could potentially learn from the other party is whether its degree is one—but if the dealer has degree one it already knows that the degree of its neighbor must be at least two (as we are guaranteed a connected graph with a strict honest majority).

1.2.5. *Organization of the Paper*

In Sect. 2, we provide the necessary definitions and preliminaries. In Sects. 4 and 5, we present our THB and THAB lower bounds, respectively. In Sects. 6 and 7, we present

our information-theoretic and KA-based upper bounds. In Sect. 8, we present corollaries and implications of our techniques as well as a characterization of 1-THB for graphs with at most four nodes.

2. Preliminaries

Notations For $n \in \mathbb{N}$ let $[n] = \{1, \ldots, n\}$. In our protocols, we sometimes denote by n an upper bound on the number of participating parties and by t an upper bound on the number of corrupted parties. The security parameter is denoted by λ.

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 $\{v, w\}$ of distinct vertices. A graph is *directed* if its edges are instead ordered pairs (v, w) of distinct vertices. An *oriented* graph is a directed graph having no symmetric pair of directed edges, and an *orientation* of an undirected graph is an assignation of a direction to each of its edges so as to make it oriented. A graph is *k-connected* if it has more than k vertices and remains connected whenever fewer than k vertices are removed. A graph class $\mathcal{G}$ is k-connected if every graph $G \in \mathcal{G}$ is k-connected. Throughout this paper, we only consider *connected* graphs, even if we do not systematically make this explicit. 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 $\mathcal{N}_G[v] := \mathcal{N}_G(v) \cup \{v\}$.

2.1. *Topology-Hiding Computation (THC)*

Following [29], we consider two definitions of topology-hiding computation. Our positive results (protocol constructions) in Sects. 6 and 7 are defined with respect to the stronger simulation-based definition, while our lower bounds in Sects. 4 and 5 are given with respect to the weaker indistinguishability-based definition.

UC framework The simulation-based definition is defined in the UC framework of Canetti [9]; we present an informal overview of the model in Appendix A. Unless stated otherwise, we will consider computationally unbounded, static, and semi-honest adversaries and environments.

2.1.1. *Simulation-Based THC.*

We recall the definition of simulation-based topology-hiding computation from [4,29]. The real-world protocol is defined in a model where all communication is transmitted via the functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ (described in Fig. 5). The functionality is parameterized 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 v in the graph.

Initially, before the protocol begins, $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ 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,

The functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$

The functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ is parametrized by a family of graphs $\mathcal{G}$; let n denote the maximal number of nodes in $G \in \mathcal{G}$. The functionality proceeds with a special graph party $\mathsf{P}_{\mathsf{graph}}$ and with a subset of the parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ (to be defined by the graph received from $\mathsf{P}_{\mathsf{graph}}$) 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.

Fig. 5. The communication graph functionality.

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.

Note that if all the graphs in $\mathcal{G}$ have exactly n nodes, then the exact number of participants is known to all and need not be kept hidden. In this case, defining the ideal functionality and constructing protocols become a simpler task. However, if there exist graphs in $\mathcal{G}$ that contain a *different* number of nodes, then the model must support functionalities and protocols that only know an *upper bound* on the number of participants. In the latter case, the actual number of participating parties must be kept hidden.

Given a class of graphs $\mathcal{G}$ with an upper bound n on the number of 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), where any subset of them may be activated with (potentially empty) inputs. Only the parties that have been activated participate in the protocol, send messages to one another (via $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$), and produce output.

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 $\mathcal{N}_G(v)$

The 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: Record the input message $m \in \{0,1\}$ sent by the broadcaster P_i.

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.

Fig. 6. The broadcast functionality.

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

Broadcast and anonymous broadcast In this work, we will focus on topology-hiding computation of two central functionalities. The first is the *broadcast* functionality (see Fig. 6), where a designated and publicly known party, named *the broadcaster*, starts with an input value m. Our broadcast functionality guarantees that every party that is connected to the broadcaster in the communication graph receives the message m as output. In this paper, we assume the communication graphs are always connected. However, the broadcaster may not be participating, in which case it is represented as a degree-0 node in the communication graph. (And all the participating nodes are in a separate connected component.)

Parties that are not connected to the broadcaster receive a message that is supplied by the adversary. (We can consider stronger versions of broadcast, but this simplifies the proofs.)

We denote the broadcast functionality where the broadcaster is P_i by $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_i)$.

Definition 2.2. (t-THB) Let $\mathcal{G}$ be a family of graphs, and let t be an integer. A protocol π is a t-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.

The second functionality is the *anonymous broadcast* (see Fig. 7). This functionality is similar to broadcast with the exception that the broadcaster is not known and its identity is kept hidden even after the computation completes. Namely, the environment will activate exactly one of the parties with an input value, informing this party that it is the broadcaster. We denote the anonymous broadcast functionality $\mathcal{F}_{\mathsf{anon\text{-}bc}}$.

The functionality $\mathcal{F}_{\text{anon-bc}}$

The anonymous-broadcast functionality $\mathcal{F}_{\text{anon-bc}}$ proceeds as follows.

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

Input: Upon receiving an input message $m \in \{0,1\}$ from one of the parties P_i, record it.

Output: If exactly one input message m from party P_i was received, 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.

If more than one input was received, send G and all received inputs to the adversary, and for every party in G, the output delivered to that party is supplied by the adversary (i.e., there is no security guarantee if more than one input was received.)

Fig. 7. The anonymous-broadcast functionality.

Definition 2.3. (t-THAB) Let $\mathcal{G}$ be a family of graphs, and let t be an integer. A protocol π is a t-THAB protocol with respect to $\mathcal{G}$ if π securely realizes $\mathcal{F}_{\text{anon-bc}}$ in a topology-hiding manner with respect to $\mathcal{G}$, tolerating a semi-honest adversary corrupting t parties.

2.1.2. *Indistinguishability-Based THC.*

Moran et al. [29] gave a weaker definition of topology-hiding computation: IND-CTA security (indistinguishability under Chosen Topology Attack). We will next provide the explicit definitions for THB and THAB.

Definition 2.4. (1-IND-CTA THB) A broadcast protocol π is *indistinguishable under chosen topology attack against one semi-honest corruption* (1-IND-CTA *secure)* with respect to a graph class $\mathcal{G}$, if for any PPT adversary $\mathcal{A}$ there exists a negligible function negl, such that for every $\lambda \in \mathbb{N}$ it holds that

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

where $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$ is as defined in Fig. 8 and the probability is taken over the random coins of the experiment and of the adversary.

Definition 2.4 can be extended to support t corruptions, denoted t-IND-CTA broadcast, by having the adversary choose a set $I \subseteq [n]$ of size t satisfying $I \subseteq V(G_0) \cap V(G_1)$ in $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}$, instead of choosing a single node v.

The definition of anonymous broadcast is similar except that the adversary can choose different broadcasting parties for the different executions; this is opposed to broadcast where the same node acted as broadcaster in both executions.

Definition 2.5. (1-IND-CTA THAB) An anonymous broadcast protocol π is *indistinguishable under chosen topology attack against one semi-honest corruption* (1-IND-CTA *secure*) with respect to a graph class $\mathcal{G}$, if for any PPT adversary $\mathcal{A}$ there exists a negligible

The experiment $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$

Choice phase. The challenger invokes $\mathcal{A}$ on input $(\mathcal{G}, 1^\lambda)$. $\mathcal{A}$ chooses two graphs $G_0, G_1 \in \mathcal{G}$ (whose vertex sets we denote respectively $V(G_0)$ and $V(G_1)$), a broadcaster node $u \in V(G_0) \cap V(G_1)$, a message $m \in \{0,1\}$, and a corrupted node $v \in V(G_0) \cap V(G_1)$ with $\mathcal{N}_{G_0}(v) = \mathcal{N}_{G_1}(v)$. Next, the adversary returns (G_0, G_1, u, m, v). If $\mathcal{A}$'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 $\pi(1^\lambda)$ with node u broadcasting message m over graph G_b, where the adversary controls node v.

Output phase. At the conclusion of the execution of π, $\mathcal{A}$ outputs b'. If $b = b'$, the experiment outputs 1; otherwise 0.

Fig. 8. The 1-IND-CTA broadcast experiment.

The experiment $\mathsf{ExpTHAB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$

Choice phase. The challenger invokes $\mathcal{A}$ on input $(\mathcal{G}, 1^\lambda)$. $\mathcal{A}$ chooses two graphs $G_0, G_1 \in \mathcal{G}$, two broadcaster nodes $u_0 \in V(G_0)$ and $u_1 \in V(G_1)$, a message $m \in \{0,1\}$, and a corrupted node $v \in V(G_0) \cap V(G_1)$ with $\mathcal{N}_{G_0}(v) = \mathcal{N}_{G_1}(v)$. Next, the adversary returns $(G_0, G_1, u_0, u_1, v, m)$. If $\mathcal{A}$'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 $\pi(1^\lambda)$ with node u_b broadcasting message m over graph G_b, where the adversary controls node v.

Output phase. At the conclusion of the execution of π, $\mathcal{A}$ outputs b'. If $b = b'$, the experiment outputs 1; otherwise 0.

Fig. 9. The 1-IND-CTA anonymous broadcast experiment.

function negl, such that for every $\lambda \in \mathbb{N}$ it holds that

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

where $\mathsf{ExpTHAB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$ is as defined in Fig. 9 and the probability is taken over the random coins of the experiment and of the adversary.

Definition 2.5 can be extended to support t corruptions, denoted t-IND-CTA anonymous broadcast, by having the adversary choose a set $I \subseteq [n]$ of size t satisfying $I \subseteq V(G_0) \cap V(G_1)$ in $\mathsf{ExpTHAB}^{\texttt{1-ind-cta}}$, instead of choosing a single node v.

As shown in [29], the indistinguishability-based definition is in fact implied by its simulation-based counterpart.

Proposition 2.6. *If π is 1-THB (resp., 1-THAB) with respect to $\mathcal{G}$, then π is a 1-IND-CTA secure broadcast (resp., anonymous broadcast) protocol with respect to $\mathcal{G}$.*

In our lower bounds in Sects. 4 and 5, we will require basic sequential composition guarantees from 1-IND-CTA (anonymous) broadcast protocols. Namely, we will compose λ executions of 1-bit (anonymous) broadcast protocols over the same graph to transmit longer messages, and we will run two instances of λ-bit (anonymous) broad-

cast over two different graphs within the class $\mathcal{G}$. Definitions 2.4 and 2.5 remain secure under sequential composition via a simple hybrid argument, as we consider *independent* executions and a *semi-honest* adversary.

Definition 2.7. (sequential composition) A broadcast protocol π is 1-IND-CTA secure under q sequential composition with respect to a graph class $\mathcal{G}$, if for any PPT adversary $\mathcal{A}$ there exists a negligible function negl, such that for every $\lambda \in \mathbb{N}$ it holds that

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

where $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(q,\lambda)$ is as defined in a similar way to $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$ 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 i^{th} execution is with graph G_b^i and sender u^i broadcasts message m^i.

1-IND-CTA anonymous broadcast secure under q sequential composition is defined in an analogous way.

Lemma 2.8. *Let $q \in \text{poly}(\lambda)$. If π is a 1-IND-CTA secure broadcast (resp., anonymous broadcast) protocol with respect to $\mathcal{G}$, then π is secure under q sequential composition.*

Proof. We prove the lemma for broadcast; the case of anonymous broadcast follows in an analogous way. The proof proceeds via a sequence of hybrid experiments, where the i^{th} hybrid H_i is defined as $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(q,\lambda)$, except that for $j=1,\ldots,i$ the j^{th} execution of π is with graph G_1^j and for $j=i+1,\ldots,q$ with graph G_0^j. It follows that H_0 is exactly $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(q,\lambda)$ with $b=0$ and H_q is exactly $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(q,\lambda)$ with $b=1$.

For each $i=0,\ldots,q$, let ε_i be the probability that H_i outputs 1 after interacting with adversary $\mathcal{A}$. It is thus left to show that for each $i \in [q]$ it holds that $|\varepsilon_{i-1}-\varepsilon_i| \leq \mathsf{negl}(\lambda)$. Indeed, for every i we construct an adversary $\mathcal{A}_i$ that wins $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$ with the same probability. Adversary $\mathcal{A}_i$ awaits input $\{(G_0^j, G_1^j, u^j, m^j)\}_{j\in[q]}$ and v from $\mathcal{A}$ and forwards $(G_0^i, G_1^i, u^i, m^i, v)$ to its own experiment. During the challenge phase, $\mathcal{A}_i$ proceeds as follows:

- For $j=1,\ldots,i-1$, $\mathcal{A}_i$ simulates the j^{th} execution toward $\mathcal{A}$ using (G_1^j, u^j, m^j).
- For $j=i$, $\mathcal{A}_i$ interacts with $\mathcal{A}$ by redirecting the messages from its own experiment.
- For $j=i+1,\ldots,q$, $\mathcal{A}_i$ simulates the j^{th} execution toward $\mathcal{A}$ using (G_0^j, u^j, m^j).

Now, if in the experiment $\mathsf{ExpTHB}^{\text{1-ind-cta}}_{\pi,\mathcal{G},\mathcal{A}}(\lambda)$ the challenger tossed $b=0$, then the experiment simulated by $\mathcal{A}_i$ is equal to H_{i-1}; otherwise, if $b=1$, the experiment is equal to H_i. We thus have $|\varepsilon_{i-1}-\varepsilon_i| \leq \mathsf{negl}(\lambda)$, as required. □

$$\mathcal{G}_{\text{oriented-5-path}} := \left\{ \begin{array}{l} ① \to ② \to ③ \to ④ \to ⑤, \\ ① \to ③ \to ④ \to ⑤ \to ②, \\ ① \to ④ \to ⑤ \to ② \to ③, \\ ① \to ⑤ \to ② \to ③ \to ④ \end{array} \right\}$$

Fig. 10. Oriented lines on five nodes. Each node knows its *direction* to the broadcaster ①, but may not know its *distance*. For example, the view of node ④ is identical in the first and second lines; the view of node ⑤ is identical in the second and third lines; the view of node ② is identical in the third and fourth lines; and the view of node ③ is identical in the first and fourth lines.

3. Key Agreement

We recall in Definition 3.1 the definition of key agreement and infinitely often key agreement from [20].

Definition 3.1. (Two-Party (infinitely often) Key-Agreement [20, Def. 2.2]) Let π be two party, interactive Turing machine protocol which generates communication transcript Γ, and outputs $A \in \{0,1\}$ (for the first party) and $B \in \{0,1\}$ for the second. π is a *key-agreement* (KA) protocol if on input 1^λ, $\Pr[A=0] = \Pr[B=0] = 1/2$ and for any PPT Turing machine E, for almost all λ it holds that

1. **(hiding)** $\Pr[E(1^\lambda, \Gamma) = A \mid A = B] < 1/2 + \mathsf{negl}(\lambda)$.
2. **(agreement)** $\Pr[A = B = 0] = \Pr[A = B = 1] \geq 1 - \mathsf{negl}(\lambda)$.

We say that a protocol achieves *infinitely often* key agreement (io-KA) if for any probabilistic polynomial time E, (1) and (2) hold for infinitely many values of λ.

4. THB Lower Bounds

In this section, we demonstrate that achieving broadcast while hiding certain graph properties necessitates cryptographic assumptions. In Sect. 4.1, we show that hiding the *distance* to the broadcaster requires infinitely often KA, and in Sect. 4.2 that hiding distance-of-neighbors requires KA.

4.1. *Hiding Distance Requires io-KA (The Oriented 5-Path)*

In this section, we show that hiding the distance from the broadcaster, in constant rounds, requires *infinitely often key agreement* (io-KA). In particular, we will show that any constant-round protocol for the class $\mathcal{G}_{\text{oriented-5-path}}$ (Fig. 10) implies io-KA. In this class, the nodes ②, ③, ④, ⑤ always know the direction of the broadcaster, node ① (it is in the direction of their lowest-valued neighbor, mod 5), but cannot distinguish (from their local neighborhood) whether they are distance 2 or 3 from the broadcaster. For example, ③ cannot distinguish between ①-②-③-④-⑤ and ①-⑤-②-③-④, as in both cases its local neighborhood is ②-③-④. Note that if just distance is leaked to this class, the trivial flooding protocol is secure.

Intriguingly, the key-agreement "construction" of this section is not fully black-box (nor is it even explicit). Our result critically requires that the THB protocol for $\mathcal{G}_{\text{oriented-5-path}}$ is *constant-round*. We remark that such a limitation is inherent, as we demonstrate in Appendix C that $\mathcal{G}_{\text{oriented-5-path}}$ *unconditionally* admits an ε-statistically 1-secure THB protocol that works in $O(1/\varepsilon)$ rounds, for any $\varepsilon > 0$.[17] In contrast, the key-agreement construction of Sect. 4.2 *is* fully black-box and rules out the existence of such an upper bound for the class $\mathcal{G}_{\text{triangle}}$. It remains open whether an ε-statistically 1-secure THB for $\mathcal{G}_{\text{oriented-5-path}}$ in $o(1/\varepsilon)$ rounds requires io-KA, or more generally whether negligible security in polynomial rounds requires io-KA.[18]

We refer to Definition 3.1 for the definitions on key agreement and infinitely often key agreement, as taken from Haitner et al. [20]. We now state the main theorem of this section.

Theorem 4.1. *If there exists a constant-round* 1-*IND-CTA-secure broadcast protocol for the class* $\mathcal{G}_{\text{oriented-5-path}}$*, then infinitely often key agreement exists.*

The high-level idea of the proof follows the *artificial overextension* argument, described in Sect. 1.2.2. Consider for simplicity that there exists an R-round THB protocol for $\mathcal{G}_{\text{oriented-5-path}}$ we will attempt to construct a candidate KA protocol. This protocol works by trying to simulate the THB protocol on two graphs in the class. The *hiding* property follows from the security of THB protocol, but *agreement* is not clear. What we can argue is that either, there *exists* a means achieving agreement (in which case we have a key-agreement protocol) *or* we can argue that a new candidate protocol has the *hiding* property. This new protocol has the feature that one of the graphs contains a node that is distance 5 from the broadcaster (as opposed to 4, as before). We then have the same either/or guarantee as before: Either there exist a means of achieving agreement, or a new candidate protocol is hiding (which contains a node distance 6 from the broadcaster). We continue arguing in this manner. We ultimately conclude that some intermediate KA protocol must be secure, because the R^{th} such protocol contains a node distance $R + 3$ from the broadcaster. There is no hope of the broadcast bit ever reaching such a node, and from this fact, *agreement* is trivial. It follows that some intermediate protocol must have achieved both *hiding* as well as *agreement*.

Proof. Let R be a constant, and let π be an R-round 1-IND-CTA-secure broadcast protocol for $\mathcal{G}_{\text{oriented-5-path}}$. By the 1-IND-CTA security of π, it holds that for each of the nodes ②, ③, ④, and ⑤ there exist a pair of graphs in $\mathcal{G}_{\text{oriented-5-path}}$ in which they cannot know their distance to the broadcaster ①; see Fig. 10 for an illustration.

We leverage this fact to specify a basic key-agreement protocol, π_1 (see Fig. 11), that we will either be able to extend into genuine key agreement *or* reason about a new protocol, π_2, on longer graphs (that, in turn, we will either be able to extend into key agreement *or* reason about a new protocol on longer graphs, π_3, etc.). If none of $\pi_1, \ldots, \pi_R$ can be extended into key agreement, we will reach a contradiction.

[17] In fact, the upper bound holds for a large body of graph classes, where only distance needs to be hidden.

[18] Our techniques can be extended to show an ε-statistically 1-secure THB for $\mathcal{G}_{\text{oriented-5-path}}$ in $c \cdot \log \varepsilon^{-1}$ rounds, where c is a constant, requires io-KA, but the gap between this and the upper bound remains exponential.

Protocol $\pi_1(1^\lambda)$

Public Parameters: The security parameter λ and a broadcast protocol π for $\mathcal{G}_{\text{oriented-5-path}}$.

The Protocol:

- `Initialization:` Alice sends two random messages $m^1, m^2 \leftarrow \{0,1\}^\lambda$ to Bob.
- `Setting the underlying graphs:`
  - Alice flips a coin A. If $A = 0$, Alice imagines herself holding the partial graphs:

    $$G_A^1 = ①\text{-}②\text{-}③ \qquad G_A^2 = ①\text{-}⑤\text{-}②\text{-}③$$

    If $A = 1$, Alice imagines herself holding the partial graphs:

    $$G_A^1 = ①\text{-}⑤\text{-}②\text{-}③ \qquad G_A^2 = ①\text{-}②\text{-}③$$

  - Bob flips a coin B. If $B = 0$, Bob imagines himself holding the partial graphs:

    $$G_B^1 = ④\text{-}⑤ \qquad G_B^2 = ④$$

    If $c_B = 1$, Bob imagines himself holding the partial graphs:

    $$G_B^1 = ④ \qquad G_B^2 = ④\text{-}⑤$$

- `Running the broadcast protocol:` Alice and Bob jointly simulate $\pi(1^\lambda)$, first on $G^1 = G_A^1$-G_B^1 with broadcast message m^1 and then on $G^2 = G_A^2$-G_B^2 with broadcast message m^2; in both cases Alice simulates the broadcaster node ①. The graphs formed by connecting the rightmost node in G_A^1 (resp., G_A^2) to the leftmost node in G_B^1 (resp., G_B^2). More specifically, Alice simulates nodes in G_A^1 (resp., G_A^2) via their next-message functions and Bob simulates nodes in G_B^1 (resp., G_B^2) via their next-message functions. When the rightmost node of G_A^1 (resp., G_A^2) needs to send a message to the leftmost node in G_B^1 (resp., G_B^2) and vice versa, Alice sends the message through the channel to Bob and vice versa.
- `Output:`
  - If the outputs of Bob's "rightmost" nodes are m^1 in the first execution and m^2 in the second, Bob outputs B; otherwise, Bob sends `abort` to Alice and outputs $\perp$.
  - If Alice did not receive `abort`, she output A; Otherwise, she outputs $\perp$.

Fig. 11. The basic key-agreement protocol π_1.

The basic protocol π_1 starts with Alice sending two long and random message m^1 and m^2 to Bob. Next, Alice and Bob flip secrets coins A and B, resp., and execute two instances of the broadcast protocol: the first on m^1 and the second on m^2. If $A = 0$, Alice simulates ①-②-③ in the first execution and ①-⑤-②-③ in the second, and if $A = 1$, she reverses the order. Similarly, if $B = 0$, Bob simulates ④-⑤ in the first execution and ④ in the second, and if $B = 1$, he reverses the order. If the output of Bob's "rightmost" node is m^1 in the first execution and m^2 in the second, Alice and Bob output A and B, resp.; otherwise, they abort. For convenience, we show all possible graphs that Alice and Bob might jointly simulate the THB protocol in Fig. 12.

Observe that conditioned on $A = B$ (Alice and Bob's coins agreeing), both executions of π (within π_1) are on graphs in $\mathcal{G}_{\text{oriented-5-path}}$. Thus, if Eve is eavesdropping on Alice and Bob's communication, her view is a subset of the view of node ③ under protocol π. Because the view of ③ is locally identical in all possible graphs, if Eve can distinguish between the case of $A = B = 0$ and $A = B = 1$ with non-negligible advantage, we can use Eve to break the security of π.

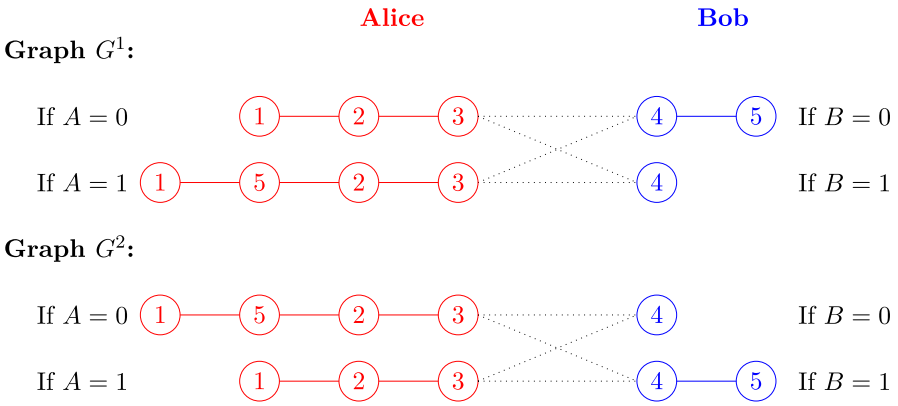


Fig. 12. The possible graphs simulated in π_1. Color indicates who is responsible for simulation: Alice - Bob. Eve can see communication on the dotted edges between them.

Thus, we are halfway to a key agreement. The problem is that A and B are uncorrelated. We would like to "filter out" the cases where $A \neq B$, without hurting the security of π too much. We will focus on a very specific method for doing so which will allow us make progress: distinguishing via the view of ④ in $G_B =$ ④-⑤.

We have two cases:

Case 1: The view of ④ in ①-②-③-④-⑤ under $\pi(1^\lambda)$ with message m (denoted $VIEW_4^{12345}(\lambda, m)$) is *not* computationally indistinguishable from the view of ④ in ①-⑤-②-③-④-⑤ under $\pi(1^\lambda)$ with message m (denoted $VIEW_4^{152345}(\lambda, m)$). In particular, there exist a distinguisher D and a constant c_D such that D runs in time λ^{c_D} and for infinitely many values of λ can distinguish with advantage λ^{-c_D}. Moreover, let p_D denote the probability D outputs 1 on $VIEW_4^{12345}(\lambda, m)$.[19]

We will use such a distinguisher, D, to construct an infinitely often key-agreement protocol, KA_1^{D,c_D} (formally described in Fig. 13).[20] It is tempting to use D to reject runs where $A \neq B$, but we have no guarantee that the view of Eve when $A = B = 0$ and $A = B = 1$ remains indistinguishable *after conditioning on* $D = 1$. We get around this by simply amplifying the distinguisher's advantage until it effectively never makes a mistake, and thus, Eve's view conditioning on the amplified decision is statistically close to simply conditioning on $A = B$.

Claim 4.2. *If there exists an infinite set $I \subseteq \mathbb{N}$ and a constant c_D such that there is a randomized D that runs in time λ^{c_D} and for which*

$$\left|\Pr\left[D\left(1^\lambda, m, VIEW_4^{12345}(\lambda, m)\right) = 1\right]\right.$$

[19] Note that because the advantage of D is bounded from below by λ^{-c_D}, for each λ we can approximate p_D in polynomial time up to a $\lambda^{-c_D}/2$ factor.

[20] Note that our protocol is only infinitely often correlated ($A = B$); hence, the security property will hold for all but finitely many λ for any efficient distinguisher.

Protocol $\mathsf{KA}_1^{D,c_D}(1^\lambda)$

Public Parameters: The security parameter λ, a broadcast protocol π for $\mathcal{G}_{\text{oriented-5-path}}$, and a distinguisher D that runs in time $O(\lambda^{c_D})$ (assumed to have advantage λ^{-c_D}).

