Abstract—The question of whether and how mutually distrusting parties can collaborate is a central theme in cryptography. The goal of secure function computation is to ensure that parties may correctly compute functions of their data without learning additional information. A remarkable result of Ben Or, Goldwasser, and Wigderson from 1988 shows that it is possible for parties connected by pairwise, private, noise-free links to compute functions with zero error and perfect information theoretic security provided the number of parties who may collude meets a certain threshold; specifically, if the colluders form a strict minority for the honest-but-curious model and they are strictly less than a third for the malicious model. In this paper we provide basic lowerbounds on the amount of communication required to compute with zero-error and perfect security in a three-party setting under the honest-but-curious model.

I. INTRODUCTION

The question of whether and how mutually distrusting parties can collaborate is a central theme in cryptography. Secure multiparty computation, pioneered by the seminal works of Shamir, Rivest, Adleman [1], Rabin [2], Blum [3], Yao [4] and others, investigates the question of how a set of mutually distrusting parties can compute functions of data they hold without revealing to each other anything more than the function values they compute. Put another way, the goal is to emulate a centrally trusted party who accepts data from the parties, evaluates the functions and provides the answers to the parties.

While the initial efforts were primarily based on computational limitations of the adversaries, a remarkable result by Ben Or, Goldwasser, and Wigderson [5] (BGW) and Chaum, Crépeau, and Damgård [6] showed that information theoretically secure function computation is possible between parties connected by pairwise, private, noise-free links when only a strict minority may collude in the honest-but-curious model (and a strictly less than one-third minority may collude in the malicious model). Another line of work on information theoretically secure computation relies on noisy channels between (or distributed sources at) the computing parties [7]. In this paper we focus on the setting of BGW where such stochastic resources are unavailable.

An important question, which, to the best of our knowledge, has received little attention is deriving lowerbounds on how much communication is needed to compute securely. Lowerbounds on the amount of communication required to compute, not necessarily securely, has been the subject of study of communication complexity [8], [9] in computer science and interactive function computation, [10], for instance, in information theory. In this paper, we derive some basic lowerbounds on communication requirements for the secure computation problem in Figure 1. We consider the honest-but-curious setting where the parties do not deviate from the protocol, but may attempt to obtain additional information about other parties inputs/outputs at the end of the protocol from everything they have access to. By the BGW result [5], we know that any function is computable with zero-error and securely such that no party acting individually may gain any additional information in an information theoretic sense.

The lowerbounds in this paper are based on basic cut-set like arguments. We will present improved bounds in [11]. Section II sets up the problem. In Section III, we present the main communication lowerbound result of this paper. We observe that our lowerbound is tight for the XOR function. In the following two sections, the lowerbounds are worked out for two specific functions: REMOTE ERASURE and AND. We also compare the lowerbounds against protocols for securely computing these functions. We close with a discussion of open issues and on going work in Section VI.

A. Notation

For a discrete random variable $X$, we will denote its distribution (probability mass function) by $p_X(x)$ and its
realization by \(x\). The subscript of \(p_x\) will be dropped where it is clear from the context. For a sequence of random variables \(X_1, X_2, \ldots, X_n\), we denote by \(X^n\) the vector \((X_1, \ldots, X_n)\).

We denote a directional link from node 1 to 2 by \(\rightarrow\), and an undirected link between nodes 1 and 2 by \(12\). Thus, (as will be defined below), \(M_{12}\) will stand for the messages sent by node 1 to node 2 and \(M_{21}\) for messages exchanged between nodes 1 and 2, i.e., \(M_{12}\) is the transcript of communication over the bidirectional link between 1 and 2.

All logarithms will be to the base 2. The binary entropy function is denoted by \(H_2(p) = -p \log p - (1 - p) \log(1 - p)\), \(p \in (0, 1)\).

II. PROBLEM STATEMENT

We consider the following three-party secure computation problem. Alice (party-1) and Bob (party-2) have data in the form of random variables \(X \in \mathcal{X}\) and \(Y \in \mathcal{Y}\), respectively, where the alphabets \(\mathcal{X}\) and \(\mathcal{Y}\) are finite. In this paper, we confine our attention to the case where \(X\) and \(Y\) are independent random variables. Let \(X \sim p_X(x)\) and \(Y \sim p_Y(y)\). Charlie (party-3) is interested in computing the function \(f(X, Y)\), where \(f: \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}\). Without loss of generality, we assume that \(p_X(x) > 0, p_Y(y) > 0\), for all \(x \in \mathcal{X}, y \in \mathcal{Y}\), and that for every \(z \in \mathcal{Z}\), there is a pair \((x, y) \in \mathcal{X} \times \mathcal{Y}\) such that \(z = f(x, y)\).

