SECURE MULTIPARTY COMPUTATION

JESSIE COVINGTON AND MEGAN GOLBEK

Advisor: Mike Rosulek
Oregon State University

Abstract. A function is private if there exists a protocol that is secure; that is, no party can learn any additional information about the other parties' inputs other than what follows from their own input and the function's output. Previous studies have investigated multiparty computation with broadcast communication, but broadcast communication does not fully capture all secure multiparty protocols. We investigate secure multiparty computation with point to point communication in order to capture these protocols not captured by broadcasting. We present a secure three party protocol that computes an ordered product on a non-abelian group. We also begin a characterization of functions that can be securely computed for four or more parties. We present a proof that shows if four or more parties want to compute an ordered group product, then that group must be abelian.

1. Introduction

Secure multiparty computation has many applications in data mining such as, for example, Yao's millionaires' problem [5]. Another application for secure multiparty computation is in medical research [4]. Consider the scenario where multiple hospitals wish to jointly share data obtained from their patients for medical research, without revealing any information about the patient other than the required data. Privacy policies can prevent revealing confidential patient information. Suppose that hospitals could learn the information required for their research without the need for pooling patient records. They would then learn only the output of the data mining algorithm. Hospitals having greater access to a larger amount of data would greatly benefit research.

The purpose of this study was to provide a characterization for secure multiparty computation with point to point communication. Although we do not provide a full characterization for all secure multiparty computation with point to point communication, we take a step towards a more complete characterization using an interesting result found for secure multiparty computation on a group. For the multiparty case, we consider a set of n parties who wish to compute some ordered product with each of their inputs x_i. The parties communicate via point to point communication using secure channels.

In section 2, we present a simplified proof for one direction a theorem proven by Beaver and Kushilevitz independently of one another. In section 3, we consider the multiparty case and address the properties of an operation on a group. We present a secure protocol for the

Date: 14 August 2015.

This work was done during the Summer 2015 REU program in Mathematics at Oregon State University supported by NSF grant DMS-1359173.

three party case computed on a non-abelian group, but find there is no such protocol that is secure for the four party case on a non-abelian group. We present a proof for the following lemma: if $n \geq 4$ parties want to compute some ordered product on a group $\mathbb{G}$, then $\mathbb{G}$ must be abelian. This result is interesting because three party computation was thought to have the same properties as other multiparty computation. However, we have found that this is not true when the ordered product is on a group $\mathbb{G}$. Near the end of section 3, we consider whether an ordered operation must be associative for $n \geq 4$ parties.

1.1. **Definitions.**

Real/Ideal Paradigm We can think of security in terms of the *real* world versus the *ideal* world.

Suppose n parties with private inputs $x_1, x_2, ...x_n$ wish to compute some function of their inputs, $y = f(x_1, x_2, ...x_n)$.

In the ideal world, there exists a trusted and incorruptible party, call it I. Each party sends its input to I who then computes the value y and reveals y to all parties.

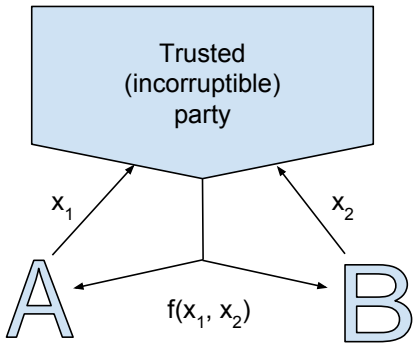


FIGURE 1. Ideal World: Two-party Example

In the real world, however, there does not exist a party who can be trusted and incorruptible. Instead, parties run a protocol amongst themselves, and by the end all parties learn the value of y.

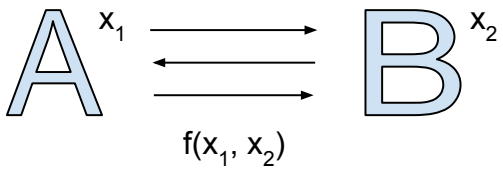


FIGURE 2. Real World: Two-party Example

The *view* of a party includes any messages they receive during the execution of the protocol as well as their own input, the output of the function, and any information they can derive based on these factors. For example, suppose we have a set of S parties with a set of possible inputs $\vec{x}$ carrying out protocol π. Then $view_S^{\pi}(\vec{x})$ represents the view of S on protocol π with inputs $\vec{x}$. Note that we use the symbol $\equiv$ to denote when two distributions are equivalent.

