PSI-Stats: Private Set Intersection Protocols Supporting Secure Statistical Functions

Jason H. M. Ying[1,2], Shuwei Cao[2,3], Geong Sen Poh[2,3], Jia Xu[2,3], and Hoon Wei Lim[2,3]

[1] National University of Singapore
dcsyhmj@nus.edu.sg
[2] NUS-Singtel Cyber Security Lab
[3] Trustwave
shuwei.cao@trustwave.com
geongsen.poh@trustwave.com
jia.xu@trustwave.com
hoonwei.lim@trustwave.com

Abstract. Private Set Intersection (PSI) enables two parties, each holding a private set to securely compute their intersection without revealing other information. This paper considers settings of secure statistical computations over PSI, where both parties hold sets containing identifiers with one of the parties having an additional positive integer value associated with each of the identifiers in her set. The main objective is to securely compute some desired statistics of the associated values for which its corresponding identifiers occur in the intersection of the two sets. This is achieved without revealing the identifiers of the set intersection. This has many useful business applications, for examples in measuring effectiveness of advertising campaigns.

In many cases, the parties wish to know various statistical information with regards to the set intersection and the associated integer values. For instance, information relating to arithmetic mean, geometric mean, harmonic mean, standard deviation, minimum, maximum, range or an approximate distribution of the sum composition. A potential use case is for a credit card company to provide the percentage of high spending to a shopping mall based on their common customers. Therefore, in this paper we introduce various mechanisms to enable secure computation of statistical functions, which we collectively termed *PSI-Stats*. The proposed protocols maintain strong privacy guarantee, that is computations are performed without revealing the identifiers of the set intersection to both parties. Implementations of our constructions are also carried out based on a simulated dataset as well as on actual datasets in the business use cases that we defined, in order to demonstrate practicality of our solution. *PSI-Stats* has a lower communication overhead compared to the current state-of-the-art circuit-based PSI protocol of Pinkas *et al.* (EUROCRYPT'19) by a factor of at least 3.9× and consequently has a lower run time than the latter at low network bandwidth settings from our experiments. Our solution is more tailored towards business applications where communication cost is the primary consideration.

1 Introduction

Private set intersection (PSI) enables two parties to learn the intersection of their sets without exposing other elements (identifiers or items) that are not within this intersection. This has wide-ranging applications in data sharing, private contact discovery, private proximity testing [43], privacy-preserving ride-sharing [32], botnet detection [42], human genomes testing [12] and more recently for contact tracing [57] in the event of a pandemic such as COVID-19. We highlight a number of notable work that have been achieved in this domain in Section 2.

The main problem statement of our work can be simply described as follows. Two parties A and B hold sets of identifiers with party B additionally holds positive integer values associated with each of the identifiers. Denote the sets held by A, B to be X and Y respectively. The objective is for B to learn the desired statistical output function of some collection (dependent on X) of the associated values, while preserving certain private information about their respective sets. More formally, B seeks to learn the value $F_D(X, Y)$, where D is the decisional rule and F is the desired statistical function computed over D. To preserve privacy, A does not learn Y and $D(X, Y)$ while B does not learn X, $D(X, Y)$ and $|D(X, Y)|$. In our context, D is the private set intersection (PSI) of the identifiers contained in X and Y. In a *reverse protocol*, A learns $F_D(X, Y)$ instead of B. These settings arise in numerous business and practical applications.

1.1 Our Contributions

We present *PSI-Stats* to address this main problem statement. *PSI-Stats* is a collection of protocols to support the secure computations of statistical functions over PSI. These include a myriad of frequently applied standard statistical functions such as various generalized means, standard deviation, variance, percentile sum, etc. The proposed protocols achieve all the privacy requirements outlined in the problem statement. The main contributions are summarized here.

- We introduce new approaches to securely compute various standard statistical functions while maintaining the privacy guarantees as defined in the main problem statement. The functions include arithmetic mean, geometric mean, harmonic mean, min, max, percentile sum, standard deviation, variance, etc. Our techniques are also applicable to non-symmetric functions such as weighted arithmetic mean.
- *PSI-Stats* can be enabled to securely compute multiple related statistical functions (e.g. percentile sum, range and arithmetic mean) within a single executed protocol with minimal additional communication and computational overhead.
- We provide formal security analysis of our protocols which are based upon well-established hardness assumptions as well as standard security definitions. The reliance of such hardness assumptions (e.g. integer factorization) have been well studied by the cryptographic as well as the theoretical computer science communities and are generally regarded to be valid. Our protocols

achieve strong security guarantees from statistical indistinguishability. Solutions based on well-established assumptions carry an additional subtle importance that is often overlooked. This basis is generally a strong factor in the driving force for adoption by industry stakeholders.

- Ion *et al.* (Google 2019) [35] recently deployed a scheme to solve the main problem statement in the sole special case where F denotes the sum. It should be noted that any natural attempts to convert the computation of sum to arithmetic mean by sending the set intersection size to B violates the privacy requirements of the problem statement. In contrast, our work here provides solutions to a large class of statistical functions F. By doing so our protocols provide more flexible and comprehensive utilities.
- A *reverse protocol* is presented in [35] for the functionality of intersection sum. However, that protocol leaks size information of the set intersection to both parties. *PSI-Stats* can be readily adapted to a *reverse protocol* without revealing this size information to B, while still enabling support for the outputs of various statistical functionalities.
- We carried out experiments of our protocols to determine their practicality and feasibility. Our test input sizes range from small to large. The results obtained are comparable to the recently deployed private intersection-sum protocol of [35] both with regards to running time as well as communication cost. The experimental results demonstrate that *PSI-Stats* is practical and scales well for large input sizes. We also conducted experimental comparisons of our protocols with the current state-of-the-art circuit based PSI protocols.