Between every pair of parties, there is a noise-free, bidirectional link which is private and secure from the other party. The parties engage in a protocol where they exchange messages with each other over these links over several rounds. The parties are assumed to have access to private randomness, i.e., randomness which is independent between the parties and also independent of the data. We assume that no common randomness (pairwise or three-way) is available (in fact, our communication lowerbounds hold even when a three-way common random string is available; see Section VI) Message \(M_{ij,t}\), from party-\(i\) to party-\(j\) in round \(t\) is a function of all the messages party-\(i\) has received up to the previous round \(t - 1\), party-\(i\)’s data (if any), and its private randomness. The message \(M_{ij,t}\) must be a string of bits (possibly even the empty string). We further restrict that \(M_{ij,t}\) be a codeword in a (potentially random) prefix-free code such that the prefix-free code itself is determined (with probability 1) by the messages \((M_{ij,t}^{t-1}, M_{ji,t}^{t-1})\) previously exchanged on the \(ij\) link (in both directions). A consequence of this restriction is that the receiving node can parse the end of each message without the need for an additional end-of-message symbol which we do not allow. We require that for a valid protocol, in each link, the number of rounds after which the link remains unused (i.e., only empty strings are exchanged) must be finite with probability 1.

Another consequence of our restriction that the code be determined by previously exchanged messages is that the participating nodes will know when the exchanges over the link connecting them has come to an end. Thus, we allow the number of rounds and the (potentially variable-length) code used for each round to be random. The view \(V_i\) of party-\(i\) is the collection of everything it has access to at the end of the protocol. Specifically, its data (if any), all the messages exchanged over the two links it is part of, and its private randomness. Charlie outputs a \(Z\) which is a function of his view \(V_3\).

The protocol is said to be zero-error if \(Z = f(X, Y)\) with probability 1. A protocol is said to be secure against Alice if Alice learns nothing about Bob’s data, i.e., \(I(Y; V_2) = 0\). Similarly, a protocol is secure against Bob if he learns nothing about Alice’s data, i.e., \(I(X; V_2) = 0\), and it is secure against Charlie if he learns nothing more about the inputs of Alice and Bob than he can infer from his ideal output \(Z' = f(X, Y)\), i.e., \(I(X, Y; V_3|Z') = 0\). A protocol is said to be perfectly secure (or simply, secure) if it is secure against Alice, secure against Bob, and secure against Charlie.

In this paper, we assume that the parties are honest-but-curious (or semi-honest), i.e., they execute the protocol faithfully, but at the end of the execution they may examine their views and the goal is to ensure that they do not derive additional information than their input/output. The parties are assumed to not collude. This is consistent with [5] when specialized to this three-party setting. In this paper, we focus only on the case of zero-error and perfect security, the same setting as [5].

It is easy to see that we can summarize the joint distribution of the inputs, output, and the messages exchanged as

\[
p(x, y, (m_{ij,t})_{ij,t}, z) = p(x)p(y) \left( \prod_{t=1}^{r} \prod_{i=1}^{r} p(m_{ij,t} | x_i, m_{i,j-1}^{t-1}) \right)p(z|m_{3,t}^{n}),
\]

where we denote by \(M_{ij,t} = (M_{ij,t})_{ji,t}\) the messages sent by party-\(i\) in round \(t\), and by \(M_{i,t} = (M_{ij,t}, M_{ji,t})_{ji,t}\) the messages sent and received by party-\(i\) in round \(t\). Also, \(x_i\) is the data at party-\(i\), i.e., \(x_1 = x, x_2 = y, \text{ and } x_3 = 0\). Note that the number of rounds \(r\) depends on the argument through \((x, y, (m_{ij,t})_{ij,t})\), i.e., after \(r\), the rest of the messages are empty with conditional probability 1 conditioned on \((x, y, m_{i,j-1})_{ji,t}\). Here, the effect of private randomness is accounted for by the conditional probabilities \(p(m_{ij,t} | x_i, m_{i,j-1}^{t-1})\) and \(p(z | m_{3,t}^{n})\). Let us denote by \(M_{ij}\) the collection of messages exchanged over link between parties \(i\) and \(j\) in either direction. Similarly, let \(M_i\) be the set of all messages exchanged (in either direction) over the two links which party-\(i\) participates in. The zero-error and perfect security conditions can be stated equivalently as