Definition 1.1. *A protocol is* secure *if for all* $S \subseteq [n] = \{1, \ldots, n\}$*, there exists a simulation* sim*, where* $\forall \vec{x} : view_s^{\pi} \vec{x} \equiv sim((x_{i \in S}),\ f(\vec{x}))$ *where* $\vec{x} = (x_1, x_2, \ldots, x_n)$*.*

In other words, a protocol is secure if for all actions in the real world, there is a way to simulate that action in the ideal world with the same results. We use the ideal world as a way to justify that a protocol is secure in the real world. In order to show that a protocol is secure, we need to show that all of the probability distributions in the real world are equivalent to all probability distributions in the ideal world.

Below is an example of how we can show a given protocol is secure:

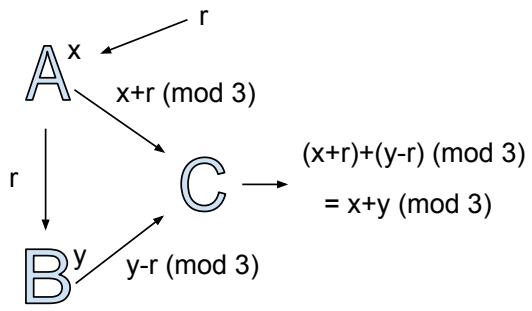
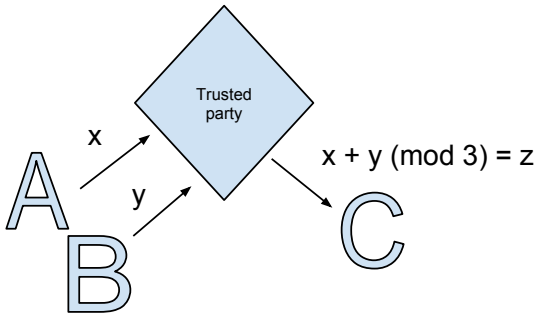


FIGURE 3. Real World Protocol

x	y	view distribution	x+y (mod 3)
0	0	⅓ (0, 0) + ⅓ (1, 2) + ⅓ (2, 1)	0
0	1	⅓ (0, 1) + ⅓ (1, 0) + ⅓ (2, 2)	1
0	2	⅓ (0, 2) + ⅓ (2, 0) + ⅓ (1, 1)	2
1	0	⅓ (0, 1) + ⅓ (1, 0) + ⅓ (2, 2)	1
1	1	⅓ (0, 2) + ⅓ (2, 0) + ⅓ (1, 1)	2
1	2	⅓ (0, 0) + ⅓ (1, 2) + ⅓ (2, 1)	0
2	0	⅓ (0, 2) + ⅓ (2, 0) + ⅓ (1, 1)	2
2	1	⅓ (0, 0) + ⅓ (1, 2) + ⅓ (2, 1)	0
2	2	⅓ (0, 1) + ⅓ (1, 0) + ⅓ (2, 2)	1

FIGURE 4. Real World Protocol View Distribution



z	r' = 0 r' = 1 r' = 2
0	⅓ (0, 0) + ⅓ (1, 2) + ⅓ (2, 1)
1	⅓ (0, 1) + ⅓ (1, 0) + ⅓ (2, 2)
2	⅓ (0, 2) + ⅓ (1, 1) + ⅓ (2, 0)

FIGURE 5. Ideal World Protocol Simulation (left) and View Distribution (right)

In the ideal world simulation, r' represents the simulated random variable r from the real world protocol.

As you can see from the two figures above, if $z = x + y(mod3)$ then the probability distributions are equivalent. This is sufficient in showing that the protocol in the real world is secure.

Other ideas used through the paper include:

A *semi-honest* adversary, also called *honest-but-curious*, follows the protocol, but may use the messages they receive during the execution of the protocol to attempt to learn more information that to which they are entitled. The [4]

From here on, we will be dealing with the case of the semi-honest adversary.

Point-to-point communication (as opposed to broadcasting - where every party learns the same information) is made up of authenticated channels so that individual parties can exchange certain information without other parties also receiving the messages [4].