In an interactive protocol, there are three factors in the overall measurement of efficiency: the first relates to the communication overhead, the second relates to the computational cost and the third relates to the number of communication rounds (or round complexity). The work in this paper does not claim to outperform circuit-based PSI protocols across all the three factors above. As an example, the current state-of-the-art for circuit-based PSI protocols is the very recent work of Pinkas *et al.* [47] which we reckon to potentially attain the lowest computational cost (after the necessary circuit modifications in order to accommodate outputs of statistical functions).

A goal of our work aims to present protocols with minimal communication overhead based upon well-established, time-tested hardness assumptions while concurrently ensuring that running times remain practical. To that end, the *PSI-Stats* protocols in this paper incur the lowest communication overhead over all circuit-based types (inclusive of the most recent state-of-the-art [47]) by several factors. In that regard, *PSI-Stats* is especially relevant in settings where communication cost comes at a premium or in instances where bandwidth is limited.

Circuit-based PSI approaches can generally be instantiated by either Yao's garbled circuit protocol [59] or the GMW protocol [31]. While Yao's protocol provides a constant round complexity, the GMW protocol is typically the overall preferred option as it has several advantages over the former. A comprehensive comparison between Yao's protocol and the GMW protocol can be found in [55].

However for circuit-based PSI under the GMW family, the round complexity is dependent on the circuit depth which increases with increasing set sizes and/or increasing bit-length of items. This can potentially be a bottleneck in high latency networks. By contrast, the *PSI-Stats* protocols operate with a low constant round complexity of 3 (and 4 in the reverse), independent of the input set sizes *and* the bit-length of items.

1.2 Use Cases

Our motivation lies in that different organisations are increasingly geared toward sharing data among them in order to derive new insights and create new business opportunities from the shared data. Here, we present two use cases that can benefit from learning more than just the sum of the intersected set. For example, to find min, max or threshold of values from the set. Certainly, a simple approach to obtain the above result is to use existing state-of-the-art PSI protocol to find the set intersection and then B computes the statistical functions on the values associated to the identifiers in the intersection. Here, the main goal is to also hide the identifiers in the set intersection from being known to both parties. Hiding the elements in the set intersection can be useful to alleviate privacy concerns from the participating parties especially when an organisation would like to prevent the other party from learning some of their existing customers based on the set intersection. We first look at an extension of the use case of the Private Intersection-Sum protocol [35] between a merchant and an advertising entities. By learning the min, max, or the percentage of small spending (or large spending) after users viewed an advertisement, the merchant may use this statistical information for targeted advertisement to advertise products within the range of spending of the users. Alternatively, the merchant may devise new reward scheme to encourage users to broaden their spending habits.

A second use case is data sharing between a mall and a credit card company with similar intent as the above use case. The proposed solution enables the credit card company to match its customer to the mall's customer, without revealing the underlying individuals (not even the matched customers). This prevents the credit card company (or the mall) from learning more information such as which customers are also the customers of the mall (or credit card company), which can be a privacy concern to both participating parties. The credit card company then reveals the min (or max) spending, or percentage of high spending to the mall.The mall may market these insights to existing tenants or attract new relevant tenants.

2 Related work

2.1 Existing PSI Protocols

An early PSI protocol is based upon the Diffie Hellman paradigm [40] which is also applicable in elliptic curve cryptography. A similar idea can be traced back

to [56] and also in [34]. That is based on the multiplicative property related to the Diffie-Hellman key exchange. A method based on oblivious polynomial evaluation was introduced in [28, 29]. An approach via blind RSA was presented in [19]. All the above methods are based on public-key cryptography.

Oblivious transfer (OT) extension was first introduced in [36]. The main objective of an OT extension is to enable the computation a large number of OT based off a smaller number along with symmetric cryptographic operations to achieve better running times. This technique engendered numerous OT-based PSI protocols. The notion of garbled bloom filter based on OT extension was coined in [22] and utilized to perform PSI. The main idea is to allow one of the parties to learn the bit-wise AND of two Bloom filters via OT. This outcome results in a valid Bloom filter for the set intersection. That was subsequently optimized to some extent in [49]. There are a number of other notable OT-based schemes presented in [18, 39, 44, 46, 49, 50]. The particular work of [39] is based on an OT extension protocol found in [38].

Generic multi-party protocols such as garbled circuits can also be used to compute PSI. The first such protocol involving garbled circuit appeared in [33] which was later improved in [49]. Other notable circuit-based PSI protocols are presented in [26, 46–48]. Circuit-based approaches can typically serve for generic computation purposes. On the other hand, they result in larger communication overheads as compared to other custom based PSI protocols.