\[
p(z|x, y) = 1 = f(x, y), \quad \forall x, y, z,
\]

\[
I(Y; M_1|X) = I(X; M_2|Y) = I(X, Y; M_3|Z) = 0.
\]

\[1\]Our results for secure computation readily extend to joint distributions which have full-support as we show in [11].

\[2\]While we allow this generality, the protocols we provide have a deterministic number of rounds with deterministic message lengths. The generality is in order to prove impossibility results (communication lowerbounds) with wide applicability.
As stated earlier, the fact that the above goal can be achieved for any \( p_X, p_Y \) and \( f \) can be inferred from the results in [5]. Our interest is in providing bounds on the amount of communication needed. For each link \( ij \), we will derive lower bounds on the entropy \( H(M_{ij}) \) of the messages exchanged over the bidirectional link between party-\( i \) and party-\( j \) by any zero-error, perfectly secure protocol. As we argue below, these bounds will lowerbound the expected number of bits exchanged over the link \( ij \) in either direction. We are interested in lowerbounds for \( E[L_{ij}] \). We have

\[
H(M_{ij}) = \sum_{t=1}^{\infty} H(M_{ij,t} | M_{ij}^{t-1}, M_{ji}^{t-1}) \\
\leq \sum_{t=1}^{\infty} H(M_{ij,t} | M_{ij}^{t-1}, M_{ji}^{t-1}) + H(M_{ji,t} | M_{ij}^{t-1}, M_{ji}^{t-1}) \\
= \sum_{t=1}^{\infty} E[L_{ij,t}] + E[L_{ji,t}] \\
= E[L_{ij}],
\]

where (a) follows from the fact that the prefix-code of which \( M_{ij,t} \) is a codeword, is a function of the conditioning random variables \( (M_{ij}^{t-1}, M_{ji}^{t-1}) \), and hence the conditional entropy \( H(M_{ij,t} | M_{ij}^{t-1}, M_{ji}^{t-1}) \) is no larger than \( E[L_{ij,t}] \) (by Kraft’s inequality and non-negativity of Kullback-Leibler divergence); similarly for the second term.

For canonical examples, we will give protocols which have deterministic number of rounds with deterministic message lengths. Figure 2 shows an error-free, secure protocol which computes the bit-wise XOR of vectors of bits \( X^n \) and \( Y^n \). In Section III, we will show that its communication requirement of 1 bit per bit-wise XOR computation on each link is in fact optimal.

Algorithm 1: Secure Computation of XOR

Require: Alice & Bob have input bit strings \( X, Y \in \{0,1\}^n \).
Ensure: Charlie securely computes the (bit-wise) XOR \( Z = X \oplus Y \).

1. Charlie samples \( n \) i.i.d. uniformly distributed bits \( K \) from its private randomness; sends it to Alice as \( M_{31} = K \).
2. Alice sends \( M_{21} = X \oplus M_{31} \) to Bob.
3. Bob sends \( M_{22} = Y \oplus M_{32} \) to Charlie.
4. Charlie outputs \( Z = K \oplus M_{32} \).

Fig. 2. An optimal algorithm to compute XOR. The algorithm requires 1 bit to be exchanged per bit-wise XOR computation over each of the three links. We show this to be optimal in Example 3.1 of Section III.