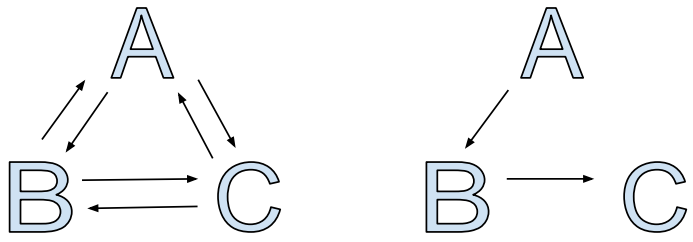


FIGURE 6. Broadcast (left) versus Point-to-point (right) communication

2. Two Party

The set of secure two-party functions has been characterized by Beaver and Kushilevitz, independently of one another. They proved that the set of finite functions that can be computed privately in an information-theoretic sense by two parties is the set of decomposable

functions [1]. In other words, there exists a secure protocol to compute a function if and only if that function is decomposable [3].

Definition 2.1. *A function, f, is* decomposable *if*

(1) *f is constant, or*
(2) *f is partitionable, meaning either:*
   - *$\exists P, Q$ with $P \cup Q = X$; $\forall y \in Y, x_1 \in P, x_2 \in Q$; $f(x_1, y) \neq f(x_2, y)$ and $f : P \times Y, f : Q \times Y$ are also decomposable*
   - *$\exists P, Q$ with $P \cup Q = Y$; $\forall x \in X, y_1 \in P, y_2 \in Q$; $f(x, y_1) \neq f(x, y_2)$ and $f : X \times P, f : X \times Q$ are also decomposable*

Example 2.2. *Below is an example of a decomposable and non-decomposable function.*

1	2	2	2
1	3	4	4
1	3	5	6
1	3	5	7

1	1	2
4	5	2
4	3	3

FIGURE 7. decomposable (left) versus non-decomposable (right) function

Building off of this proof, we present a simplified proof for one direction of the following theorem:

Theorem 2.3. *A secure protocol exists for f if and only if f is decomposable.*

Proof. If a protocol π is secure, then f is decomposable.

Two parties, P_1, P_2 with inputs $x \in X$ and $y \in Y$ respectively, want to compute some function $f : X \times Y \to \mathbb{G}$. Let $n(\tau)$ be the expected number of rounds for some secure protocol τ, when x and y are sampled uniformly. Let n^* be the infimum over all the secure protocols computing f.

There may not be protocol where the expected number of rounds is equal to n^* because it is defined as the greatest lower bound over all secure protocols. This means that we may never reach that lower bound in terms of the expected number of rounds, but we know that there exists a protocol δ where $n(\delta)$ is within the n^* and $n^* + 1$. Let us pick a secure protocol π such that $n(\pi) < n^* + 1$.

We will prove by contradiction.

Assume f is non-decomposable. Without loss of generality, assume that P_1 sends the first message. Let us denote this first message by m_1. Let D_x be the probability distribution on m_1. Since m_1 depends only on x, D_x is well defined.

We have two cases.

Case 1: $\forall x, x' \in X\ D_x \equiv D_{x'}$

Since the probability distributions are the same for all $x \in X$, then m_1 is input independent and does not rely on input by P_1. Thus, P_2 can sample from this distribution and generate m_1. Define a new protocol π' such that P_2 sends m_1 along with a response to m_1, then π' proceeds as π does.

Since the probability distributions for π and π' are equivalent because all the same messages are still sent and π is secure, π' must also be secure.

We eliminated the first round of protocol π, thus we have $n(\pi') = n(\pi) - 1$, but:
$n(\pi) < n^* + 1$
$n(\pi) - 1 < n^*$
$n(\pi') < n^*$

This is a contradiction.

Case 2: $\exists x, x'\ D_x \not\equiv D_{x'}$

This means that D_x does depend on P_1's set of inputs.

For some arbitrary $x_0 \in X$, we define a partition with $P \cup Q = X$.
$P := \{x \mid D_x \equiv D_{x_0}\}$
$Q := X \setminus P$

Note that P, Q are both nonempty sets by definition of the sets and in Case 1 we have shown that if all x are contained in P, we get a contradiction.

Since f is non-decomposable, then $\exists y \in Y, x_i \in P, x_j \in Q$ such that $f(x_i, y) = f(xj, y)$. Then, by the definition of security, $view^{\pi}_{P_2}(x_i, y) = view^{\pi}_{P_2}(x_j, y)$. Therefore, since the views are the same, then the distributions are also equivalent and $D_{x_i} \equiv D_{x_j}$ by definition of security. But this is a contradiction since $x_i \in P$ and $x_j \in Q$ and P contains all inputs with the same message distribution.

Since both cases lead to a contradiction, f must be decomposable.