Unbalanced PSI refers to the setting where one party has a much smaller set size compared to the other in the two-party scenario and the bandwidth between them is typically limited. As such, since *PSI-Stats* protocols incur minimal communication overhead, they are also applicable in the context of unbalanced PSI especially in low or limited bandwidth settings. A widespread application arising from unbalanced PSI is that of private contact discovery. A user who holds a small set of contacts wishes to learn which of these contacts also use a particular service by a service provider. There are a few protocols which seek to optimize existing generic PSI protocols to handle such settings [17, 48, 51]. The current state-of-the-art of an unbalanced PSI protocol based upon fully homomorphic encryption is presented in [16] which is an improved follow-up of [17]. A recent finding in relation to private contact discovery for mobile systems is given in [37]. The work of [51] based upon techniques of [12] attempts to reduce the communication overhead of unbalanced PSI via a compression method but in the process introduces a non-negligible false positive rate.

2.2 Private Intersection-Sum Protocol

The most related existing work in relation to this paper is that of Ion *et al.* [35]. In a nutshell, the core of the protocol comprises of two main components. The first is a PSI protocol that enables element hiding while the second is a homomorphic encryption which has the additive property. By adjoining these two components, the functionality of the intersection sum can be obtained. In that work [35], several probable methods are proposed. They vary in accordance to the types of PSI and additive homomorphic encryption schemes. The candidates for

the various PSI schemes utilized are based upon Diffie-Hellman [34, 40], Random Oblivious Transfer [46, 49, 52] as well as Bloom Filter [20, 22, 24]. The candidates for the additive homomorphic encryption schemes are based upon Paillier [45], Exponential ElGamal [25] and Ring-LWE [14, 15, 27, 30].

The eventual choice of deployment is the Diffie-Hellman based protocol due to its communication efficiency. The Paillier encryption was chosen as the homomorphic encryption scheme due to the overall consideration of two factors: its underlying well established classical hardness assumption as well as its computational cost. As such, the Diffie-Hellman based paradigm augmented with Paillier encryption will also form the basis in the design of our protocols.

2.3 Commercial Solutions

Secure multiparty computing (MPC), while has been known to academia for a few decades, is now starting to become commercially viable. With the increasing demands of privacy protection in businesses, security and risk management leaders have placed emphasis on not only data at rest and in transit, but also data in use (e.g. while it is being processed by an application). It has recently been reported by Gartner [58] that MPC-based technology has emerged in the market to help businesses ensure data security and privacy in a more effective and efficient manner. Gartner has seen a number of vendors, such as Unbound Tech [10], Sharemind [9], Sepio [8], and Baffle [3], that leverage MPC to provide clients with data and cloud security. These solutions typically focus on supporting standard SQL queries over encrypted data or threshold-based key management. Moreover, there exist private data linkage solutions, such as Anonos [2], Data Republic [4] (which make use of conventional hash-based de-identification techniques), and Privitar [7] (which is based on more modern PSI techniques).

Despite the commercial potential of MPC solutions, there are still privacy concerns in data sharing applications particularly in terms of how much information can be inferred by consumers of shared data. All the above-mentioned commercial solutions do not protect the privacy of elements within the intersection of two or more datasets. The elements in the intersection, when combined with auxiliary information, may lead to revelation of information that can be used to re-identify specific individuals. In our work, we focus on a PSI-based solution that addresses such privacy concerns.

3 Preliminaries

The security model in this paper operates in the semi-honest setting. In this model, adversaries can attempt to obtain information from the execution of the protocol but they are unable to perform any deviations from the intended protocol steps. The semi-honest model is typically suited in scenarios where execution of the software is ensured through software attestation or business restrictions, without any assumption that an external untrusted party is unable to obtain the transcript of the protocol upon completion. Indeed, the majority of

the research in related domains also focus on solutions in the semi-honest model. A formal description of the semi-honest security model and the simulation based security proof of *PSI-Stats* is provided in the Appendix.

In this paper, we say an integer x is l-bit (length) if $x \in \mathbb{Z} \cap [2^{l-1}, 2^l - 1]$ and x is at most l-bit (length) if $x \in \mathbb{Z} \cap [0, 2^l - 1]$. The standard ceiling function is given by $\lceil . \rceil$, where $\lceil x \rceil$ represents the smallest integer greater than or equal to x. The nearest integer function is denoted by $\lfloor . \rceil$, log refers to the natural logarithm, e is the standard mathematical constant (i.e. the base of the natural logarithm) and lcm is a shorthand for the lowest common multiple. The second Chebyshev function $\psi(x) = \sum_{p^k \leq x} \log p$ computes the sum over all prime powers not exceeding x.

For the remainder of this section, we review a number of technicalities which are applicable in the construction of our protocols. We state the types of statistical functions which are supported by our protocols. The security assumptions of the protocols are discussed and the settings of the participants are also provided.

3.1 Definitions & Building Blocks

The Decisional Diffie-Hellman assumption

Definition 1. *Let G be an abelian group of prime order p and g a generator of G. The Decisional Diffie-Hellman assumption states that for random $a, b, c \in \mathbb{Z} \cap [1, p]$, the distributions (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^c) are computationally indistinguishable.*

The Decisional Composite Residuosity Problem

Definition 2. *The decisional composite residuosity problem is hard if for all polynomial-sized circuits $\mathcal{C} = \{\mathcal{C}_k\}$ there exists a negligible function negl such that*

$$|Pr(\mathcal{A}(\mathcal{A}(N, x) = 1 | x = y^N (N+1)^r \ mod\, N^2)$$
$$- Pr(\mathcal{C}(N, x) = 1 | = y^N \ mod\, N^2)| \leq negl(k)$$

where N is a random RSA composite of length k-bit, r is selected randomly in $\mathbb{Z}_N$ and probabilities are taken over choices of r, y and N.