We will find it useful to also consider the corresponding three-party secure sampling problem, where, instead of being provided with data \( X, Y \) in advance, the three parties are interested in sampling from the joint distribution \( p_X(x)p_Y(y)1_{z=f(x,y)} \). The parties engage in a protocol as before, but without any data. The only source of randomness is their private randomness. At the end of the protocol, each party outputs its sample as a function of its view. The protocol is said to be zero-error if the joint distribution of the outputs is the same as the desired one above. Such a protocol is said to be perfectly secure if none of the parties can infer anything about the other parties’ outputs other than what they can from their own outputs. Specifically, the joint distribution is now

\[
p(x, y, (m_{ij,t}, m_{ji,t}, z)) = \left( \prod_{i=1}^{r} \prod_{j=1}^{s} p(m_{ij,t}|m_{ji,t}^{t-1}) \right) p(x|m_{i1})p(y|m_{21})p(z|m_{31}),
\]

and the zero-error and perfect security conditions are

\[
p(x, y, z) = p_X^*(x)p_Y^*(y)1_{z=f(x,y)}, \quad \forall x, y, z,
\]

Clearly, any error-free, secure protocol for the secure computation problem yields an error-free, secure protocol for the secure sampling problem with the same communication requirements. Specifically, Alice and Bob may sample \( X \) and \( Y \) from their private randomness and use these as the inputs to the secure computation protocol. Some of our lowerbounds will be derived for the secure sampling problem, which, by the above reasoning, will imply the same lowerbounds for the secure computation problem.

III. LOWERBOUNDS ON COMMUNICATION REQUIRED

Theorem 3.1: Suppose the function \( f \) satisfies the following conditions:

1. For every pair \( x, x' \in X \) such that \( x \neq x' \), there is a \( y \in Y \) satisfying \( f(x, y) \neq f(x', y) \). Similarly, for every \( y \neq y' \), both in \( Y \), there is an \( x \in X \) such that \( f(x, y) \neq f(x, y') \).

2.1 There is no non-trivial partition \( X = X_1 \cup X_2 \) (i.e., \( X_1 \cap X_2 = \emptyset \) and neither \( X_1 \) nor \( X_2 \) is empty), such that if \( Z_k = \{ f(x, y) : x \in X_k, y \in Y \}, k = 1, 2 \), their intersection \( Z_1 \cap Z_2 \) is empty.

2.2 Similarly, there is no non-trivial partition \( Y = Y_1 \cup Y_2 \) such that \( \{ f(x, y) : x \in X, y \in Y_1 \} \cap \{ f(x, y) : x \in X, y \in Y_2 \} \) is empty.

For \( z \in Z \), let \( S_z = \{ (x, y) \in X \times Y : f(x, y) = z \} \). Any error-free, secure protocol for secure sampling must satisfy

\[
H(M_{12}) \geq \max \left( \log |X|, \log |Y|, \max \log |S_z| \right),
\]

\[
H(M_{23}) \geq \max \left( \log |Y|, \log |Z| \right)
\]

\[
H(M_{31}) \geq \max \left( \log |X|, \log |Z| \right).
\]

The same bounds also hold for any error-free, secure protocol for secure computation.

Remark 1: Condition 1 in the Theorem 3.1 is without loss of generality. It simply requires that the no two rows (or columns) of the truth-table of the function \( f \) be identical. If
this is not the case, and for \( x \neq x' \), the rows are identical, without loss of generality, we may redefine the problem by merging the two rows by absorbing \( x' \) into \( x \). However, Condition 2, is not without loss of generality. Section IV shows a non-trivial example of a function which does not satisfy this condition. However, many functions of interest (e.g., XOR, AND, and all non-trivial, binary-output functions such as DISJ, EQ, MED, IP, GT, MAX; see [9, Chapter 1] for definitions of these boolean functions) do satisfy this condition.

Example 3.1 (Secure computation of XOR): Alice’s and Bob’s data are independent \( n \)-length bit strings such that all possible strings have a non-zero probability. Thus \( X = Y = \{0,1\}^n \). Figure 2 gives a protocol for secure computation of XOR which requires \( n \) bits to be communicated over each bidirectional link, i.e., \( L_{12} = L_{23} = L_{31} = n \). It is easy to see that this problem satisfies the conditions of Theorem 3.1, and hence we have \( H(M_{12}), H(M_{23}), H(M_{31}) \geq n \), which implies that the protocol is optimal.

A. Proof of Theorem 3.1

The proof idea is to first show the following lemmas for secure sampling (and, therefore, also for secure computation); proofs are in the appendix:

(i) Lemma 3.2: Condition 1, zero-error, and secrecy against Alice imply that examining the cut isolating Alice from Bob and Charlie must reveal \( X \), i.e., \( H(X|M_{12}, M_{31}) = 0 \). Similarly, \( H(Y|M_{12}, M_{23}) = 0 \).