The Protocol:

- `Initialization:`
  - For each $i \in [\lambda^{2c_D}]$ Alice sends random messages $m_i^1, m_i^2 \leftarrow \{0,1\}^\lambda$ to Bob.
  - For each $i \in [\lambda^{2c_D}]$ Bob samples X_i according to $\text{VIEW}_4^{12345}(\lambda, m_i)$ where m_i is drawn uniformly at random. Let
$$\tilde{p}_D^{A=B} := \frac{\sum_{i=1}^{\lambda^{2c_D}} D(1^\lambda, m_i, X_i)}{\lambda^{2c_D}}.$$
- `Setting the underlying graphs:`
  - Alice flips a coin A. If $A = 0$, Alice imagines herself holding the partial graphs:
$$G_A^1 = ①\text{-}②\text{-}③ \qquad G_A^2 = ①\text{-}⑤\text{-}②\text{-}③$$
    If $A = 1$, Alice imagines herself holding the partial graphs:
$$G_A^1 = ①\text{-}⑤\text{-}②\text{-}③ \qquad G_A^2 = ①\text{-}②\text{-}③$$
  - Bob flips a coin B. If $B = 0$, Bob imagines himself holding the partial graphs:
$$G_B^1 = ④\text{-}⑤ \qquad G_B^2 = ④$$
    If $B = 1$, Bob imagines himself holding the partial graphs:
$$G_B^1 = ④ \qquad G_B^2 = ④\text{-}⑤$$
- `Running the broadcast protocol:` For $i = 1, \ldots, \lambda^{2c_D}$,
  - Alice and Bob jointly simulate $\pi(1^\lambda)$, first on $G^1 = G_A^1\text{-}G_B^1$ with broadcast message m_i^1 and then on $G^2 = G_A^2\text{-}G_B^2$ with broadcast message m_i^2 (as described in π_1).
  - Bob sets d_i to the output of D on the view of party ④ in the graph containing ④-⑤ (G^1 if $B = 0$ and G^2 if $B = 1$).
- `Output:`
  - If $|\sum_i d_i - \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}| > \lambda^{c_D}/2$, Bob sends `start-over` to Alice and starts over. (After λ `start-over` messages, halt.) Otherwise, output B.
  - If Alice did not receive `start-over`, output A. Otherwise, start over. (After λ `start-over` messages, halt.)

Fig. 13. First attempt at secure bit agreement, KA_1^{D,c_D}.

$$-\Pr\left[D\left(1^\lambda, m, VIEW_4^{152345}(\lambda, m)\right) = 1\right]\Big| > \lambda^{-c_D}$$

(where m is uniformly distributed) for all $\lambda \in I$, then KA_1^{D,c_D} (Fig. 13) is a key-agreement protocol in the infinite set I.

Proof. Without loss of generality, assume $c_D \geq 1$. Let A' and B' denote the final outputs of Alice and Bob.

First notice that by symmetry our distinguisher does not bias Alice and Bob's output when they agree:

$$\Pr\left[A' = B' = 0 \;\middle|\; \sum d_i \geq \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}/2\right] = \Pr\left[A' = B' = 1 \;\middle|\; \sum d_i \geq \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}/2\right].$$

Now, we will restrict our attention to $\lambda \in I$ (where $|\Pr[D(1^\lambda, m, VIEW_4^{12345}(\lambda, m)) = 1] - \Pr[D(1^\lambda, m, VIEW_4^{152345}(\lambda, m)) = 1]| > \lambda^{-c_D}$).

Observe that by Hoeffding bounds[21] the following event happens with probability at most $\exp(-\Omega(\lambda))$:

$$\left|\tilde{p}_D^{A=B} - \mathbb{E}[D(1^\lambda, m, VIEW_4^{12345}(\lambda, m))]\right| > \lambda^{-c_D}/4,$$

where $\tilde{p}_D^{A=B}$ is the fraction of instances that output 1 in Bob's sampling of λ^{2c_D} i.i.d. instances of $D(1^\lambda, m, VIEW_4^{12345}(\lambda, m))$ (in the `Initialization` step).

Note that by our assumption on the distinguisher D, it holds that

$$\left|\mathbb{E}[D(1^\lambda, m, VIEW_4^{12345}(\lambda, m))] - \mathbb{E}[D(1^\lambda, m, VIEW_4^{152345}(\lambda))]\right| > \lambda^{-c_D}.$$

Thus from the triangle inequality we can deduce that conditioned on the above event not happening,

$$\left|\tilde{p}_D^{A=B} - \mathbb{E}[D(1^\lambda, m, VIEW_4^{152345}(\lambda, m))]\right| > 3\lambda^{-c_D}/4.$$

Next, by applying Hoeffding's bound again we have that with probability $\exp(-\Omega(\lambda))$, $|\sum_i d_i - \mathbb{E}[D(1^\lambda, m, VIEW_4^{12345}(\lambda, m))] \cdot \lambda^{2c_D}| > \lambda^{c_D}/4$ in the case that $A = B$, and $|\sum_i d_i - \mathbb{E}[D(1^\lambda, m, VIEW_4^{152345}(\lambda, m))] \cdot \lambda^{2c_D}| > \lambda^{c_D}/4$ in the case that $A \neq B$.

By a union bound, none of these events happen with probability $1-\exp(-\Omega(\lambda))$. If this is indeed the case, then we can apply the triangle inequality again to deduce that if $A = B$, then $|\sum_i d_i - \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}| < \lambda^{c_D}/2$, and if $A \neq B$ then $|\sum_i d_i - \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}| > \lambda^{c_D}/2$. Therefore, if $A = B$, Alice and Bob will output A and B, resp., with overwhelming probability. And if $A \neq B$, they will start over with overwhelming probability.

Finally, because $A = B$ with probability $1/2$, we expect that Alice and Bob will produce output in λ iterations with overwhelming probability. Let O denote the event that Alice and Bob produce output.

It remains to show that security is preserved. Because $VIEW_3^{12345}$ is computationally indistinguishable from $VIEW_3^{15234}$, by a hybrid argument $(X_1, \ldots, X_{\lambda^{2c_D}})$ is computationally indistinguishable from $(Y_1, \ldots, Y_{\lambda^{2c_D}})$ where each $X_i \sim (VIEW_3^{12345},$

[21] Recall that Hoeffding's bound says that given $X_1, \ldots, X_n$ i.i.d. indicator random variables, for any $\delta \geq 0$, $\Pr\left[|\sum X_i - \mathbb{E}[\sum X_i]| \geq \delta\right] \leq 2\exp(-2\delta^2/n)$.

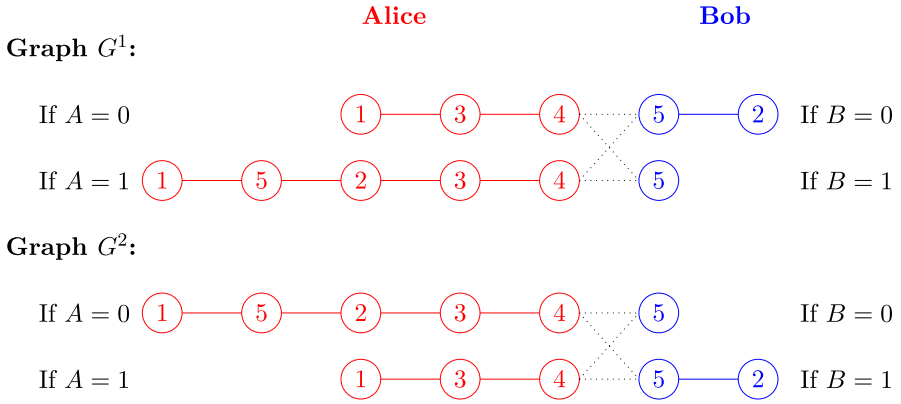


Fig. 14. The possible graphs simulated in π_2. Color indicates who is responsible for simulation: Alice - Bob. Eve can see communication on the dotted edges between them.

$VIEW_3^{15234}$) (corresponding to $A = B = 0$) and each $Y_i \sim (VIEW_3^{15234}, VIEW_3^{12345})$ (corresponding to $A = B = 1$),[22]

We showed that conditioned on $A = B$, the event O happens with overwhelming probability. Thus, we have that $(X_1, \ldots, X_d)$ is computationally indistinguishable from $(X_1, \ldots, X_d)$ conditioned on O. Similarly, $(Y_1, \ldots, Y_d)$ is computationally indistinguishable from $(Y_1, \ldots, Y_d)$ conditioned on O. Thus, no efficient adversary can distinguish the case that the output bits are $A' = B' = 0$ from the case that the output bits are $A' = B' = 1$.

This concludes the proof of Claim 4.2. □

Case 2: The view of ④ in ①-②-③-④-⑤ under $\pi(1^\lambda)$ is *computationally indistinguishable* from the view of ④ in ①-⑤-②-③-④-⑤ under $\pi(1^\lambda)$.

By the security of π, we have that $VIEW_4^{12345} \approx_c VIEW_4^{13452}$. Thus, $VIEW_4^{13452} \approx_c VIEW_4^{152345}$. In this case, we construct the basic key-agreement protocol π_2 which is defined identically as π_1 with the sole exception that Alice and Bob use different underlying graphs. If Alice's coin is $A = 0$, she simulates $G_A^1 =$ ①-③-④ in the first execution and $G_A^2 =$ ①-⑤-②-③-④ in the second, and if $A = 1$, she reverses the order. Similarly, if $B = 0$, Bob simulates $G_B^1 =$ ⑤-② in the first execution and $G_B^2 =$ ⑤ in the second, and if $B = 1$ he reverses the order. See Fig. 14 for an illustration.

It follows from a hybrid argument, that conditioned on $A = B$, no efficient eavesdropper can distinguish $A = 0$ from $A = 1$ in π_2. Thus again, we are halfway to key agreement. Notice (Fig. 14) that the length of the longest line graph has increased by 1 over π_1.

As before, we have two cases based on whether the view of the second to last node in the longest graph can be distinguished from the other graph where that node is also penultimate: here, whether or not $VIEW_5^{15'23452}$ is indistinguishable from $VIEW_5^{13452}$

[22] Lemma 2.8 implies that each $X_1, \ldots, X_{\lambda^{2c_D}}$ (and similarly, $Y_1, \ldots, Y_{\lambda^{2c_D}}$) is indistinguishable from $Z_1, \ldots, Z_{\lambda^{2c_D}}$ where each $Z_i \sim (VIEW_3^{12345} VIEW_3^{12345})$.

(where the prime in $5'$ is simply used to indicate which copy of ⑤'s view we are considering, both copies "think" they are ⑤):

1. If they are *not* indistinguishable, then we can build an infinitely often key-agreement protocol, KA_2^{D,c_D} (see Fig. 15), via the same method as before.
2. If they are indistinguishable, then we will build a new protocol stub, π_3, with security properties that contains a graph longer than any in π_2.

We proceed iteratively in this manner for R iterations. Assume for the sake of contradiction that π does not imply key agreement. If so, it must be that after R iterations, in each protocol π_i the view of the node second from right in the longest graph is indistinguishable from a graph in $\mathcal{G}_{\text{oriented-5-path}}$ (i.e., none of $\pi_1, \ldots, \pi_R$ could be turned into a key agreement KA_i^{D,c_D} as described in Fig. 15). But because π only runs for R rounds and the second to last node is distance greater than R from ① in the longest graph of π_R, the node's final output is independent of the broadcast message. Thus, there must exist a distinguisher with success probability at least $1/2 - \mathsf{negl}(\lambda)$ that simply checks if that node gets the correct output and we reach a contradiction. It follows that one of the protocols π_i, for some $i \in [R]$, can be transformed into an infinitely often key agreement KA_i^{D,c_D}.

More formally, we can generalize our above argument to the following claim.

Claim 4.3. *For $i \in [R]$ and $j \in [4]$ let $u_j = (i + j - 2 \bmod 4) + 2$. Then, there exists $i \in [R]$ for which the following hold:*

1. *For all* PPT *D, the following holds for almost all λ*

$$\left|\Pr\left[D(1^\lambda, m, VIEW_{u_2}^{1u_1u_2u_3u_4}(\lambda, m)) = 1\right] - \Pr\left[D(1^\lambda, m, VIEW_{u_2}^{1523452\cdots u_1u_2u_3}(\lambda, m)) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

   where m is uniformly distributed and $15234523452 \cdots u_1u_2u_3$ is the graph with $i + 4$ nodes.
2. *There exist an infinite set $I \subseteq \mathbb{N}$ and a distinguisher D^* that runs in time $\lambda^{c_D^*}$, for some constant c_D^*, such that for all $\lambda \in I$,*

$$\left|\Pr\left[D^*(1^\lambda, m, VIEW_{u_3}^{1u_1u_2u_3u_4}(\lambda, m)) = 1\right] - \Pr\left[D^*(1^\lambda, m, VIEW_{u_3}^{1523452\cdots u_1u_2u_3u_4}(\lambda, m)) = 1\right]\right| > \lambda^{-c_D^*},$$

   where m is uniformly distributed $15234523452 \cdots u_1u_2u_3$ is the graph with $i + 4$ nodes.

By inspecting $\mathsf{KA}_i^{D^*,c_D^*}$ (Fig. 15) and following the same argument above, we see that Item 1 in Claim 4.3 implies that $\mathsf{KA}_i^{D^*,c_D^*}$ has the *hiding* property (for any algorithm D^* and constant c_D^*), whereas Item 2 implies that $\mathsf{KA}_i^{D^*,c_D^*}$ has the *agreement* property (infinitely often). Thus given this claim, we can apply the same analysis from Case 1

above to immediately get that for some i, $\mathsf{KA}_i^{D^*,c^*_D}$ realizes infinitely often key agreement. So it suffices to prove the claim to complete the proof.

To prove the claim, observe the following:

1. Item 1 of Claim 4.3 is true for $i = 1$ by the topology-hiding property of π.
2. Item 2 of Claim 4.3 is true for $i = R$. This is by virtue of the fact that in the case of $i = R$, $\boxed{\mathsf{u}_3}$ is distance $> R$ from $\boxed{1}$ in the graph $\boxed{1}$-$\boxed{5}$-$\boxed{2}$-$\boxed{3}$-$\boxed{4}$-$\boxed{5}$-$\cdots$-$\boxed{\mathsf{u}_4}$-$\boxed{\mathsf{u}_1}$-$\boxed{\mathsf{u}_2}$-$\boxed{\mathsf{u}_3}$-$\boxed{\mathsf{u}_4}$ and thus its output is independent of m and will only coincide with negligible probability. On the other hand, in the case that graph is $\boxed{1}$-$\boxed{\mathsf{u}_1}$-$\boxed{\mathsf{u}_2}$-$\boxed{\mathsf{u}_3}$-$\boxed{\mathsf{u}_4}$, by the correctness of the topology-hiding broadcast, $\boxed{\mathsf{u}_3}$ will output m with overwhelming probability.
3. If Item 2 of Claim 4.3 is false for i, then Item 1 of Claim 4.3 is true for $i + 1$. (Item 1 for $i + 1$ is the negation of Item 2 for i, for any $i \in [R - 1]$.)

Now, suppose the claim is false, then for all i at least one of the properties must be false. Then, by Observation 1 above, this means Item 2 of Claim 4.3 must be false for $i = 1$. Further, by Observation 3 and the assumption the claim is false, it follows by induction that Item 2 of Claim 4.3 must be false for all i. But, this contradicts Observation 2: the validity of Item 2 of Claim 4.3 for $i = R$.

This concludes the proof of Theorem 4.1. □

4.2. *Hiding Distance-of-Neighbors Requires KA (The Triangle)*

In this section, we show that THB on the triangle (with a potential missing edge), in which the direction/distance of each party to the broadcaster is fixed but the distance-of-neighbors is kept hidden, implies the existence of a key-agreement protocol.

Consider the class $\mathcal{G}_{\mathsf{triangle}} = \{G^0_{\mathsf{tr}}, G^1_{\mathsf{tr}}, G^2_{\mathsf{tr}}\}$ as defined in Fig. 16, which we (abusively) call "*the Triangle*." The players are the nodes $\boxed{1}, \boxed{2}, \boxed{3}$ with the broadcaster always being $\boxed{1}$; nodes $\boxed{2}$ and $\boxed{3}$ are connected, and $\boxed{2}$ and/or $\boxed{3}$ is connected to $\boxed{1}$. The *secret of the topology* can be summarized as follows: If one of the two non-broadcasting parties $\boxed{2}$ or $\boxed{3}$ is connected to the broadcaster, it does not know whether the other is connected as well. In other words, $\boxed{2}$ cannot distinguish between G^0_{tr} and G^2_{tr}, while $\boxed{3}$ cannot distinguish between G^0_{tr} and G^1_{tr}. As discussed in Sect. 1.2.1, an eavesdropper Eve having access to only the communication between $\boxed{2}$ and $\boxed{3}$ has strictly less information than either $\boxed{2}$ or $\boxed{3}$ individually. In particular, it follows that Eve cannot distinguish neither between G^0_{tr} and G^2_{tr} nor between G^0_{tr} and G^1_{tr} and transitively cannot distinguish between G^1_{tr} and G^2_{tr}: the paths $\boxed{1}$-$\overset{\text{(Eve)}}{\boxed{2}}$-$\boxed{3}$ and $\boxed{2}$-$\overset{\text{(Eve)}}{\boxed{3}}$-$\boxed{1}$, where node $\boxed{1}$ is the broadcaster. With this observation, we adopt the technique from [4], which shows how 1-THB on a line implies KA, to show that 1-THB on $\mathcal{G}_{\mathsf{triangle}}$ implies KA.

Note that preserving the secret of the topology of $\mathcal{G}_{\mathsf{triangle}}$ can also be reformulated as "hiding the neighbor distances" from the parties. Indeed, for $\boxed{2}$ (resp., $\boxed{3}$) knowing the topology means knowing if $\boxed{3}$ (resp., $\boxed{2}$) is at distance one or two from the broadcaster.

Theorem 4.4. *(THB on $\mathcal{G}_{\mathsf{triangle}}$ requires KA) If there exists a* 1-*IND-CTA-secure broadcast protocol for the class $\mathcal{G}_{\mathsf{triangle}}$, then there exists a key-agreement protocol.*

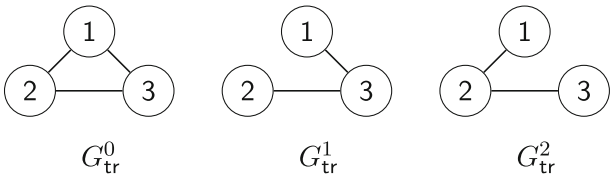
Protocol $\mathsf{KA}_i^{D,c_D}(1^\lambda)$

Public Parameters: The security parameter λ, a broadcast protocol π for $\mathcal{G}_{\mathsf{oriented\text{-}5\text{-}path}}$, and a distinguisher D that runs in time $O(\lambda^{c_D})$ (assumed to have advantage λ^{-c_D}). Let $u_j = (i + j - 2 \mod 4) + 2$ for $j \in [4]$.

The Protocol:

- `Initialization:`
  - For each $i \in [\lambda^{2c_D}]$ Alice sends random messages $m_i^1, m_i^2 \leftarrow \{0,1\}^\lambda$ to Bob.
  - For each $i \in [\lambda^{2c_D}]$ Bob samples X_i according to $\text{VIEW}_{u_2}^{1u_1u_2u_3u_4}(\lambda, m_i)$ where m_i is drawn uniformly at random. Let
$$\tilde{p}_D^{A=B} := \frac{\sum_{i=1}^{\lambda^{2c_D}} D(1^\lambda, m_i, X_i)}{\lambda^{2c_D}}.$$
- `Setting the underlying graphs:`
  - Alice flips a coin A. If $A = 0$, Alice imagines herself holding the partial graphs:
$$G_A^1 = \textcircled{1}\text{-}\textcircled{u_1}\text{-}\textcircled{u_2} \qquad G_A^2 = \textcircled{1}\text{-}\textcircled{5}\text{-}\textcircled{2}\text{-}\textcircled{3}\text{-}\textcircled{4}\text{-}\textcircled{5}\text{-}\cdots\text{-}\textcircled{u_4}\text{-}\textcircled{u_1}\text{-}\textcircled{u_2}$$
    If $A = 1$, Alice imagines herself holding the partial graphs:
$$G_A^1 = \textcircled{1}\text{-}\textcircled{5}\text{-}\textcircled{2}\text{-}\textcircled{3}\text{-}\textcircled{4}\text{-}\textcircled{5}\text{-}\cdots\text{-}\textcircled{u_4}\text{-}\textcircled{u_1}\text{-}\textcircled{u_2} \qquad G_A^2 = \textcircled{1}\text{-}\textcircled{u_1}\text{-}\textcircled{u_2}$$
  - Bob flips a coin B. If $B = 0$, Bob imagines himself holding the partial graphs:
$$G_B^1 = \textcircled{u_3}\text{-}\textcircled{u_4} \qquad G_B^2 = \textcircled{u_3}$$
    If $B = 1$, Bob imagines himself holding the partial graphs:
$$G_B^1 = \textcircled{u_3} \qquad G_B^2 = \textcircled{u_3}\text{-}\textcircled{u_4}$$
- `Running the broadcast protocol:` For $i = 1, \ldots, \lambda^{2c_D}$:
  - Alice and Bob jointly simulate $\pi(1^\lambda)$, first on $G^1 = G_A^1$-G_B^1 with broadcast message m_i^1 and then on $G^2 = G_A^2$-G_B^2 with broadcast message m_i^2.
  - Bob sets d_i to the output of D on the view of party $\textcircled{u_3}$ in the graph containing $\textcircled{u_3}\text{-}\textcircled{u_4}$ (G^1 if $B = 0$ and G^2 if $B = 1$), with the corresponding m_i^b and 1^λ.
- `Output:`
  - If $|\sum_i d_i - \tilde{p}_D^{A=B} \cdot \lambda^{2c_D}| > \lambda^{c_D}/2$, Bob sends `start-over` to Alice and starts over. (After λ `start-over` messages, halt.) Otherwise, output B.
  - If Alice did not receive `start-over`, output A. Otherwise, start over. (After λ `start-over` messages, halt.)

Fig. 15. i^{th} attempt at secure bit agreement, KA_i^{D,c_D}.



G_{tr}^0 G_{tr}^1 G_{tr}^2

Fig. 16. The class $\mathcal{G}_{\mathsf{triangle}}$.

Key-Agreement Protocol

Public Parameters: A broadcast protocol π for $\mathcal{G}_{\mathsf{triangle}}$.

The Protocol:

- Alice samples two random strings $m_1, m_2 \leftarrow \{0,1\}^\lambda$ and sends them to Bob.
- Alice and Bob locally flip random coins $A, B \leftarrow \{0,1\}$, resp. Next, they jointly simulate two instances of the THB protocol, communicating the messages between ② and ③.
  - If $A = 1$, Alice will simulate ① (broadcasting m_1) and ② (connected as in G^2_{tr}) in the first execution and will only simulate ② in the second. If $A = 0$, Alice will simulate ② in the first execution and ① (broadcasting m_2) and ② (connected as in G^2_{tr}) in the second.
  - If $B = 0$, Bob will simulate nodes ① (broadcasting m_1) and ③ (connected as in G^1_{tr}) in the first execution and will only simulate ③ in the second. If $B = 1$, Bob will simulate node ③ in the first execution and ① (broadcasting m_2) and ③ (connected as in G^1_{tr}) in the second.
- `Output:`
  - If node ② outputs m_1 in the first instance of π and m_2 in the second, Alice outputs A; otherwise, Alice outputs $\perp$.
  - If node ③ outputs m_1 in the first instance of π and m_2 in the second, Bob outputs B; otherwise, Bob outputs $\perp$.

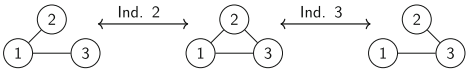
Fig. 17. Key-agreement protocol from 1-THB on $\mathcal{G}_{\mathsf{triangle}}$.

The rest of this section is dedicated to proving Theorem 4.4. Given a 1-THB protocol π for $\mathcal{G}_{\mathsf{triangle}}$, we will construct a key-agreement protocol. The construction of this KA protocol follows very closely the proof of Theorem 3.1 in [4], but we use the novel *phantom bridge* argument (see Remark 4.6) to reduce the security of the key-agreement scheme to the topology-hiding properties of π. Alice and Bob simulate two executions of the THB protocol π where each of the two players tosses a coin to determine whether to emulate the broadcaster in the first run or the second. Both they and an eavesdropper can identify when their coins yield the same outcome, and what it is, and the parties can try again. When the coins differ, however, we show that the eavesdropper cannot learn what the coins are (only that they indeed differ) and Alice and Bob can therefore use their coins to establish a secure shared key.

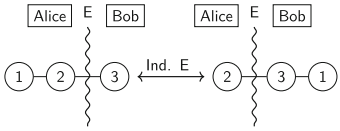
Lemma 4.5. *Let π be a 1-IND-CTA-secure broadcast protocol for $\mathcal{G}_{\mathsf{triangle}}$. Then, the protocol in Fig. 17 is a key-agreement protocol.*

Proof. Via sequential composition (Lemma 2.8) we may assume π is a λ-bit broadcast protocol. In a similar way to [4], we use π to construct the key-agreement protocol in Fig. 17.

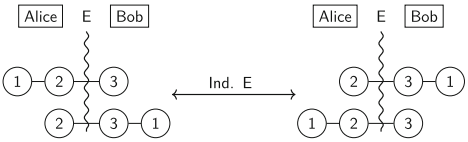
Correctness The proof of correctness is analogous to that of [4]. The high-level idea is that if $A \neq B$ (which occurs with probability $1/2$), then both Alice and Bob output $\perp$ (i.e., fails identifiably) with probability at least $1 - 2^{1-\lambda}$, since in one of the two executions of π neither Alice nor Bob are simulating a broadcasting node and therefore cannot both output the control string m_1 or m_2 with probability $\geq 2^{-\lambda}$. In the case where $A = B$ (which occurs with probability $1/2$), the behavior of the parties simulated by Alice and Bob in both executions falls under the correctness guarantees of π, which



(a) Indistinguishability of the views of ②, ③ in different topologies by the security of π.



(b) Eve cannot distinguish the two runs of π.



(c) Eve cannot determine the order of the two runs of π.

Fig. 18. Overview of the security proof: $(a) \implies (b) \implies (c)$.

means that with overwhelming probability both will output the same bit $A = B$. Wrapping up, both parties output $\perp$ with probability $1/2 + \mathsf{negl}(\lambda)$ and both output the same bit with probability $1/2 - \mathsf{negl}(\lambda)$. The parties can sequentially repeat the process until both output the same bit with overwhelming probability.

Security It remains is to prove that this candidate KA construction is secure. It suffices to show that when $A = B$ an eavesdropper Eve who has access to the communication between Alice and Bob has no advantage in determining the key-agreement output bit, i.e., in determining whether $A = B = 0$ or $A = B = 1$. Eve has access to the transcript of the communication between ② and ③ on two different runs of π, one on G^1_{tr} and the other G^2_{tr}. We reduce the problem of determining the order in which these executions are performed to that of determining, given the transcript of the communication between ② and ③ in a single run of π, if π was run on G^1_{tr} or G^2_{tr}. Finally, we show that this target problem is difficult, by topology-hiding of π. The proof overview is illustrated in Fig. 18. The technical novelty when compared to [4] resides in the proof of the last step, illustrated in Fig. 18a.