□

3. Protocols for multiparty group product

Characterizations of privately computable functions have been found for the multiparty case, but these often encapsulates broadcast but not point to point communication. Künzler, Müller-Quade, and Raub provided such a characterization [2], but it does not fully capture all possibilities. No secure three-party protocol exists for broadcast communication, but we find a secure three-party protocol for point-to-point communication, as we will present later in this section.

For the two-party case, functions take the form $f(x_1, x_2) = f_1(x_1) \oplus f_2(x_2)$ on the group $\mathbb{Z}_2$. We want to generalize the two party computation on this group to multiparty computation on a group, $\mathbb{G}$. Our results show that in the three party case, $\mathbb{G}$ can be non-abelian, but in the four party case $\mathbb{G}$ must be abelian. This result shows that the three party case and the four party case have different properties. In order to show that the three party case, where the ordered product is computed on a non-abelian group, we need to present a secure protocol for this case.

3.1. **Three Party Protocol.**
Let there be three parties: A, B, C with inputs: a, b, c respectively want to compute the value of abc.

We present a private three party protocol for computing some ordered group product $f(a, b, c) = abc$, where the group is non-abelian.

(1) A chooses x uniformly in G.
(2) A computes the product xa and sends this value to B.
(3) B then computes this value with its input b and sends the new product xab to C
(4) C computes $xabc$ and sends this product to A.
(5) Since A knows the value of x, A also knows x^{-1}. A computes $x^{-1}(xabc)$. Since G is a group, $\mathbb{G}$ is associative, therefore: $x^{-1}(xabc) = (x^{-1}x)(abc) = abc$. Finally, A sends the value abc to B and C

Below is a figure of the secure three party protocol.

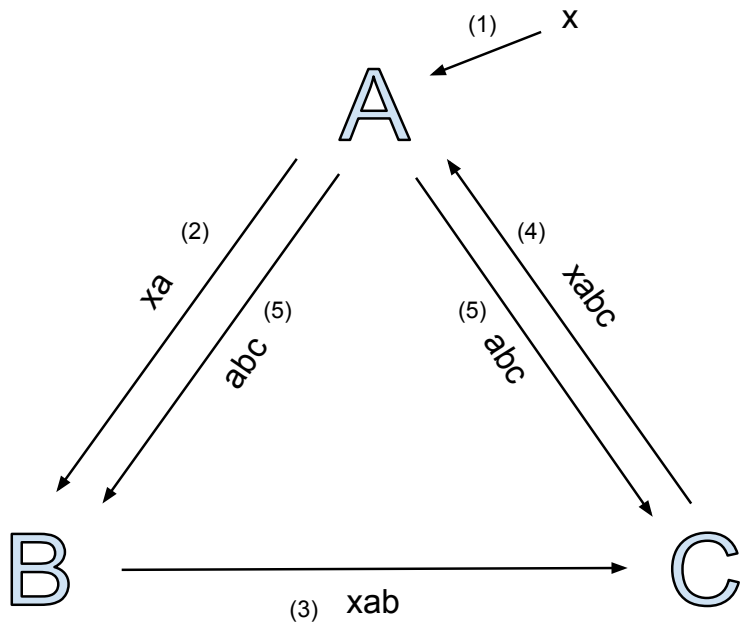


FIGURE 8. Secure Three Party Protocol with Point to Point Communication

The above protocol is secure, as we will show below.

In the ideal world, each party knows its own input, and each party learns the output of the function $f(a, b, c) = abc$. We can create a simulator *sim* that accepts the party's initial input and abc as input. *sim* then outputs a view distribution based on these parameters.

Suppose A is corrupt. A holds input a and the output abc. If A runs the simulation in the ideal world, they can also sample a random x', to simulate the randomization of the protocol. So, $sim_A(a, abc) = (a, abc, bc)$.

In the real world, we can see A's view is: $view_A(a) = (a, abc, bc)$ based on the messages that A receives during the protocol. So $sim_A(a, abc) \equiv view_A(a)$.

Suppose B is corrupt. B holds input b and the output abc. If B runs the simulation in the ideal world, they can also sample a random x', to simulate the randomization of the protocol. So, $sim_B(b, abc) = (b, abc)$.

In the real world, we can see B's view is: $view_B(b) = (b, abc)$, as well as incoming randomized message xa (which can be simulated in the ideal world with a random x') based on the messages that B receives during the protocol. So $sim_B(b, abc) \equiv view_B(b)$.