(ii) Lemma 3.3: Condition 2.1, secrecy against Alice, and secrecy against Charlie imply that Alice and Bob’s data \( X, Y \) (and Charlie’s output \( Z \)) must be independent of the messages \( M_{31} \) on the link between Alice and Charlie, i.e., \( I(X, Y, Z; M_{31}) = 0 \). Similarly, by condition 2.2, secrecy against Bob, and secrecy against Charlie, \( I(X, Y, Z; M_{23}) = 0 \).

(iii) Lemma 3.4: Secrecy against Alice and secrecy against Bob imply that \( I(X, Y; M_{12}) = 0 \).

We will make use of the following lemma on error-free, perfectly secure communication protocol. Consider jointly distributed discrete random variables \( U, V, W \) where \( U \) has the interpretation of the message, \( V \) of the secret key and \( W \) of the cipher text. Let the following three conditions be satisfied

\[
\begin{align*}
I(U; V) &= 0, \\
H(U|V, W) &= 0, \\
I(U; W) &= 0,
\end{align*}
\]

i.e., (i) the message \( U \) and secret key \( V \) are independent, (ii) the message \( U \) can be recovered without error given the secret key \( V \) and the cipher text \( W \), and (iii) the cipher text \( W \) is independent of the message \( U \). Then the lemma states that the entropy of the cipher text \( W \) and the entropy of the secret key \( V \) must be at least as large as \( \log |U| \), logarithm of the cardinality of \( U \) (assuming \( p(u) > 0 \), for all \( u \in |U| \)).

Lemma 3.5 ([12]): Let \( U, V, W \) be jointly distributed discrete random variables satisfying (1)-(3) and such that the p.m.f. of \( U \) has full support over its alphabet \( U \). Then \( H(V) \geq \log |U| \) and \( H(W) \geq \log |U| \).

For \( (X, M_{31}, M_{12}) \), by Lemmas 3.2-3.4, we have

\[
\begin{align*}
I(X; M_{31}) &= 0, \\
H(X|M_{12}, M_{31}) &= 0, \\
I(X; M_{12}) &= 0.
\end{align*}
\]

Applying Lemma 3.5,

\[
H(M_{31}) \geq \log |X|, \quad H(M_{12}) \geq \log |X|.
\]

Similarly, by applying Lemma 3.5 to \( (Y, M_{23}, M_{12}) \), we have

\[
H(M_{23}) \geq \log |Y|, \quad H(M_{12}) \geq \log |Y|.
\]

Since \( H(Z|M_{31}, M_{23}) = 0 \), we may also apply Lemma 3.5 to \( (Z, M_{31}, M_{23}) \) to get

\[
H(M_{12}) \geq \log |Z|, \quad H(M_{12}) \geq \log |Z|.
\]

To derive \( H(M_{12}) \geq \log |S_z| \), \( z \in Z \), we consider the conditional p.m.f. of \( (X, Y), (M_{13}, M_{23}), M_{12} \) conditioned on \( Z = z \). Let us notice that, by secrecy against Charlie,

\[
I(X, Y; M_{31}, M_{23}|Z = z) = 0
\]

Furthermore, since \( H(X|M_{13}, M_{12}) = H(Y|M_{23}, M_{12}) = 0 \) and \( Z = f(X, Y) \),

\[
H(X, Y|M_{13}, M_{23}, M_{12}|Z = z) = 0.
\]

Also, \( I(X, Y; M_{12}) = 0 \) and \( Z = f(X, Y) \) implies that

\[
I(X, Y; M_{12}|Z = z) = 0.
\]

Thus, by Lemma 3.5

\[
H(M_{12}|Z = z) \geq \log |S_z|.
\]

However, since \( M_{12} \) is independent of \( Z \) (which is a consequence of \( I(X, Y; M_{12}) = 0 \) and \( Z = f(X, Y) \)), we may conclude that \( H(M_{12}) \geq \log |S_z| \).