Partial Homomorphic Encryption. Simply stated, a probabilistic public key encryption E is additively homomorphic if $E(m_1 + m_2)$ can be efficiently computable given $E(m_1)$ and $E(m_2)$ for messages m_1, m_2 without knowledge of the private key.

A crucial building block of our protocols is a semantically secure additively homomorphic encryption scheme. Any such general encryption scheme satisfying these properties is applicable in our protocols. In our implementations, we choose the Paillier [45] encryption scheme described below for this purpose.

Paillier Encrytion Scheme. Let the public key N be a RSA composite; the

private key be $\phi(N)$ where ϕ denotes the Euler's totient function and $m \in \mathbb{Z}_N$ be a valid message. The encryption is given by

$$E(m, r) = r^N (N+1)^m \bmod N^2 \tag{1}$$

where $r \in \mathbb{Z}_N^*$ is chosen at random. Given a ciphertext $c \in \mathbb{Z}_{N^2}^*$, its decryption is given by

$$D(c) = \frac{\phi(N)^{-1}}{N} \left[\left(c^{\phi(N)} \bmod N^2 \right) - 1 \right] \bmod N. \tag{2}$$

The Pallier encryption scheme is known to be semantically secure, assuming the hardness of the decisional composite residuosity problem.

3.2 Statistical Functions

The main goals of this paper are to perform secure computations of various statistical functions over two party private set intersection. The statistical functions that we consider in this paper are those that are commonly applied on databases such as arithmetic mean, geometric mean, harmonic mean, variance, standard deviation, percentile, etc. Hereinafter, we shall simply refer to mean as being arithmetic mean while references to other generalized means will be stated explicitly. In addition, we also build upon our method of computing intersection mean to compute other statistical quantities like skewness and kurtosis.

3.3 Security Assumptions

We refer to implicit information as the information for which the party who knows the intersection sum (along with her private set of identifiers and associated values) can deduce certain aspects of the intersection content. For example, if the output sum is an odd integer and there is only one associated odd value in the set, then this information implicitly reveals that the identifier corresponding to the odd associated value has to be in the intersection. The output sum also reveals that no identifier with associated value greater than the output sum can be in the intersection. Indeed, any generic black box construction which reveals the output sum incurs this implicit information loss. In view of this, the security assumptions in this work (as with [35]) do not take into account such implicit information loss or inference attack from the output functionality. Nevertheless, the security and privacy considerations are taken into account for all communications prior to the final output functionality. The security of all the protocols described in this paper is based upon the validity of the Decisional Diffie-Hellman assumption and the semantic security of the Paillier encryption scheme which assumes the hardness of the Decisional Composite Residuosity problem.

3.4 Participants' Setting

The participants' setting for all our protocols in this paper is identical. We provide it here to serve as a convenient common reference.

Notations
a_i, b_i: identifiers.
A holds $X = \{a_1, a_2, \ldots, a_m\}$.
B holds $Y = \{(b_1, t_1), (b_2, t_2), \ldots, (b_n, t_n)\}$, $t_i \in \mathbb{Z}^+$.
$Y' = \{b_1, b_2, \ldots, b_n\}$.
$E(.)$: Paillier encryption.
$h(.)$: SHA-256 hash function.
G: a group of large prime order.

4 Private Set intersection-mean

This section describes a protocol to output only the intersection mean (i.e. without disclosing intersection-sum nor intersection cardinality to B). Our protocol outputs the intersection mean which can be made arbitrary close to the exact value.

PROTOCOL 1 (Private Set Intersection-Mean)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$, B outputs intersection mean.

1. **Setup:** A and B jointly agree on E, a hash function h and a group G of large prime order. B generates a public-private key pair of E, announces the public key and keeps the private key to herself.
2. A**'s encryption phase:** A
(a) selects a random private exponent $k_1 \in G$;
(b) computes $h(a_i)^{k_1}$.
A sends $h(a_i)^{k_1}$ to B.
3. B**'s encryption phase:** B
(a) selects a random private exponent $k_2 \in G$;
(b) computes $h(a_i)^{k_1 k_2}$;
(c) computes $\{h(b_j)^{k_2}, E(t_j)\}$.
B returns $h(a_i)^{k_1 k_2}$ in shuffled order to A. B sends $\{(h(b_j)^{k_2}, E(t_j)\}$ to A.
4. **Matching & homomorphic computations:** A
(a) computes $\{h(b_j)^{k_2 k_1}, E(t_j)\}$;
(b) computes the set I of intersection indices where

$$I = \{j : h(a_i)^{k_1 k_2} = h(b_j)^{k_2 k_1} \text{ for some } i\};$$

(c) samples a uniformly random 1024-bit value of r.