We now proceed with the reduction. Let $V^\sigma_{2,3}$, V^σ_2, and V^σ_3 be the random variables equal to, respectively, the transcript of the communication between ② and ③, the view of ②, and the view of ③ for the execution $\sigma \in \{1, 2\}$ of the protocol π. We denote $\pi_{2,3}(G)$, $\pi_2(G)$, and $\pi_3(G)$ their respective distributions on graph G. From now on condition on the event $A = B$, which occurs with probability $1/2$. The (conditional) advantage of Eve is the following:

$$\begin{aligned} \mathsf{Adv}(\text{Eve}) &:= |\Pr[\text{Eve wins}] - 1/2| \\ &= |\Pr_{V^1_{2,3}, V^2_{2,3} \sim \pi_{2,3}(G^2_{\mathsf{tr}})}[\text{Eve}(V^1_{2,3}, V^2_{2,3}) = 1] \\ &\quad - \Pr_{V^1_{2,3}, V^2_{2,3} \sim \pi_{2,3}(G^1_{\mathsf{tr}})}[\text{Eve}(V^1_{2,3}, V^2_{2,3}) = 1]| \end{aligned} \tag{1}$$

We now use the reduction from [4, Claim 3.3],[23] which shows that if Eve has non-negligible advantage then there is an efficient distinguisher between the transcripts of

[23] The claim itself cannot be invoked as the graphs are different, but the same proof works *verbatim*.

the communication between ② and ③ on graphs G^1_{tr} and G^2_{tr} in a single run of π (whereas Eve has to distinguish between the transcript for G^1_{tr} followed by the transcript for G^2_{tr} and the reverse). We then get that

$$\mathsf{Adv}(\text{Eve}) \leq 2 \cdot \sup_{\mathcal{A}\ \text{ppt}} \left| \Pr_{V_{2,3} \sim \pi_{2,3}(G^2_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] - \Pr_{V_{2,3} \sim \pi_{2,3}(G^1_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right|. \tag{2}$$

The reduction works by constructing a distinguisher $\mathcal{A}$ from Eve. $\mathcal{A}$ starts with a view $V_{2,3} \sim \pi_{2,3}(G^\sigma_{\mathsf{tr}})$ for some unknown $\sigma \in \{1, 2\}$, and flips two coins $\sigma' \in \{1, 2\}$ and $b \in \{0, 1\}$. $\mathcal{A}$ then crafts a view $V'_{2,3} \sim \pi_{2,3}(G^{\sigma'}_{\mathsf{tr}})$ and gives both views $V_{2,3}, V'_{2,3}$ to Eve in the order determined by coin b. With probability $1/2$, $\sigma = \sigma'$ and Eve is of no help, and with probability $1/2$, $\sigma \neq \sigma'$ and $\mathcal{A}$ inherits Eve's distinguishing advantage. It can be shown that $\mathsf{Adv}(\mathcal{A}) \geq \mathsf{Adv}(\text{Eve})/2$.

Finally, we show that the target problem of the reduction is hard. Namely, for every PPT $\mathcal{A}$ it holds that

$$\begin{aligned}
&\left| \Pr_{V_{2,3} \sim \pi_{2,3}(G^2_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] - \Pr_{V_{2,3} \sim \pi_{2,3}(G^1_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right| \\
&\leq \left| \Pr_{V_{2,3} \sim \pi_{2,3}(G^2_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] - \Pr_{V_{2,3} \sim \pi_{2,3}(G^0_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right. \\
&\quad \left. - \Pr_{V_{2,3} \sim \pi_{2,3}(G^1_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] + \Pr_{V_{2,3} \sim \pi_{2,3}(G^0_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right| \\
&\leq \left| \Pr_{V_{2,3} \sim \pi_{2,3}(G^2_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] - \Pr_{V_{2,3} \sim \pi_{2,3}(G^0_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right| \\
&\quad + \left| \Pr_{V_{2,3} \sim \pi_{2,3}(G^1_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] - \Pr_{V_{2,3} \sim \pi_{2,3}(G^0_{\mathsf{tr}})}[\mathcal{A}(V_{2,3}) = 1] \right| \\
&\leq \left| \Pr_{V_2 \sim \pi_2(G^2_{\mathsf{tr}})}[\mathcal{A}(V_2) = 1] - \Pr_{V_2 \sim \pi_2(G^0_{\mathsf{tr}})}[\mathcal{A}(V_2) = 1] \right| \\
&\quad + \left| \Pr_{V_3 \sim \pi_3(G^1_{\mathsf{tr}})}[\mathcal{A}(V_3) = 1] - \Pr_{V_3 \sim \pi_3(G^0_{\mathsf{tr}})}[\mathcal{A}(V_3) = 1] \right| \\
&\leq 2 \cdot \mathsf{negl}(\lambda).
\end{aligned} \tag{3}$$

The third inequality holds since V_2 and V_3 encapsulate $V_{2,3}$ and giving more information as input to $\mathcal{A}$ can only increase its distinguishing advantage. The fourth inequality follows from the topology-hiding properties of π since ② (resp., ③) has the same neighborhood in G^2_{tr} and G^0_{tr} (resp., G^1_{tr} and G^0_{tr}). Combining Equations (2) and (3) concludes the proof of Lemma 4.5. □

Remark 4.6. (The Phantom-Bridge) Theorem 4.4 relies on one core property of $\mathcal{G}_{\mathsf{triangle}}$ and can be extended to other graph-classes with that same property. If a graph class contains two graphs G_1 and G_2 with a bridge between two nodes u and v (i.e., an edge whose removal would separate the graph in two connected components) such that the broadcaster is on u's side in G_1 and on v's side in G_2, then 1-THB on this class implies KA. This observation is used in Sect. 8.1 to demonstrate the generality if this technique.

5. THAB Lower Bounds

In this section, we present our lower bounds for topology-hiding *anonymous* broadcast. In Sect. 5.1, we show that t-THAB on graphs that are not $(t+1)$-connected implies key agreement. In Sect. 5.2, we show that 1-THAB on the class of either 2 or 3 parties connected on a line implies infinitely often oblivious transfer (Definitions 5.3 and 5.2).

5.1. *Low Vertex Connectivity Requires KA*

We start by showing how t-THAB on a class which contains even a single graph with at least $t+2$ vertices and which is not $(t+1)$-connected implies key agreement. As discussed in Introduction, this result does not rely on the topology-hiding aspect of the broadcast protocol, but on the *anonymity* aspect, and in fact, it can be derived by combining ideas from Ishai et al. [22] and Ball et al. [4]. We choose to present the result in this form for completeness and because it provides a matching lower bound to Theorem 7.1.

Proposition 5.1. *Let t be an integer, and let $\mathcal{G}$ be a class containing a graph with at least $t+2$ vertices which is not $(t+1)$-vertex-connected. Then, t-IND-CTA anonymous broadcast with respect to $\mathcal{G}$ implies the existence of key agreement.*

Proof sketch. We prove the proposition for the case of $t = 1$, i.e., for the singleton class $\mathcal{G}_{\{1\text{-}2\text{-}3\}} = \{①\text{-}②\text{-}③\}$. The general case follows via a standard player-partitioning argument (in a similar way to [4, Cor. 3.4]).

The key-agreement protocol follows in a similar way to Sect. 4.2 (the triangle graph). Alice simulates nodes ①-②, while Bob simulates ③. Alice and Bob simulate two instances of π, where each party randomly chooses whether to simulate the anonymous broadcaster in the first instance or in the second. The protocol is formally described in Fig. 19. The proof follows in the same lines as the proof of Lemma 4.5 (see also [4]), since the transcript that Eve can see forms a partial view of node ②; therefore, security of the KA protocol is reduced to the 1-IND-CTA security of π. □

5.2. *Uncertain Honest Majority Requires io-OT (The 2-vs-3 Paths)*

In the previous section, we showed that, for a large number of graph classes, key agreement is *necessary* to achieve 1-THAB. A natural follow-up question is to ask whether key agreement is sufficient to achieve 1-THAB *on all graphs* or not. We answer this question in the negative (at least in a black-box way) by showing that constant-round 1-THAB on the class of paths of length two and three implies *infinitely often oblivious transfer*.[24]

This result may be reminiscent to the result of Ball et al. [3] who showed that without assuming an honest majority, THB on arbitrary graphs implies the existence of oblivious

[24] Recall that although KA can be constructed from OT in a black-box way, OT *cannot* be constructed from KA in a black-box way [18].

Key-Agreement Protocol

Public Parameters: An anonymous broadcast protocol π for $\mathcal{G}_{\{1\text{-}2\text{-}3\}}$.

The Protocol:

- `Initialization:` Alice samples two random strings $m_1, m_2 \leftarrow \{0,1\}^\lambda$ and sends them to Bob.
- Alice locally flips a coin $A \leftarrow \{0,1\}$; similarly, Bob locally flips a coin $B \leftarrow \{0,1\}$. Next, they jointly simulate two instances of π, communicating the messages between ② and ③. Alice simulates ①-② and Bob simulates ③.
  - If $A = 1$, Alice will simulate ① broadcasting m_1 in the first execution (but not broadcast in the second), and if $A = 0$ she simulates ① broadcasting m_2 in the second execution (but not broadcast in the first).
  - If $B = 1$, Bob simulates ③ broadcasting m_1 in the first execution (but not broadcast in the second), and if $B = 0$ he simulates ③ broadcasting m_2 in the second execution (but not broadcast in the first).
- `Output:` If ① (resp., ③) outputs m_1 in the first instance of π and m_2 in the second, Alice (resp., Bob) outputs A (resp., B); otherwise, outputs $\perp$.

Fig. 19. Key agreement from 1-THAB on $\mathcal{G}_{\{1\text{-}2\text{-}3\}}$.

transfer. We note that our result requires inherently different techniques, as in the one-corruption setting there exists only one graph with no honest majority (the path of length 2), and 1-THAB on this graph is trivial. However, considering in addition the path of length 3 (where an honest majority is guaranteed), we prove an implication to infinitely often oblivious transfer (io-OT).

Note that this lower bound *only* applies to anonymous broadcast. In fact, even simple flooding gives secure THB on the path of length 2 and the path of length 3, because given the identity of the broadcaster every node can trivially derive its distance from the broadcaster by its local view.

Before stating the result, we need to recall the definition of io-OT and define the class of graphs we will be working with.

5.2.1. *Infinitely Often Oblivious Transfer*

We start by recalling the definition of uniform infinitely often security for deterministic two-party functionalities by Lindell et al. [[28], Def. 2.3].

Definition 5.2. (Uniform infinitely often security) A protocol π securely computes a deterministic functionality $\mathcal{F}$ in the presence of semi-honest adversaries with uniform infinitely often security if there exists an infinite subset $I \subseteq \mathbb{N}$ such that:

- *Correctness* There exists a negligible function such that for every $\lambda \in I$ and every pair of inputs $x, y \in \{0,1\}^*$ it holds that

$$\Pr\left[OUTPUT^{\pi}(x, y, \lambda) = \mathcal{F}(x, y)\right] \geq 1 - \mathsf{negl}(\lambda),$$

  where the probability is taken over the random coins of the parties, and where $OUTPUT^{\pi}(x, y, \lambda)$ is the random variable denoting the output of both parties in an honest execution on inputs x and y and with security parameter λ.

- *Privacy* There exist two PPT algorithms $\mathcal{S}_{\mathsf{Alice}}$ and $\mathcal{S}_{\mathsf{Bob}}$ (called "simulators"), such that:

$$\left\{\mathcal{S}_{\mathsf{Alice}}(x, \mathcal{F}_{\mathsf{Alice}}(x, y), 1^\lambda)\right\}_{x,y\in\{0,1\}^*,\lambda\in I} \approx_c \left\{VIEW^{\pi}_{\mathsf{Alice}}(x, y, \lambda)\right\}_{x,y\in\{0,1\}^*,\lambda\in I}$$

and

$$\left\{\mathcal{S}_{\mathsf{Bob}}(y, \mathcal{F}_{\mathsf{Bob}}(x, y), 1^\lambda)\right\}_{x,y\in\{0,1\}^*,\lambda\in I} \approx_c \left\{VIEW^{\pi}_{\mathsf{Bob}}(x, y, \lambda)\right\}_{x,y\in\{0,1\}^*,\lambda\in I},$$

where $\mathcal{F}_{\mathsf{Alice}}$ and $\mathcal{F}_{\mathsf{Bob}}$ correspond to the output of the functionality $\mathcal{F}$ to Alice and Bob, respectively.

We can now define infinitely often oblivious transfer (io-OT), building on the 1-out-of-2 OT functionality defined as $\mathcal{F}_{\mathsf{OT}}((m_0, m_1), b) = (\epsilon, m_b)$, where ϵ denotes the empty string.

Definition 5.3. (io-OT) A protocol π is an *infinitely often oblivious transfer* protocol, if π computes $\mathcal{F}_{\mathsf{OT}}$ in the presence of semi-honest adversaries with uniform infinitely often security.

5.2.2. *The Lower Bound*

We define the class of all paths of length two or three labeled with $\{1, 2, 3\}$ in cyclic order as

$$\mathcal{G}_{\text{2-vs-3}} = \left\{①\text{-}②,\ ②\text{-}③,\ ③\text{-}①,\ ①\text{-}②\text{-}③,\ ②\text{-}③\text{-}①,\ ③\text{-}①\text{-}②\right\}.$$

Theorem 5.4. *Let π be a constant-round 1-IND-CTA anonymous broadcast with respect to $\mathcal{G}_{\text{2-vs-3}}$. Then, there exists a uniform io-OT protocol secure in the presence of a semi-honest adversary.*

In Sect. 1.2.2, we gave a high-level overview of the proof of Theorem 5.4. Recall that the proof in the simple case where π is a 2-round protocol relied on the following properties:

- π is 1-IND-CTA anonymous broadcast with respect to $\mathcal{G} = \{②\text{-}③,\ ①\text{-}②\text{-}③,\ ②\text{-}③\text{-}①\}$.
- π is not 1-IND-CTA anonymous broadcast with respect to $\mathcal{G} \cup \{①\text{-}②\text{-}③\text{-}①\}$.

To mirror this in the general case, we have to find a path $\boldsymbol{P}^* = (u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*_R$, where u_i^* is a single node for $i \in \{1, 2, 3\}$ and $\boldsymbol{P}^*_R$ consists of the rest of the nodes, such that

- π is 1-IND-CTA anonymous broadcast with respect to

$$\mathcal{G} = \left\{(u_2^*)\text{-}(u_3^*),\ (u_1^*)\text{-}(u_2^*)\text{-}(u_3^*),\ (u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}^*_R\right\}.$$

- π is not 1-IND-CTA anonymous broadcast with respect to $\mathcal{G} \cup \{\boldsymbol{P}^*\}$.

In order to find such a path we perform an over-extension argument, observing that any protocol π that is 1-IND-CTA with respect to $\mathcal{G}_{\text{2-vs-3}}$ either breaks down on $\boldsymbol{P}^*$ or is still secure and thus can be extended to include $\boldsymbol{P}^*$ in $\mathcal{G}$.

More precisely, assume we start with $\mathcal{G} = \{$②-③, ①-②-③, ②-③-①$\}$. Then, either π breaks down on $\boldsymbol{P}^* =$ ①-②-③-① and we are done, or we use an extension argument as follows. We set the new graph class to be $\mathcal{G} = \{$①-②, ③-①-②, ①-②-③-①$\}$ and obtain a new $\boldsymbol{P}^* =$ ③-①-②-③-①. As we started with a protocol π that is 1-IND-CTA with respect to $\mathcal{G}_{\text{2-vs-3}}$, by assumption π is indeed 1-IND-CTA with respect to the new graph class $\mathcal{G}$. Now, either π breaks down on $\boldsymbol{P}^* =$ ③-①-②-③-①, or we can extend the argument, starting with $\mathcal{G} = \{$③-①, ②-③-①, ③-①-②-③-①$\}$ and $\boldsymbol{P}^* =$ ②-③-①-②-③-①.

Note that for this argument to go through, it is crucial that $\mathcal{G}_{\text{2-vs-3}}$ comprises all possible paths with cyclic labeling of length 2 and 3, as with the extension argument the paths that we use in $\mathcal{G}$ change. Further, note that in order to argue that π has to eventually break down, we need that π has constant round complexity and thus breaks down as soon as the length of the path on which π is run exceeds the round complexity of π.

Proof of Theorem 5.4. We start by defining the class $\mathcal{G}_\ell$ of all paths of length ℓ with cyclic labeling from $\{1, 2, 3\}$ as follows: For a path $\boldsymbol{P}$ and an integer $a \in \mathbb{N}$ denote by $(\boldsymbol{P})^a$, the concatenation of a copies of $\boldsymbol{P}$. We use this notation to extend $\mathcal{G}_{\text{2-vs-3}}$ to longer paths by defining for every $\ell = 3a + b \in \mathbb{N}$

- $\mathcal{G}_\ell = \{$(①-②-③)^a, (②-③-①)^a, (③-①-②)$^a\}$ if $b = 0$,
- $\mathcal{G}_\ell = \{$(①-②-③)^a-①, (②-③-①)^a-②, (③-①-②)^a-③$\}$ if $b = 1$, and
- $\mathcal{G}_\ell = \{$(①-②-③)^a-①-②, (②-③-①)^a-②-③, (③-①-②)^a-③-①$\}$ if $b = 2$.

Note that as for $\ell \geq 4$ labels do not point to a unique nodes anymore; therefore, it is not sufficient to simply describe a node by its label in $\{1, 2, 3\}$. Given a node $v \in V(\boldsymbol{P})$, the neighborhood $\mathcal{N}_{\boldsymbol{P}}(v) \subseteq \{1, 2, 3\}$ refers to *only* the labels of the neighbors of v, which—together with its own label—correspond to the *local* view of v. □

Claim 5.5. *Let $R \in \mathbb{N}$ be a constant, and let π be an R-round 1-IND-CTA anonymous broadcast (AB) with respect to $\mathcal{G}_{\text{2-vs-3}}$. Then, there exists an integer $\ell^* \leq R + 1$ such that*

1. *π is 1-IND-CTA secure AB protocol with respect to $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell^*}$, where the broadcaster is always the left-most node on a path.*[25]
2. *π is not 1-IND-CTA secure AB protocol with respect to $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell^*+1}$, where the broadcaster is always the left-most node on a path.*

Proof. For each $\ell \in \mathbb{N}$, we have:

- *either* π is 1-IND-CTA secure AB protocol with respect to $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_\ell$ (with left-most broadcaster) ($\star$)

[25] Note that the cyclic labeling implicitly defines an orientation of the path, allowing to talk about "left" and "right"

- *or* π is not 1-IND-CTA secure AB protocol with respect to $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell}$ (with left-most broadcaster) ($\star\star$).

We set $\ell^* \in \mathbb{N}$ to be the minimal number such that ($\star$) holds for ℓ^*, but ($\star\star$) holds for $\ell^* + 1$. By assumption, we have that π is 1-IND-CTA with respect to $\mathcal{G}_{\text{2-vs-3}}$, which implies that ($\star$) holds for $\ell = 3$. On the other hand, we know that for $\ell = R + 2$ the message cannot reach from the left-most node to the right-most node in R rounds and we thus have a distinguisher with success probability $\geq 1/2$. This implies $\ell^* + 1 \leq R + 2$ as required. □

Note that by Item 2 of Claim 5.5 there exists a path $\boldsymbol{P}^* \in \mathcal{G}_{\ell^*+1}$ that can be efficiently distinguished from all paths in $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell^*}$. In particular, when parsing $\boldsymbol{P}^* = (u_1^*)\text{-}(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}_R^*$, where u_i^* is a single node for $i \in \{1, 2, 3\}$ and $\boldsymbol{P}_R^*$ consists of $\ell^* - 2$ nodes, we have that $\boldsymbol{P}^*$ can be efficiently distinguished from $(u_2^*)\text{-}(u_3^*)$, $(u_1^*)\text{-}(u_2^*)\text{-}(u_3^*) \in \mathcal{G}_{\text{2-vs-3}}$ and $(u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}_R^* \in \mathcal{G}_{\ell^*}$ as required. In order to simplify the description of the AND protocol in the following we assume that the distinguisher in fact distinguishes between a run of π on $\boldsymbol{P} = (u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}_R^*$ and $\boldsymbol{P}^*$.

More precisely, there exists a constant $c_D \in \mathbb{N}$, a distinguisher D running in time at most λ^{c_D}, an infinite set $I \subseteq \mathbb{N}$, and a node $v^* \in V(\boldsymbol{P}) \cap V(\boldsymbol{P}^*) = V((u_2^*)\text{-}(u_3^*)\text{-}\boldsymbol{P}_R^*)$ such that for all $\lambda \in I$ it holds that:

$$\left|\Pr\left[D\left(1^\lambda, m, VIEW_{v^*}^{\boldsymbol{P}^*}(\lambda, m)\right) = 1\right] - \Pr\left[D\left(1^\lambda, m, VIEW_{v^*}^{\boldsymbol{P}}(\lambda, m)\right) = 1\right]\right| \geq \lambda^{-c_D},$$

where the randomness is taken over the random coins of π, of D, and the choice of m, and where $VIEW_v^{\boldsymbol{P}}(\lambda, m)$ denotes the view of node v in an execution of π on the path $\boldsymbol{P}$ on input m and security parameter λ.

Given this distinguisher, we can construct a protocol (as we show below) for securely computing the AND of two bits $x \wedge y$ with infinitely often security. By Kilian [24] (see also Lindell [28]) this implies io-OT as required. The rest of the proof is thus dedicated to the construction and proof of security of the protocol for AND.

The high-level idea of our protocol is as follows. In the first step, the parties establish a path $\boldsymbol{P}$ as follows. Alice will add one node to $\boldsymbol{P}$ if $x = 0$, and two nodes if $x = 1$. Bob will add one node to $\boldsymbol{P}$ if $y = 0$, and $\ell^* - 1$ nodes if $y = 1$. Now, the parties can use the distinguisher D (as implied by Item 2 of Claim 5.5) to distinguish the case of $x \wedge y = 0$ (where $\boldsymbol{P} \in \mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell^*}$) from the case $x \wedge y = 1$ (where $\boldsymbol{P} \in \mathcal{G}_{\ell^*+1}$). See Fig. 20 for an illustration.

Before formally describing the AND protocol, we have yet to deal with one technical issue (in a similar way to Sect. 4.1). Namely, let

$$\rho_D(\lambda) := \Pr\left[D\left(1^\lambda, m, VIEW_{v^*}^{\boldsymbol{P}^*}(\lambda, m)\right) = 1\right]$$

be the probability of the distinguisher D successfully recognizing the "over-extended" path $\boldsymbol{P}^*$. In order for the parties to distinguish the two cases, they need to know $\rho_D(\lambda)$, but ρ_D might not be efficiently computable.

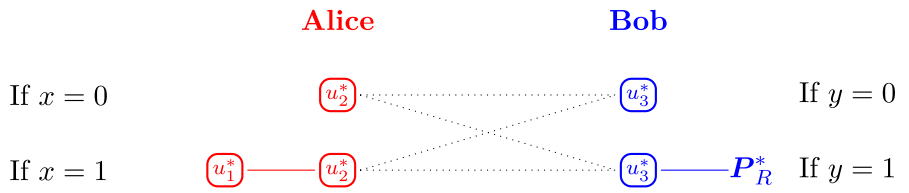


Fig. 20. The possible graphs simulated in the Boolean AND protocol (Fig. 21). Color indicate who is responsible for simulation: Alice - Bob.

This can be dealt with by running the distinguisher λ^{2c_D+1} times on $\boldsymbol{P}^*$, each time with a fresh random message and fresh coins, and let $\sigma = \sigma(\lambda)$ denote the number of times on which the algorithm D outputs 1 in λ^{2c_D+1} runs. Then, we obtain an approximation of ρ_D by setting $\tilde{\rho}_D(\lambda) := \sigma/\lambda^{2c_D+1}$. Hoeffding's inequality yields that

$$\begin{aligned}\Pr\left[\rho_D(\lambda) - \tilde{\rho}_D(\lambda) > \lambda^{-c_D}/4\right] &= \Pr\left[\sigma < (\rho_D(\lambda) - \lambda^{-c_D}/4) \cdot \lambda^{2c_D+1}\right] \\ &< e^{-2\cdot(\lambda^{-c_D}/4)^2\cdot\lambda^{2c_D+1}} < e^{-\lambda/8}.\end{aligned}$$

On the other hand, we have

$$\Pr\left[\tilde{\rho}_D(\lambda) - \rho_D(\lambda) > \lambda^{-c_D}/4\right] = \Pr\left[\sigma > (\rho_D(\lambda) + \lambda^{-c_D}/4) \cdot \lambda^{2c_D+1}\right] < e^{-\lambda/8}.$$

and thus $|\rho_D(\lambda) - \tilde{\rho}_D(\lambda)| \leq \lambda^{-c_D}/4$ except with probability negligible in λ.

With this we are ready to give a full description of the protocol in Fig. 21.

Lemma 5.6. *Let π be a constant-round 1-IND-CTA anonymous broadcast with respect to $\mathcal{G}_{\text{2-vs-3}}$. Then, the protocol in Fig. 21 securely realizes the Boolean AND functionality with uniform infinitely often security in the presence of a semi-honest adversary.*

Proof. We prove correctness and security separately.

Correctness We first show that for all $\lambda \in I$ the protocol yields the correct output with overwhelming probability in the case that both Alice and Bob follow the protocol. Recall that the probability that D outputs 1 on the "over-extended" path (i.e., in case $x \wedge y = 1$) is $\rho_D(\lambda)$ and the probability that D outputs 1 on a "short" path (i.e., in case $x \wedge y = 0$) is at most $\rho_D(\lambda) - \lambda^{-c_D}$ for all $\lambda \in I$. Let B denote the number of runs for which D outputs 1. Applying Hoeffding's inequality and $\tilde{\rho}_D \leq \rho_D + \lambda^{-c_D}/4$ yields

$$\begin{aligned}&\Pr\left[\text{Bob outputs } 0 \mid x \wedge y = 1\right] \\ &\leq \Pr\left[B < (\tilde{\rho}_D(\lambda) - \lambda^{-c_D}/2) \cdot \lambda^{2c_D+1} \mid x \wedge y = 1\right] \\ &\leq \Pr\left[B < (\rho_D(\lambda) - \lambda^{-c_D}/4) \cdot \lambda^{2c_D+1} \mid x \wedge y = 1\right] < e^{-\lambda/8}.\end{aligned}$$

Protocol $\mathsf{AND}^{D,c_D}(1^\lambda)$

Public Parameters: The security parameter λ, an anonymous broadcast protocol π for $\mathcal{G}_{\text{2-vs-3}}$, an integer ℓ^*, a path $\boldsymbol{P}^* \in \mathcal{G}_{\ell^*+1}$ of the form $\boldsymbol{P}^* = (u_1^*)$-$(u_2^*)$-$(u_3^*)$-$\boldsymbol{P}_R^*$ (where $\boldsymbol{P}_R^*$ is a path of length $\ell^* - 2$), a node $v^* \in V((u_2^*)$-(u_3^*)-$\boldsymbol{P}_R^*)$, and a distinguisher D that runs in time $O(\lambda^{c_D})$ (assumed to have advantage λ^{-c_D}).

Private Inputs: Alice holds an input bit $x \in \{0,1\}$ and Bob holds an input bit $y \in \{0,1\}$.