IV. Secure Computation of Controlled Erasure

The controlled erasure function is shown below. Alice’s input \( X \) acts as the “control” which decides whether Charlie receives an erasure (\( \Delta \)) or Bob’s input \( Y \).

\[
\begin{array}{ccc}
\text{y} & 0 & 1 \\
0 & \Delta & \Delta \\
1 & 0 & 1
\end{array}
\]

Notice that Charlie always find out Alice’s control bit, but does not learn Bob’s bit when it is erased. This function does not satisfy condition 2.1 of Theorem 3.1.
Algorithm 2: Secure Computation of CONTROLLED ERASURE

Require: Alice & Bob have input bits $X^n, Y^n \in \{0, 1\}^n$.
Ensure: Charlie securely computes the CONTROLLED ERASURE function

\[ Z_i = f(X_i, Y_i), \quad i = 1, \ldots, n. \]

1. Bob samples $n$ i.i.d. uniformly distributed bits $K^n$ from his pvt. randomness; sends it to Alice as $M_{21,1} = K^n$. Bob sends to Charlie his input $Y^n$ masked (bit-wise) with $K^n$ as $M_{23,1} = Y^n \oplus K^n$.
2. Alice sends his input $X^n$ to Charlie and also the sequence of key bits $K_i$ corresponding to the locations where his input $X_i$ is 1.
3. Charlie outputs

\[ M_{12,2} = X^n, (K_i)_{1 \leq i \leq n}. \]

Figure 3 gives an algorithm for securely computing this function on each location of strings of length $n$. Bob sends his input string to Charlie under the cover of a one-time pad and reveals the key used to Alice. Alice sends his input to Charlie. He also sends to Charlie those key bits he received corresponding to the locations where there is no erasure (i.e., where his input bit is 1). When $X \sim \text{Bernoulli}(1/2)$ and $Y \sim \text{Bernoulli}(1/2)$, both i.i.d., the expected message lengths per bit are $E[L_{21}] \leq \frac{2n}{3} + 1$, $L_{12} = n$, and $L_{23} = n$.

We show that these are asymptotically optimal by showing the following lowerbounds: $H(M_{31}) \geq \frac{3n}{2}$, $H(M_{12}) \geq n$, and $H(M_{23}) \geq n$.

Theorem 4.1: For $X \sim \text{Bernoulli}(1/2)$ and $Y \sim \text{Bernoulli}(1/2)$, both i.i.d., secure computation of CONTROLLED ERASURE over blocklength $n$ requires

\[
H(M_{31}) \geq \frac{3n}{2}, \\
H(M_{12}) \geq n, \text{ and } H(M_{23}) \geq n.
\]

Proof: The function satisfies conditions 1 and 2.2 of Theorem 3.1. Hence, making use of Lemmas 3.2-3.4 to apply Lemma 3.5 to $(Y, M_{23}, M_{12})$ as in the proof of Theorem 3.1, we arrive at

\[
H(M_{12}) \geq n \text{ and } H(M_{23}) \geq n.
\]

To show the bound on $H(M_{31})$, we first note that for any zero-error protocol, we have $H(Z^n; M_{31}, M_{23}) = 0$. By Lemma 3.3, we have $I(Z^n; M_{23}) = 0$. Hence,

\[
H(M_{31}) \geq I(Z^n; M_{31}|M_{23}) = I(Z^n; M_{31}, M_{23}) - I(Z^n; M_{23}) = H(Z^n) - \frac{3n}{2}.
\]

Remark: Notice that even if $Y^n$ were compressible, specifically, suppose $Y \sim \text{Bernoulli}(q)$, i.i.d. with $0 < q < 1/2$, we are unable to take advantage of this in our protocol. In [11], we argue that the above lowerbounds continue to hold implying that this limitation is fundamental. In fact, we show there that the protocol is optimal even if $X \sim \text{Bernoulli}(p)$ i.i.d., $0 < p < 1$, when step 2 is modified to send the $X^n$ sequence compressed using Huffman/Lempel-Ziv codes.

V. SECURE COMPUTATION OF AND

Figure 4 shows an algorithm for securely computing AND. The algorithm requires two bits to be exchanged over the Alice-Charlie (31) and Bob-Charlie (23) links and three bits over the Alice-Bob (12) link per AND computation.

Algorithm 3: Secure Computation of AND

Require: Alice has two input bits $X \& Bob$ has a bit $Y$.
Ensure: Charlie securely computes the AND $Z = X \land Y$.