(d) selects uniformly random integer values of r_1, r_2, where $0 \leq r_1 \leq 2^{128} - 1$, $2^{511} \leq r_2 \leq 2^{512} - 1$ with r_1 satisfying

$$r_1 \equiv r \bmod k$$

where $k = |I|$.
(e) additive homomorphically computes

$$E\left(r_2 + \frac{r - r_1}{k} \sum_{i \in I} t_i\right).$$

A sends r and $E\left(r_2 + \frac{r-r_1}{k} \sum_{i \in I} t_i\right)$ to B.
5. B**'s decryption phase:** B performs decryptions of the ciphertext received from A and computes (division over real numbers)

$$\frac{1}{|I|} \sum_{i \in I} t_i \approx \frac{1}{r}\left(r_2 + \frac{r - r_1}{k} \sum_{i \in I} t_i\right).$$

Theorem 1. *Protocol 1 correctly outputs the intersection mean which is arbitrary close to the exact value.*

Proof. We first note that A knows the value of $|I| = k$ in Step 4 of the protocol. We need to show that

$$\frac{1}{|I|} \sum_{i \in I} t_i \approx \frac{1}{r}\left(r_2 + \frac{r - r_1}{k} \sum_{i \in I} t_i\right). \tag{3}$$

For brevity of notation, let $r' = \frac{r-r_1}{k}$, where $r' \gg r_1, r_2$. Thus, r can be expressed as $r = kr' + r_1$. It follows that

$$\frac{1}{r}\left(r_2 + \frac{r - r_1}{k} \sum_{i \in I} t_i\right) = \frac{\frac{r_2}{r'} + \sum_{i \in I} t_i}{\frac{r_1}{r'} + k} \approx \frac{1}{|I|} \sum_{i \in I} t_i \tag{4}$$

since $\frac{r_1}{r'}, \frac{r_2}{r'} \approx 0$.

It remains to show that the above approximation is in fact close. Denote the exact mean value to be M and its approximation be M'. To achieve this, we formalize the notion of closeness by showing that the absolute difference between M and M' is within the value of M multiplied by a factor dependent on r which can be made arbitrary close to 0. More formally, M and M' are close if and only if

$$|M - M'| < \frac{M}{f(r)} \tag{5}$$

where $f(r)$ is a monotonically increasing function of r such that $\lim_{r\to\infty} \frac{1}{f(r)} = 0$.

This implies that M' can be made arbitrary close to M for corresponding appropriate choices on the size of r. Applying a series of computations, we arrive at

$$|M - M'| = \left| \frac{r_2 k - r_1 \sum_{i\in I} t_i}{k(r'k + r_1)} \right| < \frac{\max(r_1, r_2) \sum_{i\in I} t_i}{rk} < \frac{2^{512} \sum_{i\in I} t_i}{rk} = \frac{M}{f(r)} \quad (6)$$

where $f(r) = \frac{r}{2^{512}}$. It is clear that $f(r)$ is a monotonically increasing function and that $\lim_{r\to\infty} \frac{1}{f(r)} = 0$ which proves that M' is close to M. □

4.1 On the chosen sizes of r, r_1, r_2

As shown earlier, the larger the value of r, the closer the approximation. Moreover, this approximation can be made arbitrary close for arbitrary large values of r (along with corresponding large parameter sizes of E). In practice, a sufficiently large value of r already provides a very tight approximation. The size choices of r_1 and r_2 are bounded below by 2^{128} to prevent exhaustive search attacks. After the sizes of r_1, r_2 are set, the size of r can be chosen to sufficiently overwhelm r_1, r_2. In our case, we set r to be of size 1024-bit which is sufficient for M' to M to be extremely close. In particular,

$$|M - M'| < 2^{-510} M \quad (7)$$

which suffices for all practical intent.

4.2 Flexibility of Protocol

It should be noted that the exact value of mean in fact reveals additional information about the sum or cardinality. For instance, if the mean is an integer M then it follows that the sum is divisible by M. More generally, if the mean is a rational number $\frac{a}{b}$ with $\gcd(a, b) = 1$, then the sum is divisible by a and the cardinality is divisible by b. As discussed in the preliminary section, we do not consider such implicit information which be can deduced from the output functionality. Nevertheless, Protocol 1 has the added benefit of flexibility which enables the adjustment of varying degrees of approximation tightness if one wishes to circumvent the above issues. This can be achieved by adjusting the size of a randomly sampled r. The approximation weakens with decreasing sizes of r. More generally, for a random sample r of x-bit, $x \geq 515$, the difference yields

$$|M - M'| < 2^{512-x} M. \quad (8)$$

4.3 Security Analysis

The security arising from the communication in Step 2 and Step 3 follows from the validity of the Decisional Diffie-Hellman assumption as well as the hardness of the Decisional Composite Redisuosity Problem. Hence, the remaining security and privacy aspects to consider occur in Step 4 where B receives r and $E\left(r_2 + \frac{r-r_1}{k}\sum_{i\in I} t_i\right)$. Since r is sampled uniformly at random from a collection of 1024-bit integers, B is unable to distinguish r from a random uniformly selected 1024-bit integer. As before, denote M to be the exact value of the intersection mean. Thus, B obtains $r_2 + \frac{r-r_1}{k}\sum_{i\in I} t_i = r_2 + (r-r_1)M$. Moreover, only r and M[4] are known to B and thus can only effectively compute $r_2 - r_1 M$. Therefore, B is unable to solve for r_1 and r_2 explicitly since there are two unknowns in a single expression. This serves the purpose of hiding the cardinality and intersection sum. In fact, we provide a stronger notion of security by showing statistical indistinguishability. We begin with a standard security definition.