Suppose C is corrupt. C holds input c and the output abc. If C runs the simulation in the ideal world, they can also sample a random x', to simulate the randomization of the protocol. So, $sim_C(c, abc) = (c, abc, ab)$.

In the real world, C's view is: $view_C(c) = (c, abc, ab)$ based on the messages that C receives during the protocol. So $sim_C(c, abc) \equiv view_C(c)$.

Since the view of each party is equivalent in both the ideal and the real worlds, then the protocol is secure.

3.2. **Multiparty.**

Theorem 3.1. *Suppose n number of parties want to securely compute some ordered product on a group $\mathbb{G}$. If $n \geq 4$, then $\mathbb{G}$ must be abelian.*

We can restrict the four-party case by projecting it to the two-party case; we divide the parties into two clusters (as seen in a figure below). If there exists a secure protocol for four parties, it can be restricted to the two-party case, and then there exists a secure protocol for the two party restriction.

We will prove by contradiction.

Proof. Suppose we have four parties: P_1, P_2, P_3, P_4 each holding the sets of possible inputs A, B, C, D respectively, where A, B, C, D are non-abelian groups.

Consider the two-party restriction where P_1, P_3 are in a cluster together with the collective set of inputs $X = A \times C$, and P_2, P_4 are in a cluster together with the collective set of inputs $Y = B \times D$.

For $a \in A, b \in B, c \in C, d \in D$, we define $f : X \times Y \to \mathbb{G}$ on some non-abelian group $\mathbb{G}$ (with $(a, c) \in X, (b, d) \in Y$) such that $f((a, c), (b, d)) = abcd$.

Since $\mathbb{G}$ is non-abelian, choose two arbitrary group elements x, y such that $xy \neq yx$.

Inductive Hypothesis: If $X, Y \subseteq \mathbb{G}^2$ have the property that $(\forall a \in A, \exists c \in C : (a, c) \in X)$ and $(\forall b \in B, \exists d \in D : (b, d) \in Y)$ and f is of the form $f((a, c), (b, d)) = abcd$, then f is not decomposable.

Base Case: Assume f is decomposable, this means $\exists P, Q$ where $P \cup Q = X$; $\forall (b, d) \in Y, (a, c) \in P, (a', c') \in Q$; $f((a, c), (b, d)) \neq f((a', c'), (b, d))$. Then $abcd \neq a'bc'd$. Let $(b, d) = (1, d)$. Then $a(1)cd \neq a'(1)c'd$, so $ac \neq a'c'$.

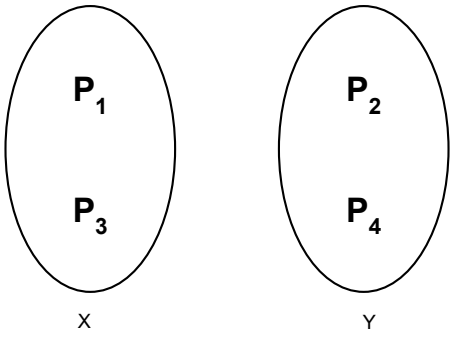


FIGURE 9. Two-Party Restriction

Then we can define partitions P and Q with $S_P, S_Q \subseteq \mathbb{G}$ such that $P := \{(a, c) \mid ac \in S_P\}$ and $Q := \{(a', c') \mid a'c' \in S_Q\}$

Inductive Step: Let $(x, y), (y, x) \in X \cup Y$. We have two cases.

Case 1: $(x, y) \in P$ and $(y, x) \in Q$

Since f is decomposable, $xbyd \neq ybxd$ and so $xby \neq ybx$. Choose a (b, d) such that $b = x^{-1}$. Then we get $y \neq y$, which is a contradiction.

Case 2: $(x, y), (y, x) \in P$

Since $X \subseteq G^2$ and $P \subseteq X$, $P \subseteq G^2$. Since $P \subseteq X$ and P is the nontrivial group by definition of P, P has the property $\forall a \in A \setminus Q, \exists c \in C \setminus Q : (a, c) \in P$. We know that the group Y has the same property after partitioning f to $f : P \times Y \to \mathbb{G}$. Since $X \subseteq G^2$ and $P \subseteq X$, $P \subseteq G^2$. Since partitioning a function does not alter the function's output, the output of f still satisfies the properties of the inductive hypothesis. Therefore f satisfies the Inductive Hypothesis.

Therefore, f is not decomposable when $\mathbb{G}$ is non-abelian.