1. Charlie samples two indep., uniformly distributed bits $K_0, K_1$ from his pvt. randomness; sends it to Alice as $M_{31,1} = (K_0, K_1)$.
2. Alice samples a uniform bit $B$ from his pvt. randomness and computes

\[
M^{(0)} = K_B, \\
M^{(1)} = X \oplus K_B, \\
M^{(0)} = Y \oplus B, \text{ and sends to } C
\]

3. Alice sends to Bob $M_{12,2} = (B, M^{(0)}, M^{(1)})$.
4. Bob computes $M = M^{(0)}, C = Y \oplus B$, and sends to Charlie

\[
M_{23,3} = (M, C).
\]

5. Charlie outputs $Z = M \oplus K_C$.

Figure 4. An algorithm to compute AND. The algorithm requires two bits to be exchanged over the Alice-Charlie (31) and Bob-Charlie (23) links and three bits over the Alice-Bob (12) link per AND computation.

For $X, Y \sim \text{Bernoulli}(0.5)$ i.i.d., using Theorem 3.1 gives $H(M_{12}) \geq \log 3 \approx 1.585$ and $H(M_{31}), H(M_{23}) \geq 1$. In fact, we show in [11] that $H(M_{31}), H(M_{23}) \geq \log 3$. There we also show that this also holds per computation for $X^n \sim \text{Bernoulli}(p)$ i.i.d. and independent of $Y^n \sim \text{Bernoulli}(q)$ i.i.d., $p, q \in (0, 1)$.

VI. DISCUSSION

Theorem 3.1 gives some basic lowerbounds on the communication required to compute perfectly securely and with zero error in the three party problem of Section II. In [11]
we extend the applicability of these basic lowerbounds and also prove stronger lowerbounds using monotones for secure computation from [13] [14]. Apart from further strengthening the bounds, there are several natural directions to consider. We list a few below.

- **Vanishing error/information leakage:** The lowerbounds have been proved for zero-error and perfectly secure computation. These results also extend to the case where probability of error and information leakage (measured as conditional mutual information) are allowed to be positive, but arbitrarily small. However, they do not apply for settings where the probability of error and/or information leakage is only required to tend to 0 as blocklength goes to infinity. This is part of our ongoing work.

- **Common randomness, dependent data, other forms of correlations:** Our lowerbounds assume that pairwise common randomness which is secret from the other party is unavailable. Any such common randomness needs to be generated by the parties by communicating over the links thereby counting towards the communication. Having access to such common randomness can lead to strictly lower amount of communication than given by our protocols (e.g., step 1 in the algorithm in Figure 2 for XOR can be eliminated). However, the lowerbounds shown above continue to be lowerbounds on the expected number of bits communicated even when all three parties have access to a common random string since the lowerbounds for entropy and proofs follows verbatim when all quantities are conditioned on the common random string.

More generally, the data maybe dependent or the parties may have other forms of dependence available, say in the form of a noisy channel between them or through observations of a distributed source which is independent of the data. In such cases, in general, they may be able to do even better than what [5] suggests, e.g., 2 party secure computation may become possible [7], [15].

- **More nodes/inputs/outputs with potential randomized functions:** Specifically, lowerbounds for the class of problems in [5].

- **Malicious parties:** Our lowerbounds are derived for semi-honest adversaries. Improved lowerbounds are possible for protocols which must handle malicious parties. Indeed, it is known that secure computation is not always possible with security against malicious parties in the three party setting [5].

APPENDIX

**Proof of Lemma 3.2:** We will only show \( H(X|M_{12}, M_{13}) = 0 \), the other one is similarly proved. We apply a cut-set argument. Consider the cut isolating Alice from Bob and Charlie.

We need to show that for every \( m_{12}, m_{31} \) with \( p(m_{12}, m_{31}) > 0 \), there is a (necessarily unique) \( x \in X \) such that \( p(x|m_{12}, m_{31}) = 1 \). Suppose, to the contrary, that we have a zero-error protocol secure against Alice resulting in a p.m.f. \( p(x, y, z, m_{12}, m_{31}) \) such that there exists \( x, x' \in X, x \neq x' \) and \( m_{12}, m_{31} \) satisfying \( p(m_{12}, m_{31}|x, p(m_{12}, m_{31}|x') > 0 \). For this \( x, x' \), by Condition 1 there is a \( y \in Y \) such that \( z \neq z' \), where \( z = f(x, y) \) and \( z' = f(x', y) \).