Definition 3. *Let X and Y be two distributions over $\{0,1\}^n$. The statistical distance of X and Y, denoted by $\Delta(X,Y)$ is defined to be*

$$\Delta(X,Y) = \max_{U\subseteq\{0,1\}^n} |Pr[X\in U] - Pr[Y\in U]|$$

$$= \frac{1}{2}\sum_{v\in Supp(X)\cup Supp(Y)} |Pr[X=v] - Pr[Y=v]|\,. \tag{9}$$

X is ϵ statistically indistinguishable from Y if $\Delta(X,Y)\leq\epsilon$.

In particular, we show that $r_2 - r_1 M$ is statistically indistinguishable from uniformly distributed 512-bit integers, when M is a fixed positive integer. It can be assumed here that the intersection mean M is at most 80-bit in all practical settings. Denote random variables $R_1 = \text{unif}[0,c]$ and $R_2 = \text{unif}[a,b]$ to be the discrete uniform distributions over $\mathbb{Z}\cap[0,c]$ and $\mathbb{Z}\cap[a,b]$ respectively such that $cM \ll a \ll b$. We first establish the following.

Theorem 2.

$$\Delta(R_2 - R_1 M, R_2) = \frac{cM}{2(b-a+1)}. \tag{10}$$

[4] Technically only M', which is a close approximation to M can be computed by B. In essence, this distinction is largely irrelevant in this specific context as we evaluate a stronger security setting than required where B has the knowledge of both M and M'.

Proof. Denote $X = R_2 - R_1 M$. Upon a series of careful computations within $\text{supp}(X)$, we obtain the following.

$$P(X=i) = \begin{cases} \frac{c-j}{(c+1)(b-a+1)} & \begin{cases} \forall i \in [a-(j+1)M, a-jM-1], \\ \forall j \in [0, c-1] \end{cases} \\ \frac{1}{b-a+1} & \forall i \in [a, b-cM] \\ \frac{c-j}{(c+1)(b-a+1)} & \begin{cases} \forall i \in [b-(c-j)M+1, b-(c-j-1)M], \\ \forall j \in [0, c-1] \end{cases} \end{cases}$$

Moreover, since $R_2 = \text{unif}[a, b]$,

$$P(R_2 = i) = \frac{1}{b-a+1} \qquad \forall i \in [a, b]. \tag{11}$$

The statistical distance of X and R_2 can now be computed via

$$\Delta(X, R_2) = \frac{1}{2} \sum_{v \in \text{Supp}(X) \cup \text{Supp}(R_2)} |Pr[X=v] - Pr[R_2 = v]|$$

$$= \frac{1}{2} \sum_{j=0}^{c-1} \frac{M(c-j)}{(c+1)(b-a+1)} + \frac{1}{2} \sum_{j=0}^{c-1} \frac{M}{b-a+1} - \frac{M(c-j)}{(c+1)(b-a+1)}$$

$$= \frac{cM}{2(b-a+1)}.$$

□

In Protocol 1, $a = 2^{511}$, $b = 2^{512} - 1$, $c = 2^{128} - 1$ and M is assumed to be at most 80-bit. Hence in Protocol 1,

$$\Delta(R_2 - R_1 M, R_2) \leq 2^{-300}. \tag{12}$$

This shows that $r_2 - r_1 M$ is 2^{-300} statistically indistinguishable from uniformly distributed 512-bit integers when M is a positive integer. In the case where $M = \frac{a}{b}$ is a non-integer positive rational number such that $a \ll r_2$, a similar argument can be applied to show that $r_2 b - r_1 a$ is statistically indistinguishable from uniformly distributed 512-bit integers which are multiplied by a factor of b.

4.4 Geometric mean

Our method can be adapted to output the intersection geometric mean functionality without revealing the intersection size to B. One such instance of this statistical measure can arise in Econometrics as specified by the generalized Cobb-Douglas production function [23], where its inputs can represent working hours of labourers and each exponent is the reciprocal of the number of inputs.

The geometric mean of a data set $\{t_1, t_2, \ldots, t_k\}$ is given by $\left(\prod_{i=1}^{k} t_i\right)^{\frac{1}{k}}$. A natural line of approach is to replace additive homomorphic encryption with a multiplicative homomorphic encryption (e.g. EL-Gamal [25], RSA [53]) in view of the multiplicative structure of the output. This works perfectly fine if the intersection size is exposed to B. However, in the security model where the intersection size is kept secret from B, there is no known efficient public-key based protocol utilizing multiplicative homomorphic encryption which can achieve this with reasonable accuracy. We present a method to attain this desired functionality using ideas based on our earlier approach for arithmetic mean.

Without being overly verbose, we do not detail the fully fledged protocol but instead highlight the crucial steps. Suppose $t_i \in [1, t+1]$. We first seek an injective function $f_c : t_i \to \lfloor c \log t_i \rceil$ for some positive integer constant c. Here $\lfloor . \rceil$ denotes the nearest integer function in accordance with the most recent IEEE Standard for floating-point arithmetic [5]. The value of c is chosen by B, the party who holds the list of associated values. We show that f_c can be constructed by taking $c \geq t+1$.