The Protocol:

- `Initialization:`
  - For each $i \in [\lambda^{2c_D+1}]$, the party in control of v^* (i.e., Alice if $v^* = (u_2^*)$, and Bob otherwise) samples X_i according to $\text{VIEW}_{v^*}^{\boldsymbol{P}^*}(\lambda, \tilde{m}_i)$ where $\tilde{m}_i \leftarrow \{0,1\}$ is drawn uniformly at random. Let
$$\tilde{\rho}_D(\lambda) := \frac{\sum_{i=1}^{\lambda^{2c_D+1}} D(1^\lambda, \tilde{m}_i, X_i)}{\lambda^{2c_D+1}}.$$
- `Setting the underlying graphs:`
  - Alice sets $G_A = (u_2^*)$ if $x = 0$, and $G_A = (u_1^*)$-(u_2^*) if $x = 1$.
  - Bob sets $G_B = (u_3^*)$ if $y = 0$, and $G_B = (u_3^*)$-$\boldsymbol{P}_R^*$ if $y = 1$.
- `Running the anonymous broadcast protocol:` For $i = 1, \ldots, \lambda^{2c_D+1}$,
  - Alice chooses a message $m_i \leftarrow \{0,1\}$ and sends m_i to Bob.
  - Alice and Bob jointly simulate $\pi(1^\lambda)$ on the path $\boldsymbol{P} = G_A$-G_B with Alice simulating her "left-most" node as broadcasting message m_i.
  - The party in control of v^* runs D on the view of node v^*, i.e., $d_i \leftarrow D(1^\lambda, m_i, \text{VIEW}_{v^*}^{\boldsymbol{P}}(\lambda, m_i))$. (If v^* does not exist on the path $\boldsymbol{P}$, because the input of the party who should be in control of v^* is 0, the party simply sets $d_i = 0$ for all i.)
- `Output:`
  - If $\sum d_i \geq \tilde{\rho}_D(\lambda) \cdot \lambda^{2c_D+1} - \lambda^{c_D}/2$, the party in control of v^* sends $b = 1$ to the other party; otherwise, it sends $b = 0$.
  - Both parties output the bit b.

Fig. 21. Infinitely often Boolean AND protocol.

On the other hand, applying Hoeffding's inequality and $\tilde{\rho}_D \geq \rho_D - \lambda^{-c_D}/4$ yields

$$\begin{aligned}
&\Pr\Big[\text{Bob outputs } 1 \mid x \wedge y = 0\Big] \\
&\quad \leq \Pr\Big[B \geq (\tilde{\rho}_D(\lambda) - \lambda^{-c_D} + \lambda^{-c_D}/2) \cdot \lambda^{2c_D+1} \mid x \wedge y = 0\Big] \\
&\quad \leq \Pr\Big[B \geq (\rho_D(\lambda) - \lambda^{-c_D} + \lambda^{-c_D}/4) \cdot \lambda^{2c_D+1} \mid x \wedge y = 0\Big] \leq e^{-\lambda/8}
\end{aligned}$$

for all $\lambda \in I$.

Security First, assume that Alice is corrupt. The simulator is given the input x by the environment, forwards x to the AND functionality and receives the output bit $b = x \wedge y$. It simulates the sub-graph G_B of $\boldsymbol{P}$ set by Bob as follows. If $b = 0$, it adds one node (u_3^*); otherwise, it adds $\ell^* - 1$ nodes (u_3^*)-$\boldsymbol{P}_R^*$. To simulate the executions of the protocol π (which can be executed independently of the parties' inputs) the simulator imitates the parties' behavior in a real protocol execution.

Now, assume Bob is corrupt. The simulator proceeds exactly as above, but simulates the sub-graph G_A of $\boldsymbol{P}$ set by Alice as follows. If $b = 0$, it adds one node (u_2^*); otherwise, it adds two nodes (u_1^*)-(u_2^*).

It is left to show that the simulated run is indistinguishable from a real protocol execution. Note that if $b = x \wedge y = 1$, then the simulator behaves *exactly* according to a real protocol execution. We thus only have to consider the case $b = 0$. In this case, the adversary is only in control of one node and the security reduces to the security of π.

More precisely, we have to show that

$$\left\{VIEW_{u_2^*}^{\boldsymbol{P}_0}(\lambda, m_i)\right\}_{i\in[\lambda^{2c_D+1}]} \approx_c \left\{VIEW_{u_2^*}^{\boldsymbol{P}_1}(\lambda, m_i)\right\}_{i\in[\lambda^{2c_D+1}]},$$

where $\boldsymbol{P}_0 = (u_2^*)$-$(u_3^*)$ and $\boldsymbol{P}_1 = (u_2^*)$-$(u_3^*)$-$\boldsymbol{P}_R^*$ for the case that Alice is corrupt, and

$$\left\{VIEW_{u_3^*}^{\boldsymbol{P}_0}(\lambda, m_i)\right\}_{i\in[\lambda^{2c_D+1}]} \approx_c \left\{VIEW_{u_3^*}^{\boldsymbol{P}_1}(\lambda, m_i)\right\}_{i\in[\lambda^{2c_D+1}]},$$

where $\boldsymbol{P}_0 = (u_2^*)$-$(u_3^*)$ and $\boldsymbol{P}_1 = (u_1^*)$-$(u_2^*)$-$(u_3^*)$ for the case that Bob is corrupt.

Since all executions of π are independent of each other, this follows using the 1-IND-CTA security of π on $\mathcal{G}_{\text{2-vs-3}} \cup \mathcal{G}_{\ell*}$ and Lemma 2.8.

This concludes the proof of Lemma 5.6. □

This concludes the proof of Theorem 5.4. □

6. Information-Theoretic Upper Bounds

In this section, we present our information-theoretic constructions: In Sect. 6.1, we present 1-IT-THAB for 2-connected graphs, and in Sect. 6.2, 1-IT-THB for the 1-connected butterfly graph.

6.1. *2-Connectivity is Sufficient for 1-IT-THAB*

We start by showing that 1-IT-THAB is possible on *all* 2-vertex-connected graphs. Our protocol requires communication scaling polynomially in the number of graphs in the class and computation additionally scaling exponentially in the maximal degree (i.e., the maximal number of neighbors of a node in any graph of the class). In fact, we show how to establish a stronger notion, namely 1-IT topology-hiding *anonymous secure channels*, where each party can anonymously send a message to each other party without anyone else learning the content of the message and without leaking anything about the topology. As we show in Sect. 8.1, when the number of parties is fixed and known this communication network can be used to run the BGW protocol, and support 1-IT-THC.

We refer the reader to Sect. 1.2.3 for a high-level overview of the *censored brute force* technique that is used in this section. In Sect. 6.1.1, we establish the required

The functionality $\mathcal{F}_{\mathsf{sum}}(\ell, \mathsf{P}_\tau)$

The sum functionality $\mathcal{F}_{\mathsf{sum}}(\mathsf{P}_\tau)$ is parametrized by the target party P_τ and the input length $\ell \in \mathbb{N}$, and proceeds as follows.

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

Input: Record an input message $m_u \in \{0,1\}^\ell$ from each P_u.

Output: Let $m = \bigoplus m_u$ be the sum of the inputs of every party P_u that is in the same connected component as P_τ in G and send m to P_τ. All other parties receive no output.

Fig. 22. The sum functionality with a single known receiver.

definitions and terminology for the protocol; in Sect. 6.1.2 we state the main technical lemma (Lemma 6.4) and its corollaries; and finally, in Sect. 6.1.3 we prove Lemma 6.4.

6.1.1. *Definitions and Notations*

The sum functionality The protocol presented in this section will realize the sum functionality $\mathcal{F}_{\mathsf{sum}}(\mathsf{P}_\tau)$, formally described in Fig. 22. This functionality takes an input message from each node and outputs the sum of all messages to the designated receiver P_τ. Note that this in particular implies topology-hiding anonymous communication between an anonymous sender and P_τ, by having the sender enter its message and all other parties enter zero as their input.

The protocol we present in Fig. 23 for realizing $\mathcal{F}_{\mathsf{sum}}(\ell, \mathsf{P}_\tau)$ provides *perfect privacy* but it has a positive *correctness error* that can be made arbitrarily small. Stated differently, for adequate parameters that guarantee a negligible correctness error, the protocol provides *statistical* security. We choose to state the slightly stronger form of achieving *perfect* security albeit for a functionality that may err with small probability.

Definition 6.1. ($(1-\delta)$-correct functionality) Let $\mathcal{F}$ be a functionality and let $\delta > 0$ be an arbitrarily small positive number. The $(1-\delta)$-correct functionality $\mathcal{F}^\delta$ proceeds just like $\mathcal{F}$, except that the functionality initially tosses a biased coin that outputs 0 with probability δ, in which case the functionality outputs $\perp$ as the output value for each output party.

A protocol π securely realizes $\mathcal{F}$ with $(1-\delta)$-correctness if π realizes $\mathcal{F}^\delta$ with perfect security.

Bipolar orientation Our protocol relies on an agreed upon *bipolar graph orientation* for each graph in the class. We define bipolar graph orientations in a manner that is convenient for situations where only an upper bound on the number of nodes is known.

Definition 6.2. (Bipolar graph orientation) Let $n \in \mathbb{N}$, let H be an undirected graph with $V(H) \subseteq [n]$, and let $\sigma, \tau \in V(H)$. A $\sigma\tau$-orientation $\langle H \rangle_\tau$ of H is a directed acyclic graph with a unique sink τ and a unique source σ formed by assigning a direction to each edge in H. A $\sigma\tau$-numbering of H is a topological ordering of a $\sigma\tau$-orientation of H, i.e., a map $\psi_{H,\tau} : V(H) \to [n]$ such that

1. $\psi_{H,\tau}$ is injective.

2. $\psi_{H,\tau}(\sigma) = 1$.
3. $\psi_{H,\tau}(\tau) = n$.
4. There exists a $\sigma\tau$-orientation $\langle H \rangle_\tau$ of H, such that $(u, v) \in E(\langle H \rangle_\tau) \implies \psi(u) < \psi(v)$.

Proposition 6.3. *([16,17,32]) For any graph H and vertices $\sigma, \tau \in V(H)$, a $\sigma\tau$-orientation and a $\sigma\tau$-numbering can be found in polynomial time (in n).*

6.1.2. *Topology-Hiding Communication for 2-Connected Graphs*

We now present our main technical lemma of the section: a protocol for topology-hiding computation of the sum functionality with $(1 - \delta)$-correctness against one computationally unbounded semi-honest corruption with respect to 2-connected graphs. Before proving the lemma in Sect. 6.1.3, we present corollaries to THAB and THC.

In the results stated below we denote the class of 2-connected graphs with *up to n* nodes as

$$\mathcal{G}^{\leq n}_{\text{2-conn}} = \{G \text{ graph } : V(G) \subseteq [n] \text{ and G is 2-connected}\},$$

and the class of 2-connected graphs with *exactly n* nodes as

$$\mathcal{G}^{n}_{\text{2-conn}} = \{G \text{ graph } : V(G) = [n] \text{ and G is 2-connected}\}.$$

Lemma 6.4. *Let $n \in \mathbb{N}$, let $\mathcal{G} \subseteq \mathcal{G}^{\leq n}_{\text{2-conn}}$, let d_{max} be the maximal degree of any graph in $\mathcal{G}$ ($d_{max} \leq n$), let $\ell \in \mathbb{N}$, let $\tau \in [n]$, and let $\delta > 0$. Then, protocol $\pi^{\delta}_{\mathsf{sum}}(\ell, \mathcal{G}, \mathsf{P}_\tau)$ (defined in Fig. 23) securely realizes $\mathcal{F}_{\mathsf{sum}}(\ell, \mathsf{P}_\tau)$ with $(1-\delta)$-correctness in a topology-hiding manner with respect to $\mathcal{G}$, tolerating a single semi-honest corruption.*

Moreover, $\pi^{\delta}_{\mathsf{sum}}(\ell, \mathcal{G}, \mathsf{P}_\tau)$ completes within n rounds with total communication complexity

$$O\Big(n \cdot d_{max} \cdot |\mathcal{G}| \cdot (\ell + \log(1/\delta) + d_{max} \cdot \log |\mathcal{G}|)\Big)$$

and computation complexity $O(|\mathcal{G}|^{d_{max}})$.

Note that a protocol for $\mathcal{F}_{\mathsf{sum}}$ can be used to construct private channels with sender-anonymity by having the sender enter its input to the protocol and every other party enter zero. In turn, this enables 1-THAB by having the broadcaster send its message independently to every potential receiver.

Theorem 6.5. *Let $n \in \mathbb{N}$, let $\mathcal{G} \subseteq \mathcal{G}^{\leq n}_{\text{2-conn}}$, let d_{max} is the maximal degree of any graph in $\mathcal{G}$, and let $\delta > 0$. Then, there exists a protocol that securely realizes $\mathcal{F}_{\text{anon-bc}}$ with $(1 - \delta)$-correctness in a topology-hiding manner with respect to $\mathcal{G}$, tolerating a single semi-honest corruption.*

Moreover, the protocol completes within n rounds with total communication complexity

$$O\Big(n^2 \cdot d_{max} \cdot |\mathcal{G}| \cdot (\ell + \log(n/\delta) + d_{max} \cdot \log|\mathcal{G}|)\Big)$$

and computation complexity $O(|\mathcal{G}|^{d_{max}})$ *per node.*

Proof (sketch). For $\delta > 0$, We define a $(1-\delta)$-correct 1-THAB protocol in the $\mathcal{F}_{\mathsf{sum}}^{\delta/n}$-hybrid model, by invoking the functionality n times (in parallel) with $(1-\delta/n)$-correctness, where in the i^{th} invocation, for each $i \in [n]$, the target party is P_i, and each party enters zero as its input to $\mathcal{F}_{\mathsf{sum}}^{\delta/n}(\ell, \mathsf{P}_i)$ except for the broadcaster who enters its input. The proof immediately follows from Lemma 6.4. □

Further, by using the sum functionality to construct secure channels between every pair of parties, we get the communication network needed to run the semi-honest BGW protocol [8]. Since the BGW protocol works for a fixed and known set of parties under an honest-majority assumption, we get $(1-\delta)$-correct 1-THC for 2-connected graphs on *exactly* n nodes, for $n \geq 3$.

Theorem 6.6. *Let* $n \geq 3$*, let* $\mathcal{G} \subseteq \mathcal{G}^n_{\text{2-conn}}$*, let* $\delta > 0$*, and let* f *be an* n*-party function. Then, there exists a protocol that securely realizes* $\mathcal{F}_{\mathsf{sfe}}^{f,\delta}$ *with perfect security in a topology-hiding manner with respect to* $\mathcal{G}$*, tolerating a single semi-honest corruption.*

6.1.3. *Security Analysis*

Proof of Lemma 6.4. The proof works by considering separately the messages associated with each graph in $\mathcal{G}$. With that in mind, let $H \in \mathcal{G}$ denote an arbitrary graph and let $G \in \mathcal{G}$ denote the actual graph used by $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ (i.e., in the protocol it holds that $\mathcal{N}(u) = \mathcal{N}_G(u)$). Denote the set of messages associated with H in the protocol $\pi_{\mathsf{sum}}^{\delta}(\ell, \mathcal{G}, \mathsf{P}_\tau)$ (Fig. 23) as,

$$\mathcal{M}_H = \left\{ s_H^{u \to v} \in \{0,1\}^{\ell+\kappa} : (u,v) \in E(\langle H \rangle_\tau) \cap E(G) \right\}.$$

In round i, messages are sent to the i^{th} node (numbered according to $\psi_{H,\tau}$) by each of its predecessors and that node prepares its message for each of its successors. For $i \in [n-1]$ consider

$$\mathcal{S}^i = \left\{ s_H^{u \to v} \in \mathcal{M}_H : \psi_{H,\tau}(u) \leq i \ \wedge \ \psi_{H,\tau}(v) > i \right\}.$$

In other words, if you order the nodes according to $\psi_{H,\tau}$ and draw a line between nodes i and $i+1$, $\mathcal{S}^i$ constitutes the messages corresponding to edges crossing that line.

Protocol $\pi^{\delta}_{\mathsf{sum}}(\ell, \mathcal{G}, \mathsf{P}_\tau)$

Public Parameters: Let $\mathcal{G} \subseteq \mathcal{G}^{\leq n}_{\text{2-conn}}$ with maximal degree d_{max} and let $\tau \in [n]$. For each $H \in \mathcal{G}$, let $\langle H \rangle_\tau$ and $\psi_{H,\tau}$ denote bipolar orientation and corresponding numbering of H, respectively, where τ is the unique sink (with $\psi_{H,\tau}(\tau) = n$). We take $\mathcal{N}^+_H(u)$ and $\mathcal{N}^-_H(u)$ to denote the successors and predecessors (respectively) of u in $\langle H \rangle_\tau$. Let $\kappa = \log(1/\delta) + d_{\mathsf{max}} \cdot \log(|\mathcal{G}|)$ denote the length of the checksum to be added to ensure correctness error δ.

Hybrid Model: The protocol is defined in the $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$-hybrid model.

The Protocol:

- `Initialization:` Upon receiving the input m_u from the environment, each party P_u initializes $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ to receive its neighborhood $\mathcal{N}(u)$.
- `Round 1:` In this round for each $H \in \mathcal{G}$ the source in $\langle H \rangle_\tau$ (which can be different for each $H \in \mathcal{G}$) initializes the sub-protocol. That is, for every $H \in \mathcal{G}$, if $\psi_{H,\tau}(u) = 1$ (i.e., u is the source in $\langle H \rangle_\tau$), then P_u does:
  - If $\mathcal{N}_H(u) = \mathcal{N}(u)$, then for each $v \in \mathcal{N}^+_H(u)$ sample $s^{u\to v}_H \leftarrow \{0,1\}^{\ell+\kappa}$ uniformly at random conditioned on $\sum_{v \in \mathcal{N}^+_H(u)} s^{u\to v}_H = m_u \| 1^\kappa$.
  - If $\mathcal{N}_H(u) \neq \mathcal{N}(u)$, then for each $v \in \mathcal{N}^+_H(u)$ sample $s^{u\to v}_H \leftarrow \{0,1\}^{\ell+\kappa}$ uniformly at random.

  P_u sends $((H, s^{u\to v}_H))_{H\in\mathcal{G}}$ to each party P_v for each $v \in \mathcal{N}^+_H(u)$.
- `Round` $i = 2, \ldots, n-1$: In the i^{th} round, for each $H \in G$ the party numbered i in $\langle H \rangle_\tau$ will receive a message from all its predecessors in $\langle H \rangle_\tau$ (that are also actual neighbors) and compute its outgoing messages. That is, for every $H \in \mathcal{G}$, let $v_{H,i}$ such that $\psi_{H,\tau}(v_{H,i}) = i$. Then, for each $H \in \mathcal{G}$ party P_u proceeds as follows:
  1. If $u \in \mathcal{N}^-_H(v_{H,i}) \cap \mathcal{N}(v_{H,i})$ (i.e., if u is a predecessor of $v_{H,i}$ in $\langle H \rangle_\tau$ and they are neighbors in the real graph), then P_u sends to $\mathsf{P}_{v_{H,i}}$ the message $(H, s^{u\to v_{H,i}}_H)$. (For this step to be well-defined, note that by the properties of a bipolar numbering for all $u \in N^{-1}_H(v_{H,i})$ we have $\psi_{H,\tau}(u) < i$.)
  2. If $u = v_{H,i}$, receive $(H, s^{v\to u}_H)$ from each $v \in \mathcal{N}^-_H(u) \cap \mathcal{N}(u)$. Set $s^u_H = \sum_{v \in \mathcal{N}^-_H(u)} s^{v\to u}_H$. Then,
     - If $\mathcal{N}_H(u) = \mathcal{N}(u)$, for each $v \in \mathcal{N}^+_H(u)$ sample $s^{u\to v}_H \leftarrow \{0,1\}^{\ell+\kappa}$ uniformly at random conditioned on $\sum_{v \in \mathcal{N}^+_H(u)} s^{u\to v}_H = m_u \| 0^\kappa + s^u_H$.
     - If $\mathcal{N}_H(u) \neq \mathcal{N}(u)$, for each $v \in \mathcal{N}^+_H(u)$ sample $s^{u\to v}_H \leftarrow \{0,1\}^{\ell+\kappa}$ uniformly at random.
- `Round` n: In the final round, for each $H \in \mathcal{G}$ the designated sink τ receives the messages for all possible $H \in \mathcal{G}$ and reconstructs the actual message with the help of the checksum.
  - For each $u \neq \tau$: if $\tau \in \mathcal{N}(u)$, then P_u sends $s^{u\to\tau}_H$ for each H such that $\tau \in N^+_H(u)$ to P_τ in a randomly permuted order.
  - For $u = \tau$: Let $d = |\mathcal{N}(\tau)|$ and $\mathcal{N}(\tau) = \{v_1, \ldots, v_d\}$. Then for each neighbor $v_i \in \mathcal{N}(\tau)$, P_τ receives $(\tilde{s}^{v_i}_1, \ldots, \tilde{s}^{v_i}_{n_{v_i}})$ where $n_{v_i} = |\{H \in \mathcal{G} : \tau \in \mathcal{N}_H(v_i)\}|$. For each $I = (i_1, \ldots, i_d) \in [n_{v_1}] \times \cdots \times [n_{v_d}]$, let $\tilde{s}_I = \sum_{j=1}^d \tilde{s}^{v_j}_{i_j}$.

    If there is a *unique* $\tilde{s}_I \in \{0,1\}^{\ell+\kappa}$ such that $\tilde{s}_I$ is of the form $\tilde{s}_I = x \| 1^\kappa$ for some $x \in \{0,1\}^\ell$ (i.e., a unique value $\tilde{s}_I$ that passes the checksum test), output $x + m_\tau$; otherwise, abort.

Fig. 23. Securely realizing $\mathcal{F}_{\mathsf{sum}}(\mathsf{P}_\tau)$ in a topology-hiding manner, for 2-connected graphs.

Further, we define the messages sent to the i^{th} node in $\langle H \rangle_\tau$ (i.e., node $v_{H,i}$ such that $\psi_{H,\tau}(v_{H,i}) = i$), as

$$\mathcal{S}^-_i = \left\{ s^{u\to v}_H \in \mathcal{M}_H : \psi_{H,\tau}(u) < i \ \wedge \ \psi_{H,\tau}(v) = i \right\} \subseteq \mathcal{S}^{i-1}$$

and the messages sent from $v_{H,i}$ as

$$\mathcal{S}^+_i = \left\{ s^{u\to v}_H \in \mathcal{M}_H : \psi_{H,\tau}(u) = i \ \wedge \ \psi_{H,\tau}(v) > i \right\} \subseteq \mathcal{S}^i.$$

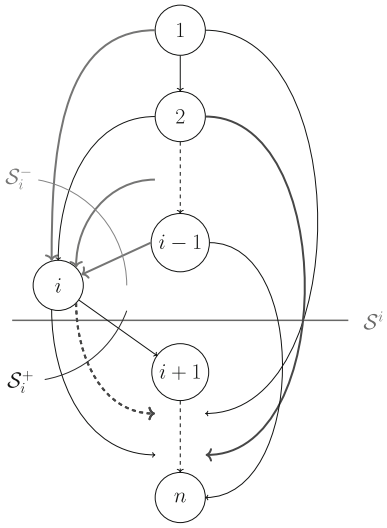


Fig. 24. The sets $\mathcal{S}_i^+$, $\mathcal{S}_i^-$, and $\mathcal{S}^i$ in the intersection of the real graph G and the bipolar orientation of H. Thick arrows are in both G and H while thin arrows are in H but not in G.

Note that this constitutes the local view of $\mathsf{P}_{v_{H,i}}$: the set $\mathcal{S}_i^-$ is received in round i (from the $\langle H \rangle_T$ predecessors in the real graph G), and $\mathcal{S}_i^+$ is sent in subsequent rounds (to the $\langle H \rangle_T$ successors in G). The sets $\mathcal{S}^i$, $\mathcal{S}_i^+$, and $\mathcal{S}_i^-$ are illustrated in Fig. 24.

The following lemma says, among other things, that messages received by the i^{th} intermediate node in $\langle H \rangle_\tau$, $\mathcal{S}_i^-$, are uniformly random. If the graph H is consistent with the real graph G, then τ receives random messages conditioned on them summing to the correct value. And as soon as a neighborhood does not match the real graph, H, everything is uniformly random from that point on. □

Claim 6.7. *Recall that $v_{H,i}$ such that $\psi_{H,\tau}(v_{H,i}) = i$ is the i^{th} node in $\langle H \rangle_\tau$ for $i \in [n]$.*[26] *For each $h \in [n]$, we distinguish between two cases:*

1. *If $\mathcal{N}_H(v_{H,i}) = \mathcal{N}(v_{H,i})$ for all $i \leq h$, then for all $i \leq h$, the set $\mathcal{S}^i$ consists of independently and uniformly distributed strings from $\{0,1\}^{\ell+\kappa}$ such that $\sum_{s \in \mathcal{S}^i} s = (\sum_{j \leq i} m_{v_{H,j}}) \| 1^\kappa$. Moreover for $i \leq h$, it holds that:*
   - *If $i = 1$, the set $\mathcal{S}_i^-$ is empty;*
   - *If $1 < i < n$, $\mathcal{S}_i^-$ is a set of independently and uniformly distributed strings from $\{0,1\}^{\ell+\kappa}$ (Note that $|\mathcal{S}_i^-|$ only depends on local information and the class itself);*
   - *If $i = n$, $\mathcal{S}_i^-$ $(= \mathcal{S}^{n-1})$ is a set of independently and uniformly distributed strings from $\{0,1\}^{\ell+\kappa}$ such that $\sum_{s \in \mathcal{S}_i^-} s = (\sum_{u \in V(H)} m_u) \| 1^\kappa$.*

[26] To account for the case $i \notin \mathrm{Im}(\psi_{H,\tau})$, which can happen if H has less than n nodes, we set $v_{H,i} = 0$, $\mathcal{N}_H(v_{H,i}) = \{0\}$, and $m_{v_{H,i}} = 0$.

2. *If $\mathcal{N}_H(v_{H,j}) \neq \mathcal{N}(v_{H,j})$ for some $j \leq h$, then $\mathcal{S}^j$ is a set of independently and uniformly distributed strings from $\{0,1\}^{\ell+\kappa}$. Moreover for all $j' > h$, $\mathcal{S}^-_{j'}$ is a set of independently and uniformly distributed strings from $\{0,1\}^{\ell+\kappa}$.*

Proof. We prove the claim by induction. By inspection, the claim holds for $h = 1$. The node $v_{H,1}$ prepares $\mathcal{S}^1$ in round 1 in this manner exactly, i.e., uniform conditioned on the sum being $m_{v_{H,1}} \| 1^\kappa$ if $\mathcal{N}_H(v_{H,1}) = \mathcal{N}(v_{H,1})$, and uniform otherwise.