(i) The definition of a protocol implies that \( p(x, y, z, m_{12}, m_{31}) \) can be written as \( p(x, y)p(m_{12}, m_{31}|x, y)p(z|m_{12}, m_{31}, y) \).

(ii) Secrecy against Alice implies that \( p(m_{12}, m_{31}|x, y, z) = p(m_{12}, m_{31}|x) \).

(iii) Using (ii) in (i), we get \( p(x, y, z, m_{12}, m_{31}) = p(x, y)p(m_{12}, m_{31}|x, y)p(z|m_{12}, m_{31}, y) \).

(iv) Correctness and (ii) implies that we can also write \( p(x, y, z, m_{12}, m_{31}) = p(x, y)p(m_{12}, m_{31}|x) \).

(v) Since \( p(x, y)p(m_{12}, m_{31}|x) > 0 \) and \( p(z|x, y) = 1 \), from (iii) and (iv), we get \( p(z|m_{12}, m_{31}, y) = p(z|x, y) = 1 \).

Applying the above arguments to \( (x', y, z', m_{12}, m_{31}) \), we get \( p(z'|m_{12}, m_{31}, y) = p(z'|x', y) = 1 \) leading to the contradiction \( p(z|m_{12}, m_{31}, y) = p(z'|m_{12}, m_{31}, y) = 1, z \neq z' \), and \( p(m_{12}, m_{31}, y) > 0 \) (which follows from (ii)).

**Proof of Lemma 3.3:** To show \( I(X; Y, Z; M_{31}) = 0 \), we need only show that \( I(X; M_{31}) = 0 \) since

\[
I(X; M_{31}) = 0
\]

\[
\implies I(XZ; M_{31}) = I(X; M_{31}) + I(Z; M_{31}|X) = 0
\]

\[
\implies I(Z; M_{31}) = 0
\]

\[
\implies I(XYZ; M_{31}) = I(Z; M_{31}) + I(XY; M_{31}|Z) = 0,
\]

where (a) follows from secrecy against Alice and (b) follows from secrecy against Charlie. We will now show that Condition 2.1, secrecy against Alice, and secrecy against Charlie imply that \( I(X; M_{31}) = 0 \). The other case follows similarly.

We need to show that \( p(m_{31}|x) = p(m_{31}|x') \) for all \( x, x' \in X \). Suppose there is a \( y \in Y \) with \( f(x, y) = f(x', y) \), i.e., the rows of the truth-table of \( f \) corresponding to \( x \) and \( x' \) have a common element \( f(x, y) = f(x', y) = z \), say. Then, by secrecy against Alice \( p(m_{31}, x, z) = p(x, z)p(m_{31}|x) \) and by secrecy against Charlie, \( p(m_{31}, x, z) = p(x, z)p(m_{31}|z) \). Hence, \( p(m_{31}|x) = p(m_{31}|z) = p(m_{31}|x') \).

Condition 2.1 implies that for every \( x, x' \in X \), there is a sequence \( x_0 = x, x_1, x_2, \ldots, x_L \) such that for every pair \( (x_{l-1}, x_l) \), \( l = 1, 2, \ldots, L \), there is a \( y_{l} \in Y \) satisfying \( f(x_{l-1}, y_l) = f(x_{l}, y_l) \). In other words, for every pair \( x_{l-1}, x_{l} \), the corresponding rows of the truth-table have a common element. Hence, \( p(m_{31}|x) = p(m_{31}|x_{1}) = p(m_{31}|x_{2}) = \ldots = p(m_{31}|x_{L}) \).

**Proof of Lemma 3.4:** This simply follows from

\[
I(X; Y; M_{12}) = I(X; M_{12}) + I(Y; M_{12}|X).
\]

The second term is 0 by security against Alice. For the first term \( I(X; M_{12}) \leq I(X; M_{12}, Y) = I(X; Y) + I(X; M_{12}|Y) = 0 \), since \( X \) is independent of \( Y \) and the second term is 0 by secrecy against Bob.
REFERENCES