Theorem 3. *f_c is injective $\forall\ c \geq t+1$.*

Proof. We first establish that

$$\log\left(1+\frac{1}{t}\right) > \frac{1}{t+1}. \tag{13}$$

The above can easily be shown by applying a change of variable $t = \frac{1-x}{x}$ and deducing that $e^x < \frac{1}{1-x}$ with a comparison of their respective Taylor series. It follows that for all $c \geq t+1$ and for all $t_i \neq t_j$,

$$|c\log t_i - c\log t_j| \geq c\log(t+1) - c\log t \geq (t+1)\log\left(1+\frac{1}{t}\right) > 1 \tag{14}$$

$$\implies |\lfloor c\log t_i \rceil - |\lfloor c\log t_j \rceil| > 1 \implies \lfloor c\log t_i \rceil \neq \lfloor c\log t_j \rceil$$

which asserts that f_c is injective. □

The injectivity condition is stipulated to ensure that no two distinct values of t_i's correspond to the same image under f_c. The protocol for the intersection geometric mean proceeds by replacing t_i with $\lfloor c \log t_i \rceil$ given in Protocol 1. One other difference lies in the final decryption step performed by B. More specifically in the final step, B obtains the intersection geometric mean by computing

$$e^{\frac{1}{cr}\left(r_2+\frac{r-r_1}{k}\sum_{i\in I}\lfloor c\log t_i\rceil\right)} \approx e^{\frac{1}{c|I|}\sum_{i\in I}\lfloor c\log t_i\rceil} \approx e^{\frac{1}{|I|}\sum_{i\in I}\log t_i} = \left(\prod_{i\in I} t_i\right)^{\frac{1}{|I|}}. \tag{15}$$

In general, larger values of c provide a greater precision. In practice, such large values of c can always be chosen since the modulus of the Paillier encryption is much larger than $\max\{\log t_i\}$. The proof of correctness and security mirrors that of the intersection mean.

4.5 Extensions to Variance, Standard Deviation, Skewness and Kurtosis

Our techniques presented in this section can be further extended to compute various other statistical functions of the associated values in the intersection. The general overall idea is for B to receive the values of nth-order moment about the origin in order to compute the nth central moment (without knowledge of the intersection size).

Let R be a random variable. In our context, R can be considered to be the discrete uniform distribution over the associated values in the intersection. The nth-order moment about the origin μ'_n is defined to be $\mu'_n = \mathbb{E}[R^n]$. The nth central moment μ_n is defined to be $\mu_n = \mathbb{E}[(R - \mathbb{E}[R])^n]$. For example, the mean in this case is $\mu'_1 = \mathbb{E}[R]$.

Variance and standard deviation provide a measure of the amount of dispersion of a list of values. A low standard deviation indicates that the values tend to be clustered around its mean, while a high standard deviation indicates that the values are dispersed over a wider range of values. In step 3 of the protocol, B sends both $E(t_j)$ and $E(t_j^2)$ to A. This enables A to return close approximate values of $\mathbb{E}[R]$ and $\mathbb{E}[R^2]$ to B. The variance can then be simply computed by $\mathrm{Var}(R) = \mathbb{E}[R^2] - (\mathbb{E}[R])^2$ and standard deviation $\sigma = \sqrt{\mathrm{Var}(R)}$. At the end of this protocol, B outputs the mean and standard deviation (or variance). At the same time, B's knowledge of $\mathbb{E}[R^2]$ during this process does not reveal any additional information since that quantity can be derived from any generic protocol which outputs the mean and standard deviation (or variance). The protocols for the skewness and kurtosis output functionalities can be similarly constructed.

5 Private Set Intersection-Percentile Sum

Intersection-percentile sum can be regarded as a generalization of Ion *et al.*'s intersection-sum. Intersection-percentile sum provides an added flexibility of computing the sum of the x-th percentile of associated values within the intersection for any specified value of x. In the case of Ion *et al.*'s protocol, the value of x is restricted to 100. The intersection range is defined to be minimum and maximum of the associated values corresponding to the items within the set intersection. Our generalized method also enables the output of the intersection range [5] when necessary. We first describe a protocol which outputs the intersection sum and intersection range followed by a protocol which outputs the percentile sum.

[5] A caveat in relation to the min and max outputs is that the identifiers with associated integers corresponding to these min and max values can be potentially revealed to B. As discussed before, this is implicit from the actual output functionality.

PROTOCOL 2 (Intersection-Sum & Range)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$, B outputs intersection sum and intersection range.

1. **Setup:** Identical to Protocol 1.
2. A**'s encryption phase:** Identical to Protocol 1.
3. B**'s encryption phase:** B
(a) selects a random private exponent $k_2 \in G$;
(b) computes $h(a_i)^{k_1 k_2}$;
(c) computes $\{h(b_j)^{k_2}, E(t_j)\}$.
B returns $h(a_i)^{k_1 k_2}$ in shuffled order to A. B sends $\{h(b_{\sigma(j)})^{k_2}, E(t_{\sigma(j)})\}_{j=1}^{n}$ to A, where σ is a permutation of j such that $t_{\sigma(j)} \geq t_{\sigma(j+1)}$.
4. **Matching & homomorphic computations:** A
(a) computes $\{h(b_{\sigma(j)})^{k_2 k_1}, E(t_{\sigma(j)})\}$;
(b) computes the set I of intersection indices where

$$I = \{\sigma(j) : h(a_i)^{k_1 k_2} = h(b_{\sigma(j)})^{k_2 k_1} \text{ for some } i\};$$