To prove the inductive step, assume the statement holds for some $h \in [n-1]$, and we will prove it for $h+1$. Note that,

$$\sum_{s \in \mathcal{S}^h} s = \sum_{s \in \mathcal{S}^-_{h+1}} s + \sum_{s \in \mathcal{S}^h \setminus \mathcal{S}^-_{h+1}} s \quad \text{and} \quad \sum_{s \in \mathcal{S}^{h+1}} s = \sum_{s \in \mathcal{S}^+_{h+1}} s + \sum_{s \in \mathcal{S}^h \setminus \mathcal{S}^-_{h+1}} s.$$

Therefore,

$$\sum_{s \in \mathcal{S}^{h+1}} s = \sum_{s \in \mathcal{S}^h} s - \sum_{\mathcal{S}^-_{h+1}} s + \sum_{s \in \mathcal{S}^+_{h+1}} s.$$

In addition, note that if $h \notin \mathrm{Im}(\psi_{H,\tau})$, then $\mathcal{S}^h = \mathcal{S}^{h+1}$. So, we will restrict our attention to the opposite case.

Case 1 In this case, $\mathcal{N}_H(v_{H,i}) = \mathcal{N}(v_{H,i})$ for all $i \leq h$. By the inductive hypothesis, $\mathcal{S}^h$ is uniformly random such that $\sum_{s \in \mathcal{S}^h} s = (\sum_{i \leq h} m_{v_{H,i}}) \| 1^\kappa$. Because $\mathcal{N}_H(v_{H,h+1}) = \mathcal{N}(v_{H,h+1})$, the set $\mathcal{S}^+_{h+1}$ consists of independently and uniformly distributed strings conditioned on $\sum_{s \in \mathcal{S}^+_{h+1}} s = m_{v_{H,h+1}} \| 0^\kappa + \sum_{s \in \mathcal{S}^-_{h+1}} s$. Thus, we have that $\mathcal{S}^{h+1}$ consists of independently and uniformly distributed strings conditioned on

$$\sum_{s \in \mathcal{S}^{h+1}} s = \sum_{s \in \mathcal{S}^h} s - \sum_{s \in \mathcal{S}^-_{h+1}} s + \sum_{s \in \mathcal{S}^+_{h+1}} s = \left(\sum_{j \leq h} m_{v_{H,j}} \right) \| 1^\kappa + m_{v_{H,h+1}} \| 0^\kappa$$
$$= \left(\sum_{j \leq h+1} m_{v_{H,j}} \right) \| 1^\kappa.$$

Moreover, because G is 2-connected, there are at least two vertex-disjoint source-sink paths in $\langle G \rangle_\tau$. This means that for $h+1 < n-1$ there is a source-sink path that does not go through $v_{H,h+1}$, and thus $\mathcal{S}^h \setminus \mathcal{S}^-_{h+1}$ is non-empty. Because $\mathcal{S}^h$ is uniform conditioned on summing to some value, it holds that $\mathcal{S}^-_{h+1}$ is uniformly distributed.

Case 2 In this case, $\mathcal{N}_H(v_{H,j}) \neq \mathcal{N}(v_{H,j})$ for some $j \leq h$. If $\mathcal{N}_H(v_{H,h+1}) = \mathcal{N}(v_{H,h+1})$, then it must be the case that $\mathcal{N}_H(v_{H,j}) \neq \mathcal{N}(v_{H,j})$ for some $j \leq h$, in which case $\mathcal{S}^h$ is uniformly distributed by the inductive hypothesis. It follows that $\mathcal{S}^h \setminus \mathcal{S}^-_{h+1}$ and $\mathcal{S}^-_{h+1}$ are comprised of uniform and independent values. (And $\mathcal{S}^-_{h+1}$ is non-empty because H is connected and $h+1$ cannot label a source.) Thus, the set $\mathcal{S}^{h+1}$ (i.e., the disjoint union of $\mathcal{S}^+_{h+1}$ and $\mathcal{S}^h \setminus \mathcal{S}^-_{h+1}$) is comprised of uniformly random values

because $\mathcal{S}^+_{h+1}$ is uniformly random conditioned on summing to $m_{v_{H,h+1}} \| 0^\kappa + \sum_{s \in \mathcal{S}^-_{h+1}} s$, which is a uniformly random value.

If $\mathcal{N}_H(v_{H,h+1}) \neq \mathcal{N}(v_{H,h+1})$, then $\mathcal{S}^+_{h+1}$ consists of (zero or more) uniformly random independent values. As by assumption we have $h+1 \in \mathrm{Im}(\psi_{H,\tau})$ and $v_{H,h+1}$ is not the source, it holds that either $\mathcal{S}^{-1}_{h+1}$ is not empty (i.e., there exists an ingoing edge to $v_{H,h+1}$ in $\langle H \rangle_\tau$), or it must be that $\mathcal{N}_H(v_{H,j}) \neq \mathcal{N}(v_{H,j})$ for some $j \leq h$. In the latter case, $\mathcal{S}^h$ is uniformly random by the inductive hypothesis and thus so is $\mathcal{S}^{h+1} = \mathcal{S}^+_{h+1} \cup (\mathcal{S}^h \backslash \mathcal{S}^-_{h+1}) = \mathcal{S}^+_{h+1} \cup \mathcal{S}^h$.

If $\mathcal{S}^-_{h+1}$ is non-empty, it follows from the inductive hypothesis that $\mathcal{S}^h \backslash \mathcal{S}^-_{h+1}$ is comprised of uniformly random values. So if $\mathcal{S}^+_{h+1}$ is empty, then $\mathcal{S}^{h+1} = \mathcal{S}^h \backslash \mathcal{S}^-_{h+1}$ and thus is uniform. If $\mathcal{S}^+_{h+1}$ is non-empty, then again by the inductive hypothesis we have that $\mathcal{S}^{h+1}$ (the disjoint union of $\mathcal{S}^+_{h+1}$ and $\mathcal{S}^h \setminus \mathcal{S}^-_{h+1}$) is comprised of uniform independent values. This concludes the proof of Claim 6.7. □

From Claim 6.7, it follows that the messages sent to τ in round n that *correspond to* $H \in \mathcal{G}$ are uniformly random such that they sum to $(\sum_{u \in V(H)} m_u) \| 1^\kappa$ if $G = H$, and are completely uniformly random otherwise. Moreover, the messages corresponding to each graph are independent of one another.

Thus, we can bound correctness error simply by the probability that some other combination of messages (one from each neighbor of τ) not all corresponding to G sum to $x \| 1^\kappa$. Since τ has $\deg_G(\tau) \leq d_{\mathsf{max}}$ neighbors, by a union bound (and since $\kappa = \log(1/\delta) + d_{\mathsf{max}} \cdot \log |\mathcal{G}|$), this happens with probability at most

$$2^{-\kappa} \cdot |\mathcal{G}|^{\deg_G(\tau)} = \frac{\delta \cdot |\mathcal{G}|^{\deg_G(\tau)}}{|\mathcal{G}|^{d_{\mathsf{max}}}} \leq \delta.$$

Remark 6.8. Note that the probability of bad checksum only depends on $\mathcal{G}$ and $\deg_G(\tau)$. Thus, by simply *not* outputting (conditioned on receiving a unique valid checksum) with the appropriate probability we can *perfectly* simulate the functionality that outputs τ with probability (exactly) $1 - \delta$.

Similarly, it follows from Claim 6.7 that for non-terminal parties $\mathsf{P}_u \neq \mathsf{P}_\tau$, the messages received corresponding to any particular $H \in \mathcal{G}$ are independent, uniform and received from each neighbor $v \in \mathcal{N}^-_H(u) \cap \mathcal{N}(u)$ in round $\psi_{H,\tau}(u)$. Moreover, messages corresponding to distinct H's are independent. Because of that the view of the terminal party P_τ in round n is simply uniformly random conditioned on a random combination (one from each neighbor) summing to $(\sum_j m_j) \| 1^\kappa$. Thus, the simulator below is a perfect simulation of the view of any party P_u.

The Simulator The simulator Sim, controlling party P_u, begins by receiving $\mathcal{N}(u)$ from $\mathcal{W}^{\mathcal{G}}_{\text{graph-info}}(\mathcal{F}^\delta_{\mathsf{sum}}(\ell, \mathsf{P}_\tau))$ and the input m_u from the environment; denote $d = |\mathcal{N}(u)|$ and $\mathcal{N}(u) = \{v_1, \ldots, v_d\}$. Sim sends m_u to $\mathcal{W}^{\mathcal{G}}_{\text{graph-info}}(\mathcal{F}^\delta_{\mathsf{sum}}(\ell, \mathsf{P}_\tau))$ and if $u = \tau$, it receives the output y.

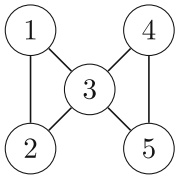


Fig. 25. The class of butterfly graphs consists of all possible permutations of the graph depicted above with nodes in {1, 2, 3, 4, 5}.

For $\mathsf{P}_u = \mathsf{P}_\tau$: Sample random $(i_1, \ldots, i_d) \in [n_1] \times \cdots \times [n_d]$, where $n_j = |\{H \in \mathcal{G} : v_j \in \mathcal{N}_H(\tau)\}|$. For each $v_j \in \mathcal{N}(\tau)$, sample a tuple of n_{v_j} independent uniformly random strings $s^j = (\tilde{s}^j_1, \ldots, \tilde{s}^j_{n_{v_j}})$ conditioned on the fact that $\sum_{j=1}^{d} s^j_{i_j} = y \| 1^\kappa$.

Before round n, the transcript of P_τ is empty. In round n, P_τ receives s^j from each neighbor v_j.

For $\mathsf{P}_u \neq \mathsf{P}_\tau$: In each round $i = 2, \ldots, n-1$, for each graph $H \in \mathcal{G}$ such that $\psi_{H,\tau}(u) = i$ and each neighbor $v \in \mathcal{N}^-_H(u) \cap \mathcal{N}(u)$, Sim simulates P_u receiving an independent uniformly random message, and sending messages as in an honest execution of the protocol on input m_u.

This concludes the proof of Lemma 6.4. □

6.2. *2-Connectivity is Not Necessary for 1-IT-THB (Butterfly Graph)*

In prior sections, we established a separation between 1-THAB and 1-THB, where the additional knowledge of the broadcaster can be leveraged. Namely, we presented graph classes for which 1-IT-THB can be trivially achieved via the flooding protocol, but 1-THAB requires computational cryptographic assumptions; e.g., 1-THAB on the class of three parties on a line implies key agreement (Proposition 5.1), and constant-round 1-THAB on $\mathcal{G}_{\text{2-vs-3}}$ implies infinitely often OT (Theorem 5.4). In this section, we provide a stronger separation by presenting classes of graphs on which 1-IT-THAB is impossible and flooding is not topology hiding, but there still exists 1-IT-THB.

One such example is the family of *butterfly graphs* (Fig. 25). As this class is not 2-connected, Proposition 5.1 rules out 1-IT-THAB. Further, given any graph in this family, a non-center node cannot tell which of its neighbors is central and which is not, and the center node cannot tell which of its neighbors are connected. For these reasons, the flooding protocol is not topology hiding, since, for example, a party of distance 2 from the broadcaster will learn which of its neighbors is central (the one who sends the message in the second round). Nevertheless, we show 1-IT-THB with perfect security can be achieved for this class.

Definition 6.9. (Butterfly graph) We denote by $G_{\text{12-3-45}}$ the butterfly graph consisting of 5 nodes $\{1, 2, 3, 4, 5\}$ and 6 edges $\{(1, 2), (1, 3), (2, 3), (3, 4), (3, 5), (4, 5)\}$ (see

Fig. 25), in which the center 3 forms one triangle with $\{1, 2\}$ and a second with $\{4, 5\}$. We denote by $\mathcal{G}_{\mathsf{butterfly}}$ the family of all permutations of the butterfly graph.

Theorem 6.10. *(IT-THB on butterfly) There exists a 1-IT-THB protocol with respect to $\mathcal{G}_{\mathsf{butterfly}}$ with perfect security.*

The main idea of the protocol is to run four *reliable message transmission (RMT)* protocols,[27] where the broadcaster P_{BC} acts as the sender and each of the other parties acts as the receiver P_R in one of the RMT executions. In an RMT execution, P_{BC} sends the message in the first round to the center node, who then forwards it to P_R. Two problems arise: First, the center node must deliver the message to P_R obliviously, so that P_R will not realize which of its neighbors is the center. We overcome this issue by splitting up the message into two additive secret shares and sending one share to P_R directly and the other share via its neighbor. The second problem is that the center node does not know which is the neighbor of P_R, and it therefore prepares a share for every possible neighbor. To hide the identity of the center node, all other parties behave accordingly, preparing secret shares of 0 for P_R. Finally, P_R can choose the correct shares, as it knows the identity of its neighbors, and reconstruct the message.

This protocol almost suffices for transmission of the message to P_R without leaking the topology, but there is one special case left to consider, namely if the center node itself is P_R. In this case, it could learn which neighbors are connected (by seeing which pairs of messages can be reconstructed to 0). We solve this by adding a blinding factor, which only comes into play if P_R is the center node. More precisely: If P_R has two neighbors (i.e., is not the center party), then in the first round P_R will send the same value to both its neighbors. (In this case, the additional blinding will be canceled out later.) If P_R has four neighbors (i.e., is the center node), then in the first round P_R will send distinct values to all its neighbors (in order to allow for this we have to work over a field with at least four elements, say over $\mathbb{F}_4 = \mathbb{Z}[X]/\langle X^2 + X + 1\rangle$), in which case the blinding terms will hide which of the parties are connected. The blinding term itself is agreed on in the second round by each pair of neighbors.

The formal description of the protocol is given in Fig. 26.

Lemma 6.11. *The protocol $\pi_{\mathsf{butterfly}}$ is a perfectly secure 1-THB protocol with respect to $\mathcal{G}_{\mathsf{butterfly}}$.*

Proof. We start by proving correctness. If P_R is the center node, then correctness is obvious. Otherwise, the output equals

$$\begin{aligned}\gamma_u^v \oplus \gamma_v^u &= m_{v,u}^0 \oplus m_{u,v}^1 \oplus \beta_u \cdot (b_{v,u} \oplus b_{u,v}) \oplus m_{u,v}^0 \oplus m_{v,u}^1 \oplus \beta_v \cdot (b_{u,v} \oplus b_{v,u}) \\ &= (m_{v,u}^0 \oplus m_{v,u}^1) \oplus (m_{u,v}^0 \oplus m_{u,v}^1) \oplus (\beta_u \oplus \beta_v) \cdot (b_{v,u} \oplus b_{u,v}) \\ &= m \oplus 0 = m,\end{aligned}$$

[27] Reliable message transmission refers to the concept of transmitting messages in an incomplete network such that the receiver is guaranteed to receive the message [14,15]. In our setting, the challenge is to realize reliable message transmission in a topology-hiding manner. Note that as described in the following, reliable message transmission in particular implies broadcast in the semi-honest setting.

Protocol $\pi_{\mathsf{butterfly}}(\mathsf{P_{BC}})$

Hybrid model: The protocol is defined in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{butterfly}}}$-hybrid model.

Input: The broadcaster $\mathsf{P_{BC}}$ holds an input $m \in \{0,1\}$.

The protocol:

- Each party P_v sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{butterfly}}}$ and receives its neighbour-set $\mathcal{N}(v)$.
- Repeat for each receiver P_R:
  1. *Forwarding the message to the center node.* The sender $\mathsf{P_{BC}}$ sends the message m to its neighbors; in particular the center party receives m.
  2. *Generating blinding terms.* Every party $\mathsf{P}_u \neq \mathsf{P}_R$ chooses a random blinding term $b_{u,v} \xleftarrow{R} \mathbb{F}_4$ for every $v \in \mathcal{N}(u)$ and sends $b_{u,v}$ to P_v.
  3. *Generating suitable offsets.* Party P_R generates values β_u for its neighbors as follows:
     - If P_R is the center party, it samples a fresh $\beta_u \xleftarrow{R} \mathbb{F}_4$ for each $u \in \mathcal{N}(R)$, conditioned on all values being distinct, and sends β_u to P_u.
     - Otherwise, P_R chooses $\beta \xleftarrow{R} \mathbb{F}^4$, sets $\beta_u := \beta$ for both $u \in \mathcal{N}(R)$, and sends β_u to P_u.
  4. *First round of forwarding m from the center to P_R.* Every party $\mathsf{P}_u \neq \mathsf{P}_R$ chooses a message $m^0_{u,v} \xleftarrow{R} \mathbb{F}_4$ for every $v \in \mathcal{N}(u)$ and sends $m_{u,v}$ to P_v. Further:
     - If P_u is the center party, it sets $m^1_{u,v} := m \oplus m^0_{u,v}$. //secret share m
     - Otherwise, if $R \in \mathcal{N}(u)$, P_u sets $m^1_{u,v} := m^0_{u,v}$. //secret share 0
  5. *Second round of forwarding m from the center to P_R.* Note that in this step only parties P_u with $u \in \mathcal{N}(R)$ send messages, and only P_R receives messages.
     For $u \in \mathcal{N}(R)$, party P_u sets $\gamma^v_u := m^0_{v,u} \oplus m^1_{u,v} \oplus \beta_u \cdot (b_{v,u} \oplus b_{u,v})$ for every $v \in \mathcal{N}(u) \setminus \{R\}$, and randomly chooses $\gamma^v_u \xleftarrow{R} \mathbb{F}^4$ for every $v \notin \mathcal{N}(u)$. Finally, P_u sends $\{(v, \gamma^v_u)\}_{v \neq R}$ to P_R.
- If P_R is the center, it outputs m; otherwise, it outputs $\gamma^v_u \oplus \gamma^u_v$, where $\mathcal{N}(R) = \{u, v\}$.

Fig. 26. Information-theoretic 1-THB over $\mathcal{G}_{\mathsf{butterfly}}$.

as $\beta_u = \beta_v$.

We proceed to prove security. Let P_{v^*} be the corrupted party and $m_{v^*} \in \{0, 1, \epsilon\}$ its input. The simulator forwards m_{v^*} to the wrapped functionality $\mathcal{W}^{\mathcal{G}_{\mathsf{butterfly}}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P_{BC}}))$ and receives m and the neighbor-set $\mathcal{N}_G(v^*)$.

The simulator simulates a real execution of $\pi_{\mathsf{butterfly}}(\mathsf{P_{BC}})$ as follows. If $\mathsf{P_{BC}} \in \mathcal{N}_G(v^*)$, the simulator sends (B, m) to the adversary in the first step. In the second step, for all $u \in \mathcal{N}_G(v^*)$ the simulator samples random blinding terms $b_{u,v^*} \xleftarrow{R} \mathbb{F}_4$ and sends (u, b_{u,v^*}) to the adversary. In each round with $R \in \mathcal{N}_G(v^*)$, to simulate the third step the simulator samples a random $\beta_{v^*} \xleftarrow{R} \mathbb{F}_4$ and sends (R, β_{v^*}) to the adversary. Else, the simulator receives β_u from the adversary for each $u \in \mathcal{N}_G(v^*)$. In the fourth step, for each $u \in \mathcal{N}_G(v^*)$, the simulator chooses a random message $m^0_{u,v^*} \xleftarrow{R} \mathbb{F}_4$, sends (u, m^0_{u,v^*}) to the adversary, and receives $m^0_{v^*,u}$.

Finally, in the rounds with $v^* = R$, the simulator proceeds as follows:

- If P_{v^*} is the center node, then for each $u \neq v^*$ and for each $v \notin \{u, v^*\}$, the simulator samples $\gamma^v_u \xleftarrow{R} \mathbb{F}_4$ at random and sends $(u, \{v, \gamma^v_u\}_{v \neq v^*})$ to P_{v^*}.

- Otherwise, let $\mathcal{N}_G(v^*) = \{u, v\}$; the simulator samples $\gamma_u^v, \gamma_v^u \xleftarrow{R} \mathbb{F}_4$ conditioned on $\gamma_u^v \oplus \gamma_v^u = m$. For all $w \notin \mathcal{N}_G(v^*) \cup \{v^*\}$, the simulator samples $\gamma_u^w, \gamma_v^w \xleftarrow{R} \mathbb{F}_4$ at random and sends $(u, \{w, \gamma_u^w\}_{w \neq v^*})$ (and accordingly for v) to the adversary.

Note that the simulation exactly mirrors the protocols behavior except for the last step. We therefore only have to analyze the round where v^* is the receiver.

- If P_{v^*} is the center node, note that from the view of P_{v^*} in a real protocol execution every message γ_u^v (for $u, v \neq v^*$) is distributed uniformly at random, because P_{v^*} does not know $m_{v,u}^0$. It is left to argue that also the joint distribution of γ_u^v and γ_v^u is uniform, because we have $m_{v,u}^0 = m_{u,v}^1$. This is true, because $\gamma_u^v \oplus \gamma_v^u = \beta_u \cdot (b_{v,u} \oplus b_{u,v}) \oplus \beta_v \cdot (b_{u,v} \oplus b_{v,u})$, and we are guaranteed $\beta_u \neq \beta_v$ (as we only consider semi-honest adversaries). Therefore, the simulated view is distributed identically to the view of P_{v^*} in a real protocol execution.
- Otherwise, the messages γ_u^v and γ_v^u for $\mathcal{N}_G(v^*) = \{u, v\}$ in a real execution are indistinguishable from random conditioned on $\gamma_u^v \oplus \gamma_v^u = m$, since P_{v^*} has no knowledge of $m_{u,v}^1$ and $m_{v,u}^1$. Further, for all $w \notin \mathcal{N}_G(v^*) \cup \{v^*\}$, the values γ_u^w and γ_v^w are random from the point of view of P_{v^*}, as P_{v^*} does not know $m_{w,u}^0$ and $m_{w,v}^0$ and each of these terms appear once only.

This concludes the proof of Lemma 6.11. □

7. Key-Agreement Upper Bounds

In this section, we present our KA-based upper bounds. In Sect. 7.1, we show that 1-THAB can be achieved, assuming KA, on every class of graphs that contain at least 3 nodes (i.e., where an honest majority is guaranteed), and in Sect. 7.2 that 1-THB can be achieved, assuming KA, on every class of graphs.

7.1. *KA is Sufficient for 1-THAB on All Graphs with at Least Three Nodes*

We start by showing that key agreement is sufficient to achieve 1-THAB on all graphs with *at least three nodes*, in other words, on all graphs where an *honest majority* is guaranteed. As shown in Sect. 5.2, this is the best we can hope to achieve for general classes of graphs, since if the class contains a 2-path as well as a 3-path infinitely often OT is required.

At its core, the protocol works by having each pair of neighbors emulate a *virtual party*, whose internal state is secret shared between them. These virtual parties—corresponding to edges in the network—broadcast by running a modified version of *flooding* on the *line graph* of the network. In each round, the virtual parties hold a partial OR of some of the inputs and at each round will update this by OR-ing it with those of the neighboring parties; then, in the final round, their (secret shared) outputs are reconstructed by the real parties, who then learn the broadcast message. Communication from one virtual party to the next can be achieved using a key-agreement protocol, but we must ensure the "degree" of each virtual party (i.e., the number of other virtual parties it is adjacent to in the line graph) remains hidden. Indeed, uv, the virtual party emulated by neighbors

u and v, has to communicate with $\deg(u) + \deg(v)$ other virtual parties and this leaks $\deg(u)$ to v and $\deg(v)$ to u. This issue is dealt with by having it update its state with the neighboring virtual parties in a circular fashion: This way, uv only needs to interact with the previous one on u's side and the next one on v's side. As discussed in Introduction (Sect. 1.2.4), we introduce the *dead-end channels* technique to deal with the case where u or v has degree one; hence, when the previous or next virtual party is entirely simulated by u or v (which should not be allowed to learn the partial-OR too early).

Virtual Parties The player whose label in the communication graph is u will be denoted (u). In addition to these *real* players, the protocols in this section will introduce some *virtual* parties, each emulated by a pair of neighbors in the graph. The virtual party emulated by (1) and (2), for instance, will be denoted (1-2). We extend this notion to include "fake" virtual parties, between two real parties which are not neighbors; it may even be that one of the two is not even in the graph. Supposing (1) and (2) are two non-neighboring parties in the graph, (1-2) denotes both the virtual party entirely emulated by (2) (who is pretending to talk to (1)) and the one entirely controlled by (1); however, this abuse will not lead to any confusion as the notation should be clear from context. There are therefore two types of virtual parties: those between two (distinct) neighboring parties in the graph, and which we call *good*, those which are not—e.g., (1-2) or (2-4) in the graph (2)-(3)-(4)—and which we call *bad*.

Reminder: OT correlations Beaver [7] showed that OT can be "precomputed" in the following sense. Consider a trusted dealer that independently and uniformly samples three bits $s, R_0, R_1 \leftarrow \{0,1\}$. The dealer gives (s, R_s) to the OT receiver and (R_0, R_1) to the OT sender. Next, on inputs $b \in \{0,1\}$ and $(m_0, m_1) \in \{0,1\}^2$, resp., the OT receiver and sender proceed as follows:

1. The receiver sends $\beta \leftarrow s \oplus b$ to the server.
2. The sender crafts the following two messages C_0 and C_1, then sends (C_0, C_1) to the receiver:
   - If $\beta = 0$: $C_\sigma \leftarrow m_\sigma \oplus R_\sigma$, for $\sigma \in \{0,1\}$.
   - If $\beta = 1$: $C_\sigma \leftarrow m_\sigma \oplus R_{1-\sigma}$, for $\sigma \in \{0,1\}$.
3. The receiver outputs $C_b \oplus R_s$ (which is equal to m_b).

In particular, note that two OT correlations are sufficient for the secure computation of an *OR* $b_0 \vee b_1$ between an input bit b_0 of the receiver and an input bit b_1 of the sender [6].

7.1.1. *High-Level Overview*

In the following, we describe the THAB protocol, on a graph $G = (V, E)$. We will outline the scheme for the special case of classes of *paths of unknown length* (subject to some public upper bound, and lower bounded by three) and then provide a generalization to classes of *arbitrary graphs with at least three nodes.*

Overview Part 1: Secure Flooding between Good Virtual Parties For the purposes of presenting the high-level idea of the protocol, it is sufficient to consider as an example the class containing exactly the paths (1)-(2)-(3)-(4) and (2)-(3)-(4), as it captures all the

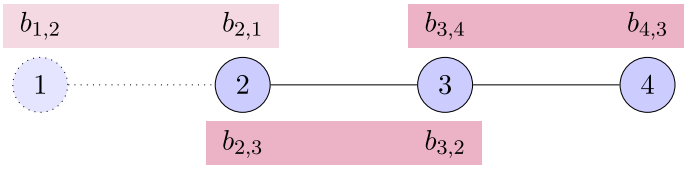


Fig. 27. Path ①-②-③-④ versus path ②-③-④.

technical difficulties. Here, hiding the topology and the identity of the broadcaster means in particular that parties ③ and ④ should not learn whether party ① is present, and no party should learn their distance to the broadcaster. Our protocol will introduce virtual parties (1-2), (2-3), and (3-4) which are represented in Fig. 27 by thulian pink bars. The first may be either good or bad depending on the graph, while the other two are always good.

The *state* of virtual party $(u\text{-}v)$ is additively secret shared as two bits $b_{u,v}$ and $b_{v,u}$, held, respectively, by (u) and (v) (unless the virtual party is bad, in which case both are held by the same node). This state is initialized to 0, i.e., $b_{u,v} = b_{v,u} \leftarrow \{0, 1\}$. Finally, the broadcaster will add the message m to its shares, i.e., (u), if broadcasting, would redefine $b_{u,v} \leftarrow b_{u,v} \oplus m$ for each neighbor $v \in \mathcal{N}_G(u)$. The protocol works by maintaining the following invariant:

> **Invariant:** At the end of round k, the state of every good virtual party $(u\text{-}v)$ is the OR of the input bits of all parties in the union of the k-neighborhoods of (u) and (v). Additionally, the state of every bad virtual party is random.