□

3.3. **Associative Property.**

Our results for the multiparty computation assume the protocol is on a group, but we know that in the two-party case, weaker structures will suffice - for example, a quasi-group. We believe that it can be shown that associativity is required when the number of participants n is $n \geq 4$. This proof is a work in progress:

Lemma 3.2. *Suppose n number of parties want to securely compute some ordered product on a set $\mathbb{S}$. If $n \geq 4$, then the operation on $\mathbb{S}$ must be associative.*

We will prove by contradiction.

Proof. Suppose there exists a secure protocol for four parties P_1, P_2, P_3, P_4 to securely compute an ordered product on the non-associative set $\mathbb{S}$. Let P_1, P_2, P_3, P_4 hold the sets of

inputs $A, B, C, D \subseteq \mathbb{S}$ respectively. Consider the following two party restriction: P_1, P_2 in a cluster with the input $A \times B = X$ and P_3, P_4 in another cluster with the input $C \times D = Y$.

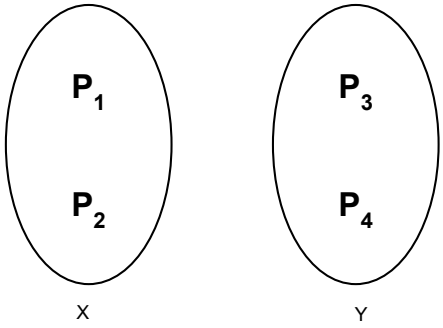


FIGURE 10. Two-Party Restriction

For $a \in A, b \in B, c \in C, d \in D$, define $f : X \times Y \to \mathbb{S}$ as $f(a, b, c, d) = (a(bc))d$.

Since $\mathbb{S}$ is not associative, choose $x, y, z, z' \in \mathbb{S}$ such that $x(yz) = (xy)z'$.

There exists a secure protocol to compute f for this two-party restriction, therefore f must be decomposable. So $\exists P, Q$ such that $\forall (a, b) \in X, (c, d) \in P, (c', d') \in Q$ $(a(bc))d \neq (a(bc'))d'$

Choose $\forall (a, b) = (x, y)$, $(c, d) = (z, 1)$ and $(c', d') = (1, z')$. Then, $(x(yz))1 \neq (x(y1))z'$ but this yields $x(yz) \neq (xy)z'$ which contradicts our assumption.

Note that we have three other cases: $f(a, b, c, d) = a(b(cd))$ and $f(a, b, c, d) = ((ab)c)d$. We believe that all three cases can be shown to fulfill Lemma 3.2. The two other cases remain unproven.

□

4. Conclusion

We have presented a simpler proof for one direction of Theorem 2.3 that was previously proven by Beaver and Kushilevitz. We have also shown that for the three party case, there exists a secure protocol on a non-abelian group. This protocol combined with the results of Theorem 3.1 uncovers an interesting result: though for four or more parties, an abelian group is necessary for the existence of a secure protocol, in the three-party case, this does not hold true. As such, the characterization for the three-party case is different than the characterization needed for four or more parties, contrary to what we believed as we began our research. We conjecture that if $n \geq 4$ parties want to compute some ordered product on a set, then the operation on that set must be associative in order for a secure protocol to exist.

The purpose of this study was to provide a characterization for secure multiparty computation with point to point communication. Although we do not provide a full characterization for secure multiparty computation with point to point communication, we provide the first step in this characterization.

This research helps provide more information about the ordered operations required for a secure protocol to exist in the semi-honest setting. There are many applications for secure multiparty computation with semi-honest parties, from Yao's millionaires' problem [5] to research in the medical field.

References

[1] Donald Beaver. Perfect privacy for two-party protocols. In *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 65–77, 1991.

[2] Robin Künzler, Jörn Müller-Quade, and Dominik Raub. Secure computability of functions in the it setting with dishonest majority and applications to long-term security. In *Theory of Cryptography*, pages 238–255. Springer, 2009.

[3] Eyal Kushelvitz. Privacy and communication complexity. *SIAM Journal on Discrete Mathematics*, 5(2):273–284, 1992.

[4] Yehuda Lindell. Secure multiparty computation for privacy preserving data mining. *Journal of Privacy and Confidentiality*, 1(1):5, 2009.

[5] Andrew Chi-Chih Yao. Protocols for secure computations. In *FOCS*, volume 82, pages 160–164, 1982.

College of Charleston
E-mail address: covingtonjg@g.cofc.edu

California State University, Monterey Bay
E-mail address: mgolbek@csumb.edu