It follows that after diameter rounds, each pair of neighbors holds the OR of the input bits of all the parties, which is the broadcast bit.

We first describe a simplified scheme which two neighboring virtual parties, i.e., virtual parties sharing a real party, can use to compute the OR of their states. This protocol is always correct and is secure if both virtual parties are good. We will then show how to boost security to tolerate bad ones.

> **Step 1: Simplified Initialization:** Using a key-agreement protocol, each party establishes a secure channel with each neighbor's neighbor. (If ① is not present, ② will simulate the key exchange, i.e., establish a secure channel with ③ by playing the role of ①.) In the following, we simply write "③ *sends a message to* ①" to signify that ③ sends a message to ① via the secure channel established during initialization.
>
> **Step 2: Computing the new message:** The following updates the state of the virtual party (2-3) as the *OR* of its message and the message held by (1-2). For now, assume that if ① is not present it is fully simulated by ②. The parties proceed as follows:
>
> - ② sends $b_{2,1}$ to ③.
> - ② sends $b_{2,3}$ to ①.
> - ② sends two OT correlations to ① and ③.

Now, ① and ③ can use the OT correlations to compute $\beta_{1,3}$ and $\beta_{3,1}$ such that

$$\beta_{1,3} \oplus \beta_{3,1} = (b_{1,2} \oplus b_{2,1}) \vee (b_{2,3} \oplus b_{3,2}).$$

Step 3: Redistributing the new message: Parties ① and ③ pass on the new message to the virtual party ②-③ as follows:

- Parties ① and ③ agree on a random value $s_{1,3}$ (hidden from ② if ① exists).
- Party ③ chooses a share $b_{3,2}^{\mathsf{new}}$ and sends $\beta_{3,1} \oplus s_{1,3} \oplus b_{3,2}^{\mathsf{new}}$ to ②.
- Party ③ sends $\beta_{1,3} \oplus s_{1,3}$ to ②.
- Party ② can now receive its share of the new message of ②-③ via:

$$b_{2,3}^{\mathsf{new}} = (\beta_{3,1} \oplus s_{1,3} \oplus b_{3,2}^{\mathsf{new}}) \oplus (\beta_{1,3} \oplus s_{1,3}).$$

This protocol is correct, but not topology-hiding because ② is potentially in control of both party ① and ② and therefore may learn in which round the message reaches the virtual party ①-②. We thus have to alter the initialization phase, letting ② simulate the setup of a secure channel between ③ and a non-existing party without learning the key itself. This can be achieved, again, using a third party to set up OT correlations (specifically, the third party samples $m_0, m_1, \sigma \xleftarrow{R} \{0,1\}$, then deals out (m_0, m_1) to one party and (m_σ, σ) to the other).

Overview Part 2: "Dead-End Channels" We now explain how establishing *dead-end channels* (as discussed in Sect. 1.2.4) solves the previous security problem introduced by bad virtual parties.

Step 1: Initialization: The parties proceed as follows.

- *Setting up keys with each neighbor's neighbor.* This step is the exact same as the simplified initialization from the previous scheme: Each party establishes a secure channel with each neighbor's neighbor, where we assume that ② simulates ① if the latter is not present. We denote $k_{u,v}$ the key established by $\mathcal{U}$ with $\mathcal{V}$. Exemplifying, the result of ③ establishing a secure channel with ① is depicted below.
  - *If ① is present:*

  - *If ① is not present:*

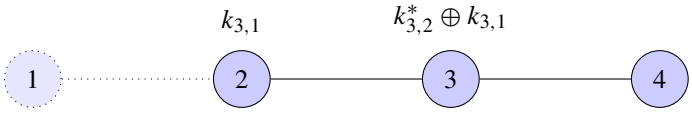
- *Establishing dead-end channels.* If a party is not present—i.e., when a good virtual party attempts to communicate with a bad one—we want to replace the

(in fact in-)"secure" channel to that party with a *dead-end* channel, over which the party that is simulating a bad virtual party cannot gain any information. This replacement must be done obliviously with respect to the other party. In our example class, the only dead-end channel which may potentially need to be established is between ③ and ②—i.e., when 2-3 needs to send a message to 2-1. This can be achieved as follows: ③ chooses a fresh key $k^*_{3,2}$ for its neighbor ②. The idea is that ② can choose, whether it wants to receive the key $k^*_{3,2}$ or not, depending whether ① exists or not. This is done by letting ④ set up OT-correlations for ③ and ②, where ③ acts as sender and ② as receiver. (To pass on the correlation to ②, party ④ uses the secure channel established in the first step.)

Now, ② uses the newly generated OT channel with ③ to receive $k^\star_{3,2}$ if and only if ① exists. Then, if ① exists, ② forwards $k^*_{3,2}$ to ①, who defines the new key as $k^*_{3,2} \oplus k_{3,1}$, as depicted below; this key is shared with ③.

– *If* ① *is present:*

– *If* ① *is not present:*



In particular, ③ has either established a *secure channel* to ① if ① is present, or a *dead-end channel*[28] otherwise, but is oblivious to which it is. From now, by sending a message, e.g., from ③ to ① we mean sending a message encrypted with $k^*_{3,2} \oplus k_{3,1}$. When using only the key $k_{3,1}$ for encryption (which will be crucial to establish correctness in the following), we will be explicit.

Note that simulating the OT correlations when establishing dead-end channels (even for more general graphs) does not leak anything about the topology because when u has degree one it already knows that its neighbor v must have degree at least two and therefore will choose[29] to receive the key $k^\star_{\cdot,\cdot}$ (as we are guaranteed $|V| \geq 3$).

One problem left to solve is to ensure that correctness is preserved if ① does not exist, as ② no longer knows the keys used to encrypt the messages to and from ①. This can be solved by redefining how the new message of the virtual party is computed, such that

[28] The channel is a dead-end because ③ used a KA protocol to set up a shared key with who they thought was ①, simulated by ②, but ② obliviously elected not to learn this key. Therefore, whatever information ③ sends over this channel is lost.

[29] Recall that we are crucially dealing with *semi-honest* adversaries in this paper.

(2) can ensure correctness even if (1) does not exist, while keeping the topology—and therefore in particular the message held by the (simulated) virtual party (1-2)—hidden from (2).

Note that we assume that (2) can still simulate the messages of (1) (without learning their content) by simply sending random strings. Assuming that all messages are encrypted via one-time pads (by either setting up a long enough key or by using the key as seed to a pseudorandom function), this is indistinguishable from the view of (3).

Step 2: Computing the new message: (1) and (3) compute $\beta_{1,3}$ and $\beta_{3,1}$ as before, but now use the key $k^*_{3,2} \oplus k_{3,1}$ to encrypt their communication (i.e., any information sent between (1) and (3) is *in any case* hidden from (2)).

Step 3: Redistributing the new message: To pass on the message from (1) and (3) to the virtual party (2-3), the parties proceed as follows:

- Parties (1) and (3) agree on a random value $s_{1,3}$ (hidden from (2) if (1) exists).
- (1) (potentially simulated by (2)) sends a random message $r_{1,3}$ to (3), *encrypted by* $k_{3,1}$. (Note that if (1) exists this message is hidden from (2), but if (1) does not exist, (2) has full control over the channel.)
- (2) sends a random message $r_{2,3}$ to (3).
- (1) sends $\beta_{1,3} \oplus s_{1,3}$ to (2).
- (3) sets $b^{\mathsf{new}}_{3,2} \leftarrow b_{3,2} \oplus r_{1,3} \oplus r_{2,3}$ and sends $\beta_{3,1} \oplus s_{1,3} \oplus b^{\mathsf{new}}_{3,2}$ to (2).
- If (1) exists, (2) defines its new share as: $b^{\mathsf{new}}_{2,3} \leftarrow (\beta_{3,1} \oplus s_{1,3} \oplus b^{\mathsf{new}}_{3,2}) \oplus (\beta_{1,3} \oplus s_{1,3})$.

If (1) does not exist, (2) defines its new share as: $b^{\mathsf{new}}_{2,3} \leftarrow b_{2,3} \oplus r_{1,3} \oplus r_{2,3}$.

We re-established correctness, as now the new value of the virtual party (2-3) is simply a re-randomization of its previous shares. Note that if (1) exists, $b^{\mathsf{new}}_{2,3}$ constitutes a perfectly random fresh share both from the view of (2) and (1) individually, therefore the above change does not hurt security.

Overview Part 3: From Paths to General Graphs The setting for general graphs is more difficult, because parties no longer know the identities of their neighbors. We solve this by organizing communication in a particular way that is locally determined and globally consistent. Again, we consider each pair of neighbors as a virtual party, i.e., in the example below the virtual parties are (z-w), (v-w), (u-v), and (v-x). Second, to decide on how the message is passed along, each node with degree > 1 establishes a cyclic ordering on their neighbors, e.g., (Fig. 28) (v) established the ordering (u) $\rightarrow$ (w) $\rightarrow$ (x) $\rightarrow$ (u). The parties that only have one neighbor will simulate another neighbor; for example (Fig. 28), (z) will set up (w) $\rightarrow$ (z) $\rightarrow$ (w). Note that it is crucial that the neighbors never learn the actual identity of their next node and predecessor according to such an ordering.

Now, the idea is that the message is passed along node-by-node according to this cyclic ordering. In each step, one node will act as the center node. These steps are predetermined by the label in $[n]$; therefore, each node (and its neighbors) knows exactly in which round it is their turn. Further, the operations are local, so all nodes that are not direct neighbors are not affected. In the following, we will describe the protocol at the example of (v).

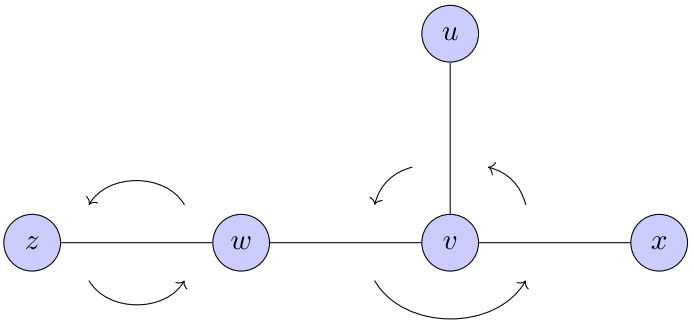


Fig. 28. Examples of cyclic orderings.

When it is (v)'s turn, each neighbor will set up a key with their next node (without learning their identity), i.e., (u) will set up a key with (w), (w) will set up a key with (x), and (x) will set up a key with (u). In the next step, the parties potentially set up dead-end channels—exactly as (2) did when (1) was not present in our example of a line. This will, e.g., prevent (z) from learning the messages sent by (w) to its *simulated* next node.

Once the secure channels are established, the message is passed through the graph as follows: Let $m_{v,w}$, $m_{v,x}$, and $m_{v,u}$ be the messages held by the parties (v-w), (v-u), and (v-x), respectively, *before it is* (v)*'s turn in the protocol*. Then, afterward, all three virtual parties will hold $m_{v,w} \vee m_{v,x} \vee m_{v,u}$. To achieve this, in each step (in parallel) (v-★) and (v-next$_v$★) will compute the OR of their messages and pass it on to (v-★), for each neighbor (★) of (v). For example, (v-u) and (v-w) will compute $m_{v,u} \vee m_{v,w}$ (by (v) acting as the center node and dealing out OT correlations) and pass it on to the virtual party (v-u). By repeating this sufficiently many times (in particular, at most n), in the end all virtual parties containing (v) will hold the value $m_{w,v} \vee m_{v,x} \vee m_{v,u}$. Parties that have degree one proceed exactly according to party (2) before, simulating dealing out OT correlations and computing the secure OR (obliviously) and finally set their shares as a fixed re-randomization of the old shares.

Note that in one round of the protocol, with this strategy the message travels at least distance one (as was the case for the line). Thus, after n rounds, all virtual parties will hold the broadcast message.

7.1.2. *The Protocol*

For an integer n, denote the set of all graphs with $V \subseteq [n]$ that consist of at least 3 nodes by

$$\mathcal{G}^{3 \le V \le n} = \{G \text{ graph} : V(G) \subseteq [n], |V(G)| \ge 3\}.$$

Protocol π_{AB}

Hybrid model: The protocol is defined in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}^{3\le V\le n}}$-hybrid model.

Input: The broadcaster holds an input $m \in \{0,1\}$.

The protocol:

- Each party $\textcircled{v}$ sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}^{3\le V\le n}}$ and receives its neighbor-set $\mathcal{N}(v)$.
- **Setting up secure channels.**
  - *Establishing an ordering on the neighbors.* For each $v \in [n]$:
    - If $\textcircled{v}$ has degree ≥ 2: $\textcircled{v}$ sets an arbitrary cyclic permutation on its neighborhood; we use $\mathtt{next}_v(\cdot)$ and $\mathtt{pred}_v(\cdot)$ to denote the successor or predecessor of neighbors of $\textcircled{v}$ respective to permutation. //Note that the neighbors will not learn who their actual successor and predecessor is.
    - If $\textcircled{v}$ has degree 1: $\textcircled{v}$ will simulate a second neighbor, i.e., for the unique $u \in \mathcal{N}(v)$ we set $\mathtt{next}_v(u) = \mathtt{pred}_v(u) = v$. In particular, throughout the protocol execution in this case $\textcircled{v}$ will delay the messages accordingly to perfectly simulate a real neighbor.
  - *Setting up pre-channels.* For each $v \in [n]$ and $u \in \mathcal{N}(v)$:
    - $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$ establish a (long) key $k_{u,\mathtt{next}_v(u)}$ using KA ($\textcircled{v}$ acting as relay between them to exchange their messages). In the following, "$\textcircled{u}$ *sends a message* m *to* $\boxed{\mathtt{next}_v(u)}$ *via* $\mathtt{channel}^1_{u\to \mathtt{next}_v(u)}$" is used as shorthand for "$\textcircled{u}$ encrypts m under (a portion of) $k_{u,\mathtt{next}_v(u)}$, then sends the ciphertext to $\textcircled{v}$ who passes it on to $\boxed{\mathtt{next}_v(u)}$, who then decrypts it using $k_{u,\mathtt{next}_v(u)}$". //In case $\textcircled{v}$ has degree one, $\textcircled{v}$ simulates $\boxed{\mathtt{next}_v(u)}$ and therefore has *full access* to $\mathtt{channel}^1_{u\to \mathtt{next}_v(u)}$(otherwise, the channel is secure).
  - *Setting up secure/dead-end channels.* For each $v \in [n]$ and $u \in \mathcal{N}(v)$:
    - $\boxed{\mathtt{pred}_u(v)}$ deals (many) OT correlations to $\textcircled{u}$ and — via $\mathtt{channel}^1_{\mathtt{pred}_u(v)\to v}$— to $\textcircled{v}$, such that $\textcircled{u}$ acts as sender and $\textcircled{v}$ as receiver.
    - $\textcircled{u}$ chooses a (long) fresh key $k^*_{u,v}$, sets $m_0 = k^*_{u,v}$ and $m_1 = 0$, and $\textcircled{v}$ sets $b = 0$ if and only if it has degree at least two. The parties use the OT correlations dealt by $\boxed{\mathtt{pred}_u(v)}$ to allow $\textcircled{v}$ to securely receive m_b. //Even if $\textcircled{u}$ has degree one and therefore sets up the OT correlations itself, it does not learn anything as in this case it already knows that $\textcircled{v}$ must have degree at least 2 (and therefore pick $b = 0$).
    - If $\textcircled{v}$ has degree at least two, it forwards $k^*_{u,v}$ to $\boxed{\mathtt{next}_v(u)}$. In the following, $\textcircled{u}$ *sends a message* m *to* $\boxed{\mathtt{next}_v(u)}$ *via* $\mathtt{channel}^2_{u\to \mathtt{next}_v(u)}$ is used as shorthand for $\textcircled{u}$ encrypts m under (a portion of) $k_{u,\mathtt{next}_v(u)} \oplus k^*_{u,v}$, then sends the ciphertext to $\textcircled{v}$ who passes it on to $\boxed{\mathtt{next}_v(u)}$ for decryption. //In any case, $\textcircled{v}$ has *no access* to messages sent via $\mathtt{channel}^2_{u\to \mathtt{next}_v(u)}$(the channel is either secure or has a dead-end).
- **Initializing the messages of the virtual parties to m or 0.**
  - For each $v \in [n]$ and $u \in \mathcal{N}(v)$, party $\textcircled{v}$ sends a random share $\rho_{v,u}$ to $\textcircled{u}$.
  - For each $v \in [n]$, party $\textcircled{v}$ sets $b^0_{v,u} = \rho_{v,u} \oplus \rho_{u,v} \oplus m_v$, where $m_v = m$ if $\textcircled{v}$ is the broadcaster and $m_v = 0$, otherwise. //A virtual party $\boxed{u\text{-}v}$ holds message $b^0_{u,v} \oplus b^0_{v,u} = m$ at the end of this step, if and only if it contains the broadcaster.

Fig. 29. 1-THAB on the class $\mathcal{G}^{3\le V\le n}$.

Theorem 7.1. *(KA is sufficient for 1-THAB on all graphs of size at least 3) Let $n \in \mathbb{N}$ and let $\mathcal{G} \subseteq \mathcal{G}^{3\le V\le n}$. Assuming the existence of a key-agreement protocol, there exists a 1-THAB protocol with respect to $\mathcal{G}$.*

Protocol π_{AB} (cont'd)

- **Passing on the message along the virtual parties.** In each round $k \in [n]$, for each party $v \in [n]$ (one by one), for each neighbor $u \in \mathcal{N}(v)$ (in parallel):

  Set $b^{k-1,0}_{(\cdot,\cdot)} := b^{(k-1)}_{(\cdot,\cdot)}$. For each $i \in [n]$ proceed as follows:

  – *Computing the new message.*
  * $\textcircled{v}$ sends $b^{k-1,i-1}_{v,\mathtt{next}_v(u)}$ to $\textcircled{u}$ if $\mathtt{next}_v(u) \neq v$. Otherwise, $\textcircled{v}$ sends a random message to $\textcircled{u}$.
  * $\textcircled{v}$ sends $b^{k-1,i-1}_{v,u}$ to $\boxed{\mathtt{next}_v(u)}$.
  * $\textcircled{v}$ distributes two OT correlations to $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$.

  Now, $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$ can use the OT correlations to securely compute values $\beta^{k-1,i}_{u,\mathtt{next}_v(u)}$, $\beta^{k-1,i}_{\mathtt{next}_v(u),u}$ (held by $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$, respectively) such that

  $$\beta^{k-1,i}_{u,\mathtt{next}_v(u)} \oplus \beta^{k-1,i}_{\mathtt{next}_v(u),u} = (b^{k-1,i-1}_{u,v} \oplus b^{k-1,i-1}_{v,u}) \vee (b^{k-1,i-1}_{\mathtt{next}_v(u),v} \oplus b^{k-1,i-1}_{v,\mathtt{next}_v(u)}),$$

  where all communication goes via $\mathtt{channel}^2_{u\to\mathtt{next}_v(u)}$. If $\mathtt{next}_v(u) = v$, then $\textcircled{v}$ simulates the above step by replacing all messages sent via $\mathtt{channel}^2_{u\to\mathtt{next}_v(u)}$ by random values. //Note that communication via $\mathtt{channel}^2_{u\to\mathtt{next}_v(u)}$ is encrypted using one-time-pad encryption and the OR is computed by each opening two OT correlations (and sending no other messages).

  – *Redistributing the new message.* To pass on the message from $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$ to the virtual party $\textcircled{u\text{-}v}$, the parties proceed as follows.
  * $\textcircled{u}$ sends a random message $s^{k-1,i}_{u,\mathtt{next}_v(u)}$ to $\boxed{\mathtt{next}_v(u)}$ via $\mathtt{channel}^2_{u\to\mathtt{next}_v(u)}$.
  * $\textcircled{u}$ sends a random message $r^{k-1,i}_{u,\mathtt{next}_v(u)}$ to $\boxed{\mathtt{next}_v(u)}$ via $\mathtt{channel}^1_{u\to\mathtt{next}_v(u)}$.
  * $\textcircled{u}$ sends a random message $r^{k-1,i}_{u,v}$ to $\textcircled{v}$.
  * $\textcircled{u}$ sets $b^{k-1,i}_{u,v} := b^{k-1,i-1}_{u,v} \oplus r^{k-1,i}_{u,\mathtt{next}_v(u)} \oplus r^{k-1,i}_{u,v}$ and sends to $\textcircled{v}$

    $$b^{k-1,i}_{u\to v} := \beta^{k-1,i}_{u,\mathtt{next}_v(u)} \oplus s^{k-1,i}_{u,\mathtt{next}_v(u)} \oplus b^{k-1,i}_{u,v}.$$

  * $\boxed{\mathtt{next}_v(u)}$ sends $b^{k-1,i}_{\mathtt{next}_v(u)\to v} := \beta^{k-1,i}_{\mathtt{next}_v(u),u} \oplus s^{k-1,i}_{u,\mathtt{next}_v(u)}$ to $\textcircled{v}$.
  * If $\textcircled{u}$ exists, $\textcircled{v}$ defines its new share as $b^{k-1,i}_{v,u} := b^{k-1,i}_{u\to v} \oplus b^{k-1,i}_{\mathtt{next}_v(u)\to v}$.
    If $\textcircled{u}$ does not exist, $\textcircled{v}$ defines its new share as $b^{k-1,i}_{v,u} := b^{k-1,i-1}_{v,u} \oplus r^{k-1,i}_{u,\mathtt{next}_v(u)} \oplus r^{k-1,i}_{u,v}$.

  Finally, set $b^k_{(\cdot,\cdot)} := b^{k-1,n}_{(\cdot,\cdot)}$.

- **Output.** For each $v \in [n]$ and $u \in \mathcal{N}(v)$, $\textcircled{u}$ and $\textcircled{v}$ exchange $b^{(n)}_{u,v}$ and $b^{(n)}_{v,u}$ and output $b^{(n)}_{u,v} \oplus b^{(n)}_{v,u}$.

Fig. 29. continued.

Proof. For the protocol, we refer to Fig. 29.

Correctness We first consider the case that all parties are honest and show that in this case every party obtains the correct output, conditioned on all the key agreements being successful, which occurs with all but negligible probability.

We start by considering the first phase of the protocol, where the parties set up secure channels.

> *Setting up secure pre-channels* The parties invoke the KA protocol poly(n) times, such that each pair of parties obtains long enough keys to encrypt all further communication. Since we conditioned on all the KA protocols being correct, for each $v \in \mathcal{N}_G(u)$ parties $\textcircled{u}$ and $\boxed{\mathtt{next}_v(u)}$ established a functioning channel $\mathtt{channel}^1_{u\to\mathtt{next}_v(u)}$ encrypted by $k_{u,\mathtt{next}_v(u)}$.

Setting up secure/dead-end channels As the OT correlations are perfect, $\textcircled{v}$ and thus $\boxed{\texttt{next}_v(u)}$ will learn $k^*_{u,v}$ (generated by $\textcircled{u}$) whenever it has degree at least two. Thus, for all $v \in [n]$ and $u \in \mathcal{N}_G(v)$, the parties $\textcircled{u}$ and $\boxed{\texttt{next}_v(u)}$ established a functioning channel $\texttt{channel}^2_{u\to\texttt{next}_v(u)}$ encrypted via $k_{u,\texttt{next}_v(u)} \oplus k^*_{u,v}$.

After initialization, for each edge $\{u, v\}$ in the graph, we have $b^0_{u,v} \oplus b^0_{v,u} = m$ if either $\textcircled{u}$ or $\textcircled{v}$ is the broadcaster, and $b^{(0)}_{u,v} \oplus b^0_{v,u} = 0$, otherwise. We will prove the following invariant:

Invariant: At the end of round k, the state of every virtual party $\boxed{u\text{-}v}$ is the OR of the input bits of all parties in the union of the k-neighborhoods of $\textcircled{u}$ and $\textcircled{v}$.

This is obviously true after initialization. Assume this is true at the end of round $k-1$, and let us show this is also true at the end of round k, i.e.,

For each $v \in [n]$ and $u, w \in \mathcal{N}_G(v)$, the following is true: If the virtual party $\boxed{w\text{-}v}$ holds message m at the end of round $(k-1)$, then $\boxed{u\text{-}v}$ holds message m at the end of the k^{th} round.

Recall that computing and passing on the message for each $v \in [n]$ proceeds in n steps $i = 1, \ldots, n$. We will show the following: *If (in the k^{th} round) the virtual party $\boxed{\texttt{next}_v(u)\text{-}v}$ or $\boxed{u\text{-}v}$ holds message m at the end of step $i-1$, then $\boxed{u\text{-}v}$ holds the message after the i^{th} step.* We will further use the following: If at the end of step $i-1$ all virtual parties hold either 0 or m, the same holds true at the end of i^{th} step. As the protocol runs for $i \in [n]$ and $\textcircled{v}$ chose a cyclic permutation on its (at most n) neighbors, this suffices to prove that the message eventually reaches $\boxed{u\text{-}v}$ as required.

In the following, we consider the cases that v has degree at least two and v has degree one separately.

Case I: Node v has degree at least two In the i^{th} step, the parties proceed as follows:

- *Computing the new message.* $\textcircled{u}$ and $\boxed{\texttt{next}_v(u)}$ compute shares $\beta^{k-1,i}_{u,\texttt{next}_v(u)}$, $\beta^{k-1,i}_{\texttt{next}_v(u),u}$ such that

$$\beta^{k-1,i}_{u,\texttt{next}_v(u)} \oplus \beta^{k-1,i}_{\texttt{next}_v(u),u} = (b^{k-1,i-1}_{u,v} \oplus b^{k-1,i-1}_{v,u}) \vee (b^{k-1,i-1}_{\texttt{next}_v(u),v} \oplus b^{k-1,i-1}_{v,\texttt{next}_v(u)}).$$

  We thus have the following:

  – If before the i^{th} step all virtual parties hold either 0 or m, then $\textcircled{u}$ and $\textcircled{v}$ now hold shares of either 0 or m.
  – If the virtual parties $\boxed{u\text{-}v}$ or $\boxed{\texttt{next}_v(u)\text{-}v}$ hold message m before the i^{th} step, then $\textcircled{u}$ and $\boxed{\texttt{next}_v(u)}$ now hold shares of m.

- *Redistributing the new message.* We have

$$\begin{aligned} b^{k-1,i}_{v,u} &= (\beta^{k-1,i}_{u,\texttt{next}_v(u)} \oplus s^{k-1,i}_{u,\texttt{next}_v(u)} \oplus b^{k-1,i}_{u,v}) \oplus (\beta^{k-1,i}_{\texttt{next}_v(u),u} \oplus s^{k-1,i}_{u,\texttt{next}_v(u)}) \\ &= \beta^{k-1,i}_{u,\texttt{next}_v(u)} \oplus \beta^{k-1,i}_{\texttt{next}_v(u),u} \oplus b^{k-1,i}_{u,v}. \end{aligned}$$

  With the previous considerations, this yields:

– If before the i^{th} step all virtual parties hold either 0 or m, then $\boxed{u}$ and $\boxed{v}$ now hold shares of either 0 or m.
– If the virtual parties $\boxed{u\text{-}v}$ or $\boxed{\texttt{next}_v(u)\text{-}v}$ hold message m before the i^{th} step, then $\boxed{u}$ and $\boxed{v}$ now hold shares of m.

Case II: Node v has degree one. In this case, we only have to show that the message of the virtual party $\boxed{u\text{-}v} = \boxed{\texttt{next}_v(v)\text{-}v}$ is not affected. This is true, as we have:

$$\begin{aligned} b_{v,u}^{k-1,i} \oplus b_{u,v}^{k-1,i} &= b_{v,u}^{k-1,i-1} \oplus r_{u,\texttt{next}_v(v)}^{k-1,i} \oplus r_{u,v}^{k-1,i} \oplus b_{u,v}^{k-1,i-1} \oplus r_{u,\texttt{next}_v(v)}^{k-1,i} \oplus r_{u,v}^{k-1,i} \\ &= b_{v,u}^{k-1,i-1} \oplus b_{u,v}^{k-1,i-1}. \end{aligned}$$

As the protocol runs for n rounds, we can be sure that all virtual parties hold the correct message m.

Security It remains to prove that we can simulate the above protocol for a corrupted party. We now describe the simulator. Let $\boxed{v^*}$ be the corrupted party, $m_{v^*} \in \{0, 1, \epsilon\}$ (where ϵ denotes the empty string) its input and $\mathcal{N}_G(v^*)$ its neighborhood. The simulator forwards m_{v^*} to the wrapped functionality $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}_B}(\mathcal{F}_{\mathsf{anon\text{-}bc}})$ and receives m. Then, the simulator proceeds by running the protocol on one of two graphs, depending on the degree of v^*:

- If v^* has degree at least two, then the simulator simulates all parties $u \in \mathcal{N}_G(v^*)$ following the protocol as if they have degree one (v^* then being their sole neighbor), i.e., the protocol is simulated using a star graph centered in v^*.
- If v^* has degree one, then the simulator simulates a single degree-two neighbor $\boxed{u}$ for it (i.e., the simulated graph is a path of length three, with $\boxed{u}$ in the center).

Additionally, in both cases, if $m_{v^*} = \varepsilon$, then an arbitrary one of the neighbors of v^* is chosen to broadcast m.

Let's now show that this is indistinguishable from a real protocol execution for $\boxed{v^*}$. For the following, we assume that the key-agreement primitive is *perfectly secure* and show that in this case simulation is perfect. Security is then reduced to the security of the key agreement, via a hybrid argument.

• *Setting up pre-channels.* This step is perfectly indistinguishable from the real protocol execution, since $\boxed{v}$ does not learn the identity of the node with which it performs key agreement, allowing $\boxed{u}$ to perfectly simulate having another neighbor than v^* is it does not.
• *Setting up secure/dead-end channels.* The first step of this phase is also perfectly simulated, as it only depends on the local view of the graph.

If channel $\texttt{channel}^1_{u \to \texttt{next}_{v^*}(u)}$ is secure for all $u \in \mathcal{N}_G(v^*)$, then the simulation of the second step is perfect, as in this case the only information that $\boxed{v^*}$ can learn is the receivers bit if it set up the OT correlations itself. Note though that in this case it already knows that $\boxed{u}$ has degree at least two and therefore picks $b = 0$.
The last step is perfectly indistinguishable from a real protocol execution, as the only case where party $\boxed{v}$ does not forward $k_{u,v}^*$ to $\boxed{\texttt{next}_v(u)}$ is if v has degree one and thus $\boxed{\texttt{next}_v(u)} = \boxed{v}$.

Initializing the messages of the virtual parties to m or 0. This step is perfectly indistinguishable from a real execution, because only the local shares held by parties $\textcircled{u}$ with $u \in \mathcal{N}_G(v^*)$ might differ.

• *Passing on the message along the virtual parties.* We will show this by replacing progressively, via a sequence of hybrids, the messages received by v^* with independent uniformly random messages, and show that the adversary cannot distinguish between the hybrids and the real world. In particular, this shows that the view of the adversary is independent of the graph and the identity of the broadcaster (assuming it is not v^*), and therefore that the adversary cannot distinguish between the real world and our simulator (which runs the protocol on a certain star or path graph with an arbitrarily chosen broadcaster).

For each $k \in [n]$, $v \in [n]$, and $i \in [n]$, we define the following hybrids:

- $\mathcal{H}_0$ simulates the real world exactly.
- $\mathcal{H}_{1,k,v,i}$ simulates the real world exactly up to the end of step (k, v, i) and then replaces all the subsequent messages received by v^* (until the output reconstruction phase) with independent random messages.

We will show that

1. $\mathcal{H}_0$ and $\mathcal{H}_{1,1,1,1}$
2. $\mathcal{H}_{1,k,v,i}$ and $\mathcal{H}_{1,k,v,i+1}$ (for $i < n$)
3. $\mathcal{H}_{1,k,v,n}$ and $\mathcal{H}_{1,k,v+1,1}$ (for $k, v < n$)
4. $\mathcal{H}_{1,k,n,n}$ and $\mathcal{H}_{1,k+1,1,1}$ (for $k < n$)

are indistinguishable given the view of $\textcircled{u}$. For simplicity, we will only show point (2), as the others work in exactly the same way (although, as will become apparent in what follows, it is crucial that we show the indistinguishability of the hybrids in the correct order). Fix therefore $(k, v, i) \in [n] \times [n] \times [n-1]$; the difference between $\mathcal{H}_{1,k,v,i}$ and $\mathcal{H}_{1,k,v,i+1}$ is at step $(k, v, i+1)$. If v is neither v^* nor one of its neighbors, then v^* receives no message and the hybrids are tautologically indistinguishable.

- *If $v = v^*$ and v^* has degree at least two:*
  - In the phase of "*computing the new message*," all the messages v^* receives are those its neighbors exchange over channels of the form $\mathsf{channel}^2_{u \to \mathtt{next}_{v^*}(u)}$ (where u is a neighbor of v^*), but v^* does not have the key associated to these channels so all messages received look independent and random.
  - In the phase of "*Redistributing the message*," again whenever two of its neighbors communicate over a channel of the form $\mathsf{channel}^1_{u \to \mathtt{next}_{v^*}(u)}$ or $\mathsf{channel}^2_{u \to \mathtt{next}_{v^*}(u)}$, the messages v^* receives look independent and pseudorandom. Since $s^{k-1,i+1}_{u,\mathtt{next}_v(u) \to v}$ is random and unknown to $\textcircled{v}$ (by the security of the channel from $\textcircled{u}$ to $\textcircled{\mathtt{next}_v(u)}$), $b^{k-1,i+1}_{v,\mathtt{next}_v(u)}$ looks random to $\textcircled{v}$. Finally, since $r^{k-1,i+1}_{u,v}$ is random, $b^{k-1,i+1}_{\mathtt{next}_v(u),v}$ also looks random to $\textcircled{v}$.
- *If $v = v^*$ and v^* has a single neighbor u:*
  - In the phase of "*computing the new message*," since $\mathsf{channel}^2_{u \to \mathtt{next}_{v^*}(u)}$ is a dead-end channel, all messages v^* receives look random.

∗ In the phase of "*Redistributing the message*," all the messages received by v^* over $\mathsf{channel}^2_{u\to \mathtt{next}_{v^*}(u)}$, which is a dead-end channel, look random. In particular, v does not learn $s^{k-1,i}_{u,\mathtt{next}_v(u)\to v}$ and so $b^{k-1,i+1}_{u\to v}$ looks random. Finally, by definition, $r^{k-1,i+1}_{u,v}$ is random.

– *If v is a neighbor of v^**:

∗ In the phase of "*computing the new message*" v^* receives $b^{k-1,i}_{v,\mathtt{next}_v(v^*)}$ (or a random message instead if v has degree one, which immediately solves the question) and $b^{k-1,i}_{v,\mathsf{next}_v^{-1}(v^*)}$ from v. But since hybrid $\mathcal{H}_{1,k,v,i}$ is indistinguishable from $\mathcal{H}_0$, these two messages, which v^* received in the previous phase, are indistinguishable from random. Additionally, v^* receives two sets of random OT selection bits, which look random, and two independent sets of random OT messages, which also look random. Finally, since the OT correlations are perfect, all the communication over $\mathsf{channel}^1_{u\to \mathtt{next}_{v^*}(u)}$ or $\mathsf{channel}^2_{u\to \mathtt{next}_{v^*}(u)}$ looks random to v^* (whether v has degree one or not).
∗ In the phase of "*Redistributing the message*," the messages v^* receives from $\mathtt{next}_v(v^*)$ and $\mathsf{next}_v^{-1}(v^*)$ are random by definition.

In any case, all messages received by v^* at step $(k, v, i+1)$ look random to v^* and so $\mathcal{H}_{1,k,v,i}$ and $\mathcal{H}_{1,k,v,i+1}$ are indistinguishable to v^*. Then, composing all these hybrids yields the desired result.

- *Output Phase.* This phase is perfectly indistinguishable from a real execution as in both cases v^* gets the correct shares it was missing of the broadcast bit.

This concludes the proof of Theorem 7.1. □

*Achieving 1-*THC Note that for graph classes that are over a fixed set of nodes, if all parties can broadcast in a manner that hides topology, every pair of parties can run a key-agreement protocol to establish a secure channel. Thus, assuming an honest majority, we can use the BGW protocol [8] to securely compute any function in a topology-hiding manner. For an integer n, denote the class of graphs with *exactly* n nodes as

$$\mathcal{G}^n = \{G \text{ graph} : V(G) = [n]\}.$$

Corollary 7.2. *Let $n \geq 3$, let $\mathcal{G} \subseteq \mathcal{G}^n$, and let f be an n-party function. Then, assuming the existence of a key-agreement protocol, there exists a 1-THC protocol for f with respect to $\mathcal{G}$.*

7.2. *KA is Sufficient for 1-THB on All Graphs*

In this section, we show that a minor tweak of the 1-THAB protocol in Sect. 7.1 gives a 1-THB protocol for the class of *all graphs* of at most n nodes.

For an integer n, denote the class of graphs with *at most* n nodes as

$$\mathcal{G}^{\leq n} = \{G \text{ graph} : V(G) \subseteq [n]\}.$$

Theorem 7.3. *(KA is sufficient for* 1-THB *on all graphs) Let* $n \in \mathbb{N}$*, and let* $\mathcal{G} \subseteq \mathcal{G}^{\leq n}$*. Assuming the existence of a key-agreement protocol, there exists a* 1-THB *protocol with respect to* $\mathcal{G}$*.*

Proof. The protocol is exactly as in Fig. 29, except during the setup of the potentially dead-end channels. There, we replace the second step as follows:

- $\boxed{u}$ chooses a (long) fresh key $k^*_{u,v}$, sets $m_0 = k^*_{u,v}$ and $m_1 = 0$, and $\boxed{v}$ sets $b = 0$ if and only if it has degree at least two *or* $\boxed{u}$ *or* $\boxed{v}$ *is the broadcaster*. The parties use the OT correlations dealt by $\boxed{\mathtt{pred}_v(v)}$ to allow $\boxed{v}$ to securely receive m_b.

This change does not affect correctness. It is left to prove security.

Again, let $\boxed{v^*}$ be the corrupted party, let $m_{v^*} \in \{0, 1, \epsilon\}$ be its input, let $\mathcal{N}_G(v^*)$ be its neighborhood, and let $\boxed{u^*}$ be the broadcaster. The simulator forwards m_{v^*} to the wrapped functionality $\mathcal{W}^{\mathcal{G}^{\leq n}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(u^*))$ and receives m.

The simulator proceeds with simulations as follows: If $u^* = v^*$ or if $u^* \in \mathcal{N}_G(v)$:

- If v^* has degree at least two, then the simulator simulates all parties $\boxed{u}$ with $u \in \mathcal{N}_G(v^*)$ following the protocol as if all nodes u have degree one.
- If v^* has degree one, then the simulator simulates the single party $\boxed{u}$ with $u \in \mathcal{N}_G(v^*)$ according to the protocol as if u has degree two.

Otherwise:

- The simulator chooses an arbitrary node $w \in \mathcal{N}_G(v)$ and simulates party $\boxed{w}$ as if it has another neighbor $\boxed{u^*}$ (of degree one) and all other parties $\boxed{u}$ for $u \in \mathcal{N}_G(v)$ as if they have degree one.

The proof of the latter case is analogous to the proof of anonymous broadcast, as the special case does not affect $\boxed{v^*}$. We therefore focus on proving security in case $u^* = v^*$ or $u^* \in \mathcal{N}_G(v^*)$. For setting up channels, we only have to reconsider the step of setting up dead-end channels:

> *Setting up secure/dead-end channels.* If channel $\mathtt{channel}^1_{u \to \mathtt{next}_{v^*}(u)}$ is secure for all $u \in \mathcal{N}_G(v^*)$, then also the simulation of the second step is perfect, as in this case the only information that $\boxed{v^*}$ can learn is the receivers bit if it set up the OT correlations itself. Note though that as either $\boxed{u}$ or $\boxed{v^*}$ is the broadcaster, $\boxed{v^*}$ already knows that $\boxed{u}$ will always choose $b = 0$, regardless of its degree.

The main difference in the rest of the proof is that if $\boxed{v^*}$ is of degree one, $\boxed{v^*}$ has access to all messages sent via $\mathtt{channel}^2_{u \to \mathtt{next}_{v^*}(u)}$. Intuitively, this does not harm security as $\boxed{v^*}$ already knows that it *always* shares the message with its (unique) neighbor. As we only consider the case where the adversary knows its distance from the broadcaster (either 0 or 1), we only have to show that passing the message does not leak anything about the topology of the graph. The only step that might leak something about the topology of the graph is computing the new message, as this step is simulated if and only if v has degree one. Adapting the proof of Theorem 7.1 boils down the following claim: □

Claim 7.4. *Let $k \in [n]$, let $v \in [n]$, let $u \in \mathcal{N}_G(u)$, and let $i \in [n]$. At the end of the i^{th} step, if $\texttt{next}_v(u) \neq v$, then $b^{k-1,i}_{\texttt{next}_v(u),v}$ and $b^{k-1,i}_{v,\texttt{next}_v(u)}$ are individually indistinguishable from random from the view of (u), and $b^{k-1,i}_{u,v}$ and $b^{k-1,i}_{v,u}$ are individually indistinguishable from random from the view of $(\texttt{next}_v(u))$.*

Proof. • *Computing the new message.* In this step party, (v) learns $b^{k-1,i-1}_{v,\texttt{next}_v(u)}$ and party $(\texttt{next}_v(u))$ learns $b^{k-1,i-1}_{v,u}$. If (v) has degree one and is in the neighborhood of the broadcaster, then (v) can learn $b^{k-1,i-1}_{u,v}$ in this step, as it has access to $\texttt{channel}^2_{u\to\texttt{next}_v(u)}$. Note though that this does not leak anything about the topology of the graph to (v), as (v) already knows $b^{k-1,i-1}_{u,v} = b^{k-1,i-1}_{v,u} \oplus m$.
• *Redistributing the new message.* Since $\texttt{next}_v(u) \neq v$, $(\texttt{next}_v(u))$ cannot gain any information about $b^{k-1,i}_{u,v}$, $b^{k-1,i}_{v,u}$ (even conditioned on knowing $b^{k-1,i-1}_{v,u}$), because $r^{k-1,i}_{u,v}$ is random and hidden from $(\texttt{next}_v(u))$.

Altogether, this shows that (u), $(\texttt{next}_v(u))$ cannot distinguish whether they received the real value or a random message in the first step. □

This shows that regardless of the role that (v^*) has in a specific round of the protocol, its simulated view is perfectly indistinguishable from a real protocol execution regarding the degree of its neighbors. By the correctness of the protocol, again, outputs are distributed according to a real protocol execution. As the party (v^*) already knows its distance from the broadcaster, this concludes the proof of Theorem 7.3. □

8. Corollaries and Implications of our Techniques

In this section, we derive corollaries from the techniques and the results given in previous sections. In Sect. 8.1, we present a characterization of 1-THAB and 1-THC. In Sect. 8.2, we consider the more complex case of 1-THB and provide a characterization for graphs with at most four nodes.

8.1. *Characterization of 1-IT-THAB and 1-IT-THC*

We start by characterizing 1-THAB and 1-THC. In a similar spirit to Sects. 6.1 and 7, we denote the class of graphs with *up to n* nodes as

$$\mathcal{G}^{\leq n} = \{G \text{ graph } : V(G) \subseteq [n]\},$$

and the class of graphs with *exactly n* nodes as

$$\mathcal{G}^{n} = \{G \text{ graph } : V(G) = [n]\}.$$

1-THAB Recall that by Theorem 6.5, 1-IT-THAB can be achieved (albeit inefficiently) with respect to a class $\mathcal{G}$ that consists of only 2-connected graphs. Further, by Proposi-

tion 5.1, 1-IT-THAB cannot be achieved with respect to a class $\mathcal{G}$ that contains at least one graph with ≥ 3 nodes which is not 2-connected.

The degenerate case of the 2-*path* (two connected nodes) trivially enables 1-IT-THAB even though it is not 2-connected. We note that augmenting a class of 2-connected graphs with additional 2-path graphs still enables 1-IT-THAB, since each party can locally check if it has degree one (in which case it is on a 2-path with its sole neighbor) or if its degree is at least two, in which case the graph is guaranteed to be 2-connected. Therefore, we get the following corollary.

Corollary 8.1. *(Characterization of 1-IT-THAB) Let $n \in \mathbb{N}$, and let $\mathcal{G} \subseteq \mathcal{G}^{\leq n}$. Then, 1-IT-THAB is possible with respect to $\mathcal{G}$ if and only if every $G \in \mathcal{G}$ is either 2-connected or is a 2-path.*

Recall that by Theorem 5.4, if a class includes both a 2-path and a 3-path, then (constant-round) 1-THAB implies infinitely often OT. By considering graphs that contain at least three nodes, we achieve a cleaner characterization. Denote by K_2^n the class of all 2-path graphs over n nodes and recall that $\mathcal{G}_{\text{2-conn}}^{\leq n}$ stands for the class of 2-connected graphs with at most n nodes whereas $\mathcal{G}_{\text{2-conn}}^{n}$ for the class of 2-connected graphs with exactly n nodes.

Corollary 8.2. *(Characterization of 1-THAB with ≥ 3 parties) Let $n \in \mathbb{N}$ and let $\mathcal{G} \subseteq \mathcal{G}^{\leq n} \setminus K_2^n$.*

- *1-IT-THAB is possible with respect to $\mathcal{G}$ if and only if $\mathcal{G} \subseteq \mathcal{G}_{\text{2-conn}}^{\leq n}$ (i.e., all graphs in $\mathcal{G}$ are 2-connected).*
- *If $\mathcal{G} \setminus \mathcal{G}_{\text{2-conn}}^{\leq n} \neq \emptyset$ (i.e., there exist graphs that are not 2-connected), then 1-THAB is possible with respect to $\mathcal{G}$ if and only if key agreement exists.*

We note that the communication complexity of the 1-IT-THAB protocol in Corollaries 8.1 and 8.2 is polynomial in $|\mathcal{G}|$ and the computation complexity is polynomial in $|\mathcal{G}|$ and exponential in the maximum degree in $\mathcal{G}$.

1-THC To achieve 1-THC, we consider a fixed and publicly known set of parties and an honest majority, i.e., $\mathcal{G}^n$ for some $n \geq 3$. Recall that by Theorem 6.6, 1-THC can be computed with respect to a class $\mathcal{G} \subseteq \mathcal{G}^n$ that consists of only 2-connected graphs with information-theoretic security. This matches the lower bound given in Proposition 5.1, showing that otherwise key agreement is needed even for computing 1-THAB. Finally, by Corollary 7.2, 1-THC can be achieved over any class of graphs (with at least three nodes) assuming key agreement. Therefore, we conclude with the following corollary.

Corollary 8.3. *Let $n \geq 3$, let $\mathcal{G} \subseteq \mathcal{G}^n$, and let f be an n-party function.*

- *If $\mathcal{G} \subseteq \mathcal{G}_{\text{2-conn}}^{n}$ (i.e., all graphs in $\mathcal{G}$ are 2-connected), then $\mathcal{F}_{\text{sfe}}^{f}$ can be securely realized with statistical information-theoretic security in a topology-hiding manner with respect to $\mathcal{G}$, tolerating a single semi-honest corruption.*
  The communication complexity of the protocol is polynomial in $|\mathcal{G}|$, and the computation complexity is polynomial in $|\mathcal{G}|$ and exponential in the maximal degree in $\mathcal{G}$.

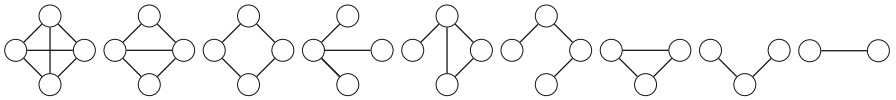


Fig. 30. The nine connected graphs of size at most four.

- *If* $\mathcal{G} \setminus \mathcal{G}^n_{\text{2-conn}} \neq \emptyset$ *(i.e., there exist graphs that are not 2-connected), then* $\mathcal{F}^f_{\text{sfe}}$ *can be securely realized in a topology-hiding manner with respect to* $\mathcal{G}$*, tolerating a single semi-honest corruption, if and only if key agreement exists.*

8.2. *Characterization of 1-IT-THB on Small Graphs*

We proceed to consider 1-THB. As indicated from previous sections, this case is more complex since the lower bounds for 1-THAB (presented in Sect. 5) that were strong enough to provide a clear-cut characterization in Sect. 8.1 do not carry over. In this section, we present a characterization for small graphs (with at most four nodes).

Isomorphically closed classes of small graphs There are nine connected graphs on four parties or fewer, up to isomorphism: the complete graph of size four (⊠), the diamond (◪), the four-cycle (□), the claw (◺), the paw (◭), the four-path (◸), the three-cycle (◸), the three-path (∧), and the two-path (／).
Rather than considering all graph classes, we only study those which are *isomorphically closed*: If a network (defined here as a labeled graph) is in the class, then so must all isomorphic networks be (where the label set is implicitly $\{1, 2, 3, 4\}$, corresponding to players ①, ②, ③, ④). In particular, there are $2^9 - 1 = 511$ non-empty classes.

Definition 8.4. (Isomorphic Closure) Let $n \in \mathbb{N}$. We say a graph class $\mathcal{G}$ comprised of graphs with at most n vertices and labeled by a subset of $\{1 \dots n\}$ is *isomorphically closed*—if, for every graph $G = (V, E) \in \mathcal{G}$ and for every injective function $p : V \to \{1, \dots, n\}$, the graph $H = (p(V), E_H)$ is also in $\mathcal{G}$, where E_H is defined by $(u, v) \in E \Leftrightarrow (p(u), p(v)) \in E_H$.

In other words, if a graph is in an isomorphically closed class, then so must also be all the graphs obtained by relabeling it using a subset of $\{1, \dots, n\}$.

Without this restriction, which allows for a purely graph-theoretic property-based dichotomy, the theorems in this section do not characterize when 1-IT-THB and 1-IT-THAB are possible on every single graph class. However, since our impossibility results rely on only specific combinations of labeled graphs being in the class (which are guaranteed to be there under an isomorphic closure hypothesis), they may be carefully applied to more, albeit often less natural, graph classes on a case by case basis.

Note that the class $\mathcal{G}_{\text{oriented-5-path}}$ from Sect. 4.1 is an example of a class which does not have this closure property: The node labeled ① is always the root of the directed path.

Theorem 8.5. *(Characterization of 1-IT-THB on small graphs) Let* $\mathcal{G}$ *be an isomorphically closed graph class containing only (connected) graphs of size at most four.*

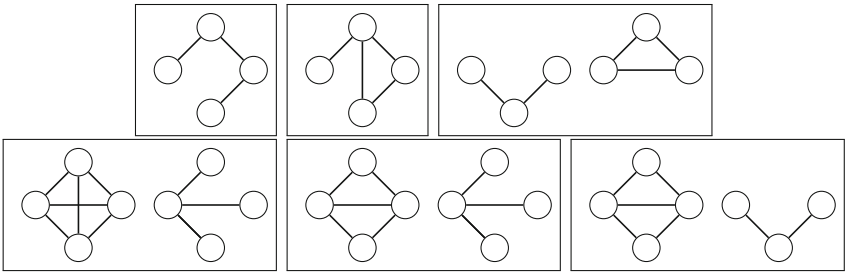


Fig. 31. The six *forbidden classes* for 1-IT-THB .

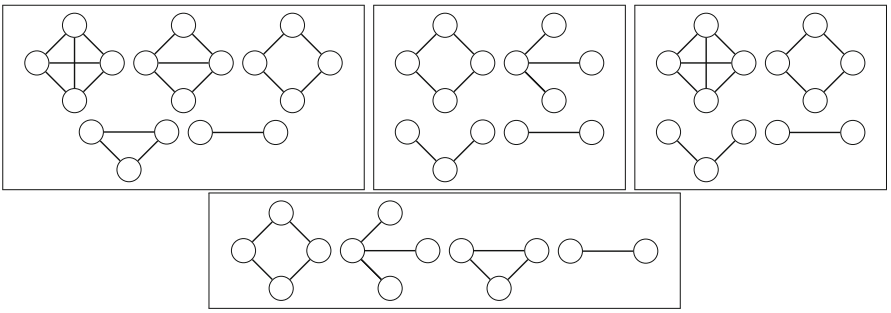


Fig. 32. The four maximal *forbidden-pattern-free classes* for 1-IT-THB .

1. *If the class is a superset of any of the six following sets, then 1-IT-THB on $\mathcal{G}$ is infeasible:* $\{\boxed{2}\}$, $\{\boxed{\angle}\}$, $\{\boxed{\triangledown}, \wedge\}$, $\{\boxtimes, \boxed{\angle}\}$, $\{\boxed{\diagup}, \boxed{\angle}\}$, *and* $\{\boxed{\diagup}, \wedge\}$. *Specifically, 1-THAB with respect to such $\mathcal{G}$ implies key agreement.*
2. *Conversely, if the class is not a superset of any of the six aforementioned sets, then 1-IT-THB is possible. Note that these are the subsets of the following four maximal sets:* $\{\boxtimes, \boxed{\diagup}, \square, \triangledown, /\}$, $\{\square, \angle, \wedge, /\}$, $\{\boxtimes, \square, \wedge, /\}$, *and* $\{\square, \angle, \triangledown, /\}$.

The proof of Theorem 8.5 follows from Lemmas 8.6 and 8.7. In Lemma 8.6, we identify the six minimal classes for which 1-THB implies KA. The main proof technique employed is the *phantom bridge argument* (that was discussed in Remark 4.6). It follows that 1-THB on a superclass of any of these six classes implies KA.
In Lemma 8.7, we show that on all other classes 1-IT-THB is possible. To that end, we identify the maximal classes which are not superclasses of any of the six aforementioned forbidden patterns: It suffices to show that 1-IT-THB is feasible on each of these four classes.

8.2.1. *Key-Agreement Implications*

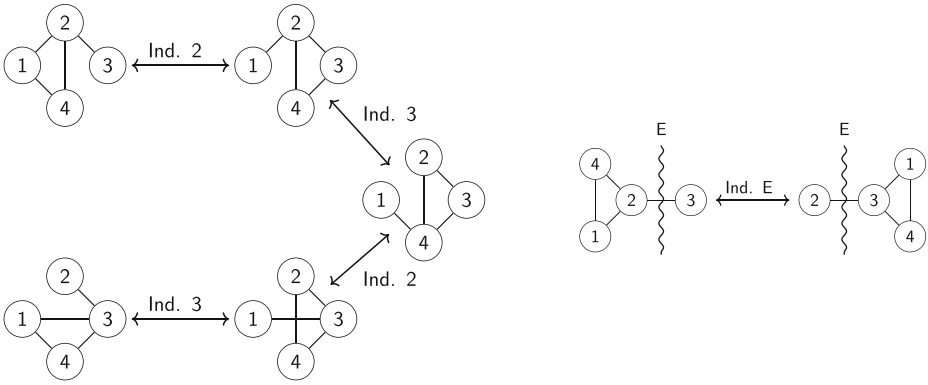
Lemma 8.6. *1-THB on each of the following classes implies key agreement:*

1. $\{\boxed{2}\}$ *("The Path")*
2. $\{\boxed{\triangledown}, \wedge\}$ *("The Triangle")*
3. $\{\boxed{\angle}\}$ *("The Paw")*

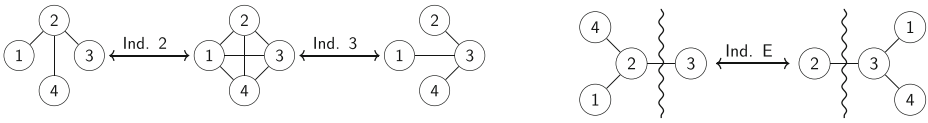
4. $\{\boxtimes, \swarrow\}$ *("The Full Claw")*
5. $\{\boxslash, \swarrow\}$ *("The Diamond Claw")*
6. $\{\boxslash, \wedge\}$ *("The Diamond Path")*

Proof (sketch). We prove the lemma for each class separately.

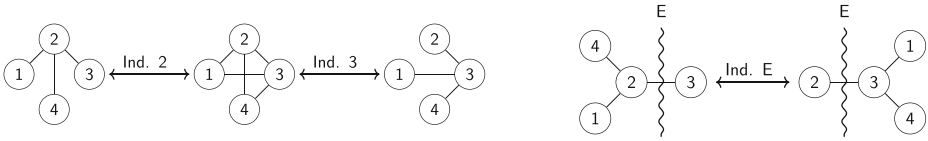
1. By Ball et al. [[4], Thm. 3.1], 1-THB on $\{\boxminus\}$ implies KA.
2. By Theorem 4.4, 1-THB on $\{\boxbslash, \wedge\}$ implies KA.
3. We prove that 1-THB on $\{\boxslash\}$ implies KA via the phantom bridge argument, and using a similar construction to Fig. 17 in Sect. 4.2. The following proof sketch lists the points which need to be adapt the construction to $\{\boxslash\}$:
   - If Alice's coin c_A is 1, she simulates parties ①, ②, and ④ (fully pairwise connected); if her coin is 0, she only simulates party ②.
   - If Bob's coin c_B is 1, he simulates parties ①, ③, and ④ (fully pairwise connected); if his coin is 0, he only simulates party ③.
   - In the event Alice and Bob toss different coins, the topology is either one of the two paws where ② and ③ are connected, and where either {①, ②, ④} or {①, ③, ④} are fully connected. The reason why an eavesdropper between Alice and Bob (having access only to communication between nodes 2 and 3) cannot determine whether $c_A = 1 - c_B = 1$ or $c_A = 1 - c_B = 0$ with more than negligible probability follows from a standard hybrid argument (similar to the one used in the proof of Theorem 4.4) and is sketched in the following diagram:



4. We prove that 1-THB on $\{\boxtimes, \swarrow\}$ implies KA via the phantom bridge argument. We list the differences needed to adapt Fig. 17 to this case:
   - If Alice's coin c_A is 1, she simulates parties ①, ②, and ④; if her coin is 0, she only simulates party ②.
   - If Bob's coin c_B is 1, he simulates parties ①, ③, and ④; if his coin is 0, he only simulates party ③.
   - In the event Alice and Bob toss different coins, the topology is either (1, 4)-2-3 or 2-3-(1, 4). The proof follows from a similar hybrid argument, as depicted in the diagram below.

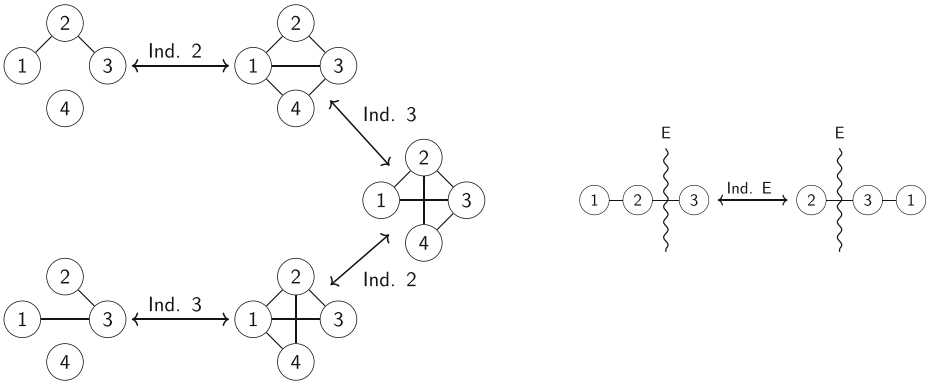


5. We prove that 1-THB on $\{\boxtimes, \angle\}$ implies KA via the phantom bridge argument. We list the differences needed to adapt Fig. 17 to this case:
   - If Alice's coin c_A is 1, she simulates parties ①, ②, and ④; if her coin is 0, she only simulates party ②.
   - If Bob's coin c_B is 1, he simulates parties ①, ③, and ④; if his coin is 0, he only simulates party ③.
   - In the event Alice and Bob toss different coins, the topology is either the claw (1, 4)-2-3 or the claw 2-3-(1, 4). The proof follows from a similar hybrid argument, as depicted in the diagram below.



6. We prove that 1-THB on $\{\boxtimes, \angle\}$ implies KA via the phantom bridge argument. We list the differences needed to adapt Fig. 17 to this case:

- If Alice's coin c_A is 1, she simulates both parties 1 and 2; if her coin is 0, she only simulates party 2.
- If Bob's coin c_B is 1, he simulates both parties 1 and 3; if his coin is 0, he only simulates party 3.
- In the event Alice and Bob toss different coins, the topology is either 1-2-3 or 2-3-1. The proof follows from a similar hybrid argument, as depicted in the diagram below.



This concludes the proof of Lemma 8.6. □

8.2.2. *Information-Theoretic Feasibility Results*

The proof of Lemma 8.7 is of little technical interest as the techniques are unlikely to extend to graph classes far beyond those considered here. In fact the lemma itself is of

little intrinsic value and is only considered because it presents an upper bound to match the lower bound of Lemma 8.6.

Lemma 8.7. *Let $\mathcal{G}$ be an isomorphically closed graph class containing only (connected) graphs of size at most four. If $\mathcal{G}$ satisfies one of the two equivalent conditions, then 1-IT-THB is possible:*

1. *The class is not a superset of any of the six following sets, $\{\boxed{\angle}\}$, $\{\boxed{\angle}\}$, $\{\boxed{\triangledown}, \wedge\}$, $\{\boxtimes, \boxed{\angle}\}$, $\{\boxed{\triangle}, \boxed{\angle}\}$, and $\{\boxed{\triangle}, \wedge\}$;*
2. *The class is a subset of the following four maximal sets: $\{\boxtimes, \boxed{\triangle}, \square, \boxed{\triangledown}, /\}$, $\{\square, \boxed{\angle}, \wedge, /\}$, $\{\square, \boxed{\angle}, \boxed{\triangledown}, /\}$, and $\{\boxtimes, \square, \wedge, /\}$.*

Proof. We now show that there is an 1-IT-THB protocol for each of the four maximal sets: $\{\boxtimes, \boxed{\triangle}, \square, \boxed{\triangledown}, /\}$, $\{\square, \boxed{\angle}, \wedge, /\}$, $\{\square, \boxed{\angle}, \boxed{\triangledown}, /\}$, and $\{\boxtimes, \square, \wedge, /\}$.

1. 1-IT-THB *is possible on* $\{\boxtimes, \boxed{\triangle}, \square, \boxed{\triangledown}, /\}$: First of all, note that all parties can locally identify if the topology is isomorphic to the 2-path $/$ (if they have degree one). Therefore, it suffices to provide a protocol for the class $\{\boxtimes, \boxed{\triangle}, \square, \boxed{\triangledown}\}$. Consider the following protocol; without loss of generality we consider the broadcaster to be ①.

- In the first round, party ① sends the output bit to each of its neighbors.

Notice that the following holds: Since ① has at least two neighbor*s, there is at most one of the three nodes ②, ③, ④ which does not hold the bit at this point. If ① has degree only two, then either that means there is no party which is not a* neighbor *of ① (topology $\boxed{\triangledown}$) or that there is (topology $\boxed{\triangle}$ or $\square$), but in the latter case it must hold that ① and that party have exactly the same neighborhood.*

- In the second round (could be done unambiguously in the first round instead), ① sends to each of its neighbors two messages: each one tagged with the ID (②, ③, or ④) of one of the two remaining potential parties (e.g., ① sends messages tagged "②" and "④" to party ③). The messages tagged with the ID of a party which is a neighbor of ① are random and those tagged with the ID of a non-neighbor of ① are additive secret-shares of the broadcast bit (i.e., the broadcast bit is reconstructed by XOR-ing all the messages tagged "P_j" if $\mathsf{P}_j \notin \mathcal{N}_G(①)$).
- In the third round, each neighbor of ① passes on the messages received in the previous round according to the ID tags (i.e., sends the message tagged "P_j" to P_j if they are neighbors). If there is a party which is not a neighbor of ①, it sends a random message to each of its neighbors instead.

Each party can now compute the sum of the messages received in this third round: neighbors of ① get a random bit, and if there is a party which is not a neighbor of ①, it gets the broadcast bit now.

Claim 8.8. *The protocol is* 1-*IT-THB.*

Proof (sketch). To prove security of the protocol, we provide a sketch of the simulator:

- If the corrupt party is ①, it gets no messages;

- If the corrupt party is a neighbor of ①, it gets the broadcast bit "from 1" in the first round, random messages appropriately tagged in the second round, and then random messages "from" each of its neighbors;
- If the corrupt party is neither ① nor one of its neighbors, then in the third round it gets secret shares (i.e., a pair of messages which XOR to the broadcast bit) of the broadcast bit from its neighbors. □

2. 1-IT-THB *is possible on* {⊠, □, ∧, ⟋}: First of all, if the topology is of type ⊠, then all parties have degree three. Since this is the only topology in the class in which any of the parties have degree three, all parties already know the topology and since there is only one network isomorphic to ⊠, there is nothing to hide about the topology. We therefore only need to consider the class {□, ∧, ⟋} (see below).
3. 1-IT-THB *is possible on* {□, ∧, ⟋}, {□, ⟀, ∧, ⟋}, *and* {□, ⟀, ∧, ⟋}: On these classes, the flooding protocol is already topology hiding. Recall that each party learns from the flooding protocol the distance of each of its neighbors to the broadcaster. One can check that in these classes this information is already contained in the local view of each party:

- For the broadcaster ①, the distance of each of its neighbors is one.
- For a neighbor of ①, the distance of its neighbors except ① is two.
- Otherwise (for a node that is not ① and is not a neighbor of ①), the distance of each of its neighbors is one.

This concludes the proof of Lemma 8.7. □

Acknowledgements

We thank the anonymous reviewers of TCC 2020 for pointing to the connection between anonymous communication and key agreement in [22], as well as the anonymous JoC reviewer. Some of M. Ball's work was done, while the author was at Columbia University, supported in part by an IBM Research PhD Fellowship. M. Ball and T. Malkin's work is supported in part by JPMorgan Chase & Co. as well as the US Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research under award number DE-SC-0001234. E. Boyle's research is supported in part by ISF grant 1861/16, AFOSR Award FA9550-17-1-0069, and ERC Starting Grant 852952 (HSS). Some of R. Cohen's was done, while the author was at Northeastern University, supported in part by NSF grant 1646671. L. Kohl is funded by NWO Gravitation project QSC. This Research of L. Kohl was done while at Technion, supported by ERC Project NTSC (742754). P. Meyer's research is supported in part by ISF grant 1861/16, AFOSR Award FA9550-17-1-0069, and ERC Starting Grant 852952 (HSS); some of his research was done, while the author was a student at École Normale Supérieure de Lyon. Some of T. Moran work was done, while the author was at Spacemesh and Northeastern University. Any views or opinions expressed herein are solely those of the authors listed and may differ from the views and opinions expressed by JPMorgan Chase & Co. or its affiliates. This material is not a product of the Research Department of J.P. Morgan Securities LLC. This material should not be construed as an individual recommendation

for any particular client and is not intended as a recommendation of particular securities, financial instruments or strategies for a particular client. This material does not constitute a solicitation or offer in any jurisdiction.

Appendix A: UC Framework

We present a highly informal overview of the UC framework and refer the reader to [10] 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* $\mathcal{A}$, describing the behavior of the corrupted parties and an *environment* $\mathcal{Z}$, 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 [10], or using the synchronous framework of [23]) 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 $\mathcal{A}$ chooses the corrupted parties at the onset of the protocol) and *adaptive* corruptions (where $\mathcal{A}$ 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* $\mathcal{Z}$, 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 $\mathcal{Z}$ to $\mathcal{F}$ and vice versa. The ideal functionality $\mathcal{F}$ defines the desired behavior 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 UC-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 $\mathcal{A}$ and any PPT environment $\mathcal{Z}$, there exists a PPT ideal-model t-adversary Sim such that the output distribution of $\mathcal{Z}$ 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 $\mathcal{A}$.
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 $\mathcal{A}$, $\mathcal{Z}$, and Sim. In that case, the requirement is for the output distribution of $\mathcal{Z}$ 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 Canetti [10] states the following.

Theorem A.1. *([10], 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 $\mathcal{A}$ and any* PPT *environment $\mathcal{Z}$, there exists a* PPT *adaptive semi-honest t-adversary Sim in the $\mathcal{F}$-hybrid model such that the output distribution of $\mathcal{Z}$ 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 $\mathcal{A}$ in the real model.*

Appendix B: DC-Nets are Topology-Hiding: From THB to THAB

In this section, we show that t-THB implies t-THAB with respect to classes of n-node graphs that are $(t+1)$-connected. Recall that in Sects. 5.1 and 6.2 we showed a separation between t-THB and t-THAB over non-$(t+1)$-connected graphs.
Our starting point is the *Dining-Cryptographers Network* that was introduced by Chaum [12] with the aim of transforming a broadcast primitive into an anonymous broadcast channel, secure in the semi-honest setting. The procedure works as follows on a (public, connected) incomplete network of point-to-point secure channels, where an anonymous party holds as input a bit b_{BC} to be broadcast and all other parties hold a dummy input bit set to zero. Each party starts by sending a random message to each of its neighbors and keeps the sum of all these outgoing messages. Each party then adds to this sum the messages it received in the previous round, one per neighbor. This new sum is now used as a one-time pad to mask the party's input; we call this ciphertext the party's randomized *input*. These randomized inputs sum to the broadcast bit. Note that the key observation is that so long as a passive adversary cannot corrupt a vertex-cut of the graph, the adversary learns nothing (other than the value of b_{BC}) about the honest parties' inputs, even if additionally given the list of randomized inputs. It is therefore safe for the parties to broadcast their randomized inputs to reconstruct the output b_{BC}.

Correctness follows by inspection. Let us recall the high-level idea of why the broadcaster is anonymous. Consider any non-empty proper subset of the vertices $S \subset V$ such that the union of their closed neighborhoods $\mathcal{N}_G[S] := \cup_{v \in S} \mathcal{N}_G[v]$ is connected. At the end of the input randomization phase, the partial sum of the inputs in S (i.e., the indicator of the event "the broadcaster is in S") is secret shared among the randomized inputs of S and the shares the parties in S sent to the parties in $V \setminus S$. Therefore to learn anything about the partial sum of the inputs in S, the adversary has to corrupt the set Z of vertices at the frontier of S, i.e., the vertices in $V \setminus S$ which are neighbors of S. Given any set Z of at most t corruptions, the adversary can isolate the set of players $S = V \setminus Z$, but no other one because the graph is $(t+1)$-connected. For this specific set, the adversary is not learning anything he should not as he knows if the broadcaster is corrupted or not, i.e., in Z or in S.

We make the simple observation that if the underlying broadcast primitive is topology hiding, and if the number and identity of the parties participating in the protocol is publicly known (so they have a way of knowing in which order they can broadcast the randomized inputs), then the anonymous broadcast protocol is topology hiding. Indeed, the input randomization phase is purely information local and cannot leak the topology of the graph, while the reconstruction phase inherits the topology-hiding properties of the broadcast primitive used.

Recall that for $n \in \mathbb{N}$ we denote by $\mathcal{G}^n = \{G \text{ connected graph } : V(G) = [n]\}$ the class of connected graphs with exactly n nodes.

Theorem B.1. *Let $n \in \mathbb{N}$, and let $\mathcal{G} \subseteq \mathcal{G}^n$ be a class of $(t+1)$-connected graphs. Then, the existence of a t-THB protocol with respect to $\mathcal{G}$ implies a t-THAB protocol with respect to $\mathcal{G}$.*

Remark B.2. Note that Theorem B.1 can be strengthened in two ways. Firstly, if a slight variation on the DC-net protocol is run on an arbitrary class of $(t+1)$-connected graphs (where the set of players/vertices is not known *a priori*), then correctness and privacy of input are still guaranteed and the only *leakage*[30] about the topology is the set of players participating in the protocol. In fact it can be strengthened, so only the number of players participating in the protocol is leaked. Secondly, if the parties set their input arbitrarily (rather than all non-broadcasters setting theirs to 0), the DC-net protocol actually constructs a t-secure sum protocol from a broadcast primitive, which is also topology-hiding.

Appendix C: Statistically Secure, Round-Inefficient THB on Oriented-5-Path

In this appendix, we justify our remark that the class $\mathcal{G}_{\text{oriented-5-path}}$ from Sect. 4.1 admits an unconditionally ε-statistically 1-secure $1/\varepsilon$-round topology-hiding broadcast protocol, via a more general lemma. In particular, we observe that the following *delayed-flooding protocol* is ε-statistically secure for any graph class for which the flooding

[30] See Ball et al. [3] for a more in-depth definition of leakage in the context of topology-hiding computation.

Broadcast Protocol π_{BC}

Public Parameters: Let d be an upper bound on the distance of any node from the broadcaster ①, let ε be the statistical security parameter, let $\Delta = d/\varepsilon$ be the upper bound on the random delay, and let $R = d/\varepsilon + d$ be the upper bound on round complexity.

The Protocol:

- `Broadcaster:` Sample r uniformly from $[\Delta]$. Send the broadcast message m in round r to all neighbors. In round R, halt and output m.
- `All other parties:` Upon receipt of broadcast message m, forward to all neighbors that did not send the message. In round R, halt and output m.

Fig. 33. ε-Delayed Flooding Protocol: A simple, inefficient, distance-hiding broadcast protocol.

protocol can be simulated given distance (i.e., only distance need be hidden).[31] The core idea is quite straightforward: If the usual flooding protocol only leaks distance (via the round in which the broadcast bit is received), then by simply having the broadcaster delay flooding for a random number of rounds, this leakage is diluted.

Lemma C.1. *Let $0 < \varepsilon \leq 1$. Let $\mathcal{G}$ be a class of graphs such that each node can always deduce a unique neighbor through which they are connected to the broadcaster ① and let d be an upper bound on distance of any node from ①.*[32]
Then, the ε-Delayed Flooding Protocol (defined in Fig. 33) is an ε-statistically 1-secure topology-hiding broadcast protocol with round complexity $d(1 + 1/\varepsilon)$.

Proof. Let $\Delta = d/\varepsilon$ be the upper bound on the random delay, and let $R = d/\varepsilon + d$ be the upper bound on round complexity, as defined in Fig. 33.
We begin by observing that the ε-Delayed Flooding Protocol will deliver the broadcast message to all nodes because for any choice of delay $r \in [\Delta]$, since $r + d \leq R$. (Recall d is an upper bound on the distance from the broadcaster.) So, the protocol is, in fact, perfectly correct.
It remains to show ε-statistical security. Let v be the corrupt node with neighborhood $\mathcal{N}(v)$. By the properties of $\mathcal{G}$, this means that there exists a fixed $u \in \mathcal{N}(v)$ such that u is a bridge to ① in all graphs $G \in \mathcal{G}$ such that $\mathcal{N}_G(v) = \mathcal{N}(v)$.
Because the protocol follows the same message pattern as the naïve flooding protocol and $\mathcal{G}$ consists of trees, we can represent the distribution of the view of v on any graph $G \in \mathcal{G}$ as the tuple $p(G) = (p_1(G), \ldots, p_R(G))$ where p_i is the probability v receives the broadcast message in round i (for $i \in [R]$) and $p_\perp$ is the probability v sees nothing. Observe that if v is distance d_v from ① in any graph $G \in \mathcal{G}$, the view of v in G is simply the broadcast message at round $r_v = r + d_v$, where r is a random variable distributed uniformly over $[\Delta]$. Thus, we have that $p_1(G) = \cdots = p_{d_v}(G) = p_{\Delta-(d-d_v)+1}(G) = \cdots = p_\Delta(G) = 0$, and that $p_{d_v+1}(G) = \cdots = p_R(G) = 1/(\Delta - (d - d_v))$.

[31] We actually show a slightly restricted case for simplicity, but the result can easily be extended.

[32] In other words, $\mathcal{G}$ is a set of trees on some potential vertex set V such that for every $v \in V$, $d_G(v, 1) \leq d$ if $v \in V(G)$ (for all $G \in \mathcal{G}$) and if $\mathcal{N}_G(v) = \mathcal{N}_H(v)$ for $H, G \in \mathcal{G}$, then there exists a (unique) $u \in \mathcal{N}_G(v) = \mathcal{N}_H(v)$ that disconnects 1 and v in both G and H.

The simulator Sim works by sampling $r \leftarrow [\Delta]$ and sending the broadcast message m in round $r + d/2$ from the neighbor u, specified above. We can write the distribution of the simulated view as the tuple $\tilde{p} = (\tilde{p}_1, \ldots, \tilde{p}_R, \tilde{p}_\perp)$ where $\tilde{p}_1 = \cdots = \tilde{p}_{d/2} = \tilde{p}_{R-d/2+1} = \cdots = \tilde{p}_R = 0$ and $\tilde{p}_{d/2+1} = \cdots = \tilde{p}_{R-d/2} = 1/\Delta$.
Thus, we can explicitly compute the statistical difference as $\frac{|d/2-d_v|}{\Delta} \leq \frac{d/2}{\Delta} \leq \varepsilon/2$. We can therefore deduce that under the simulation notion we have security $\varepsilon/2$ and under the indistinguishability-based definition (Definition 2.4), we have statistical security ε. □

References

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

[2] A. Akavia, R. LaVigne, and T. Moran. Topology-hiding computation on all graphs. In *37th Annual International Cryptology Conference (CRYPTO), part I* (2017), pp. 447–467

[3] M. Ball, E. Boyle, T. Malkin, and T. Moran. Exploring the boundaries of topology-hiding computation. In *37th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part III* (2018), pp. 294–325

[4] M. Ball, E. Boyle, R. Cohen, T. Malkin, and T. Moran. Is information-theoretic topology-hiding computation possible? In *Proceedings of the 17th Theory of Cryptography Conference(TCC), part I* (2019), pp. 502–530

[5] M. Ball, E. Boyle, R. Cohen, L. Kohl, T. Malkin, P. Meyer, and T. Moran. Topology-hiding communication from minimal assumptions. In *Proceedings of the 18th Theory of Cryptography Conference(TCC), part II* (2020), pp. 473–501

[6] D. Beaver. Foundations of secure interactive computing. In *10th Annual International Cryptology Conference (CRYPTO)* (1991), pp. 377–391

[7] D. Beaver. Precomputing oblivious transfer. In *14th Annual International Cryptology Conference (CRYPTO)* (1995), pp. 97–109

[8] M. Ben-Or, S. Goldwasser, and A. 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)* (1988), pp. 1–10

[9] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[10] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)* (2001), pp. 136–145

[11] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[12] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[13] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)* (1988), pp. 11–19

[14] D. Dolev. The Byzantine generals strike again. *J. Algorithms*, 3(1):14–30, 1982.

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

[16] S. Even and R. E. Tarjan. Computing an *st*-numbering. *Theoretical Computer Science*, 2(3):339–344, 1976.

[17] S. Even and R. E. Tarjan. Corrigendum: Computing an *st*-numbering. *Theor. Comput. Sci.*, 4(1):123, 1977

[18] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)* (2000), pp. 325–335

[19] O. Goldreich, S. Micali, and A. 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)* (1987), pp. 218–229

[20] I. Haitner, K. Nissim, E. Omri, R. Shaltiel, and J. Silbak. Computational two-party correlation: A dichotomy for key-agreement protocols. In *Proceedings of the 59th Annual Symposium on Foundations of Computer Science (FOCS)* (2018), pp. 136–147

[21] M. Hirt, U. Maurer, D. Tschudi, and V. Zikas. Network-hiding communication and applications to multi-party protocols. In *36th Annual International Cryptology Conference (CRYPTO), part II* (2016), pp. 335–365

[22] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography from anonymity. In *Proceedings of the 47th Annual Symposium on Foundations of Computer Science (FOCS)* (2006), pp. 239–248

[23] J. Katz, U. Maurer, B. Tackmann, and V. Zikas. Universally composable synchronous computation. In *Proceedings of the 10th Theory of Cryptography Conference(TCC)* (2013), pp. 477–498

[24] J. Kilian. A general completeness theorem for two-party games. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)* (1991), pp. 553–560

[25] L. Lamport, R. E. Shostak, and M. C. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[26] R. LaVigne, C. L. Zhang, U. Maurer, T. Moran, M. Mularczyk, and D. Tschudi. Topology-hiding computation beyond semi-honest adversaries. In *Proceedings of the 16th Theory of Cryptography Conference(TCC), part II* (2018), pp. 3–35

[27] R. LaVigne, C. L. Zhang, U. Maurer, T. Moran, M. Mularczyk, and D. 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* (2020), pp. 215–245

[28] Y. Lindell, E. Omri, and H. Zarosim. Completeness for symmetric two-party functionalities: Revisited. *Journal of Cryptology*, 31(3):671–697, 2018.

[29] T. Moran, I. Orlov, and S. Richelson. Topology-hiding computation. In *Proceedings of the 12th Theory of Cryptography Conference(TCC), part I* (2015), pp. 159–181

[30] M. C. Pease, R. E. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[31] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS)* (1989), pp. 73–85

[32] R. E. Tarjan. Two streamlined depth-first search algorithms. *Fundamenta Informaticae*, 9(1):85–94, 1986.

[33] A. C. Yao. Protocols for secure computations (extended abstract). In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS)* (1982), pp. 160–164