(c) determines $j_{\min}$ where $j_{\min}$ is the smallest value of j satisfying $h(a_i)^{k_1 k_2} = h(b_{\sigma(j)})^{k_2 k_1}$ for some i;
(d) determines $j_{\max}$ where $j_{\max}$ is the largest value of j satisfying $h(a_i)^{k_1 k_2} = h(b_{\sigma(j)})^{k_2 k_1}$ for some i;
(e) additive homomorphically computes $E\left(\sum_{i \in I} t_i\right)$.
A sends $E\left(\sum_{i \in I} t_i\right)$, $E(t_{\sigma(j_{\min})})$ and $E(t_{\sigma(j_{\max})})$ to B.
5. B**'s decryption phase:** B performs decryptions of the ciphertexts received from A to obtain the desired sum as well as the range given by

$$\min_{i \in I} t_i = t_{\sigma(j_{\max})} \text{ and } \max_{i \in I} t_i = t_{\sigma(j_{\min})}.$$

Protocol 2 can be generalized to output the intersection sum within any specified range percentile. That is to output the sum of all associated values which fall within the P_1-th and P_2-th percentile of the intersection. Here, $0 \leq P_1 < P_2 \leq 100$. We provide a condensed version of such a scheme given in Protocol 3.

PROTOCOL 3 (Intersection-Percentile Sum)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$, B outputs intersection sum within the P_1-th percentile and the P_2-th percentile.

1. **Setup:** Identical to Protocol 1.
2. A**'s encryption phase:** Identical to Protocol 1.
3. B**'s encryption phase:** Identical to Protocol 2.
4. **Matching & homomorphic computations:** A
(a) computes $\{h(b_{\sigma(j)})^{k_2 k_1}, E(t_{\sigma(j)})\}$;
(b) computes the set I of intersection indices where

$$I = \{\sigma(j) : h(a_i)^{k_1 k_2} = h(b_{\sigma(j)})^{k_2 k_1} \text{ for some } i\};$$

(c) additive homomorphically computes

$$E\left(\sum_{i=\lceil \frac{kP_1}{100} \rceil}^{\lceil \frac{kP_2}{100} \rceil} t'_i\right)$$

where $k = |I|$ and $\{t'_i\}_{i=1}^{k}$ is a permutation of $\{t_i\}_{i \in I}$ such that $t'_i \leq t'_{i+1}$.
A sends $E\left(\sum_{i=\lceil \frac{kP_1}{100} \rceil}^{\lceil \frac{kP_2}{100} \rceil} t'_i\right)$ to B.
5. B**'s decryption phase:** B performs decryption of the ciphertext received from A to obtain the desired intersection sum within the given percentile thresholds.

5.1 A note on Intersection-Range Query Sum

Protocol 3 outputs the sum of the associated values that fall within a specified percentile of the intersection. Here, we briefly outline an overview of a protocol which outputs the intersection-sum of the associated values that fall within a specified range. In this setting, the desired output is the sum of associated values in the intersection such that each value composing of the sum lies within a specified interval $[a, b]$. Party B holding the associated values truncates her set to only include elements with associated values in the specified interval $[a, b]$. Ion *et al.*'s protocol can then be applied to this truncated set to obtain the desired output.

6 Intersection-Sum with Approximate Composition (1)

Trivially, the intersection sum output S reveals that there are less than l elements (or identifiers) in the intersection with associated values greater than $\frac{S}{l}$. In many scenarios, B wishes to know more about the composition of the sum. In particular, given the intersection sum, B wishes to have an estimate of the number of elements in the intersection with associated value of at most $\hat{t}$ (i.e. its approximate sum composition). This information cannot be captured merely by the knowledge of intersection sum. To that end, we present two protocols,

labelled as type 1 and type 2 which enable the output of intersection sum along with its approximate sum composition. Type 1 protocol incurs a lower communication cost (its communication overhead in addition to the intersection sum is negligible) but is executed at a lower security setting compared to type 2 protocol.

PROTOCOL 4 (Sum Composition type 1)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$, B outputs $\sum_{i \in I} t_i$ and an upper bound for $\sum_{i \in I} \frac{1}{t_i}$, where I is the set of indices of $b_i \in X \cap Y'$.

1. **Setup:** Identical to Protocol 1.
2. A**'s encryption phase:** Identical to Protocol 1.
3. B**'s encryption phase:** B
(a) selects a random private exponent $k_2 \in G$;
(b) computes $h(a_i)^{k_1 k_2}$;
(c) computes $\{h(b_j)^{k_2}, E(t_j)\}$.
B returns $h(a_i)^{k_1 k_2}$ in shuffled order to A. B sends $\{h(b_{\sigma(j)})^{k_2}, E(t_{\sigma(j)})\}_{j=1}^{n}$ to A, where σ is a permutation of j such that $t_{\sigma(j)} \geq t_{\sigma(j+1)}$. B sends M to A where

$$M \geq \max\left\{\frac{t_{\sigma(j)}}{t_{\sigma(j+1)}}\right\}.$$

4. **Matching & homomorphic computations:** A
(a) computes $\{(h(b_j)^{k_2 k_1}, E(t_j)\}$;
(b) computes the set I of intersection indices where

$$I = \{j : h(a_i)^{k_1 k_2} = h(b_j)^{k_2 k_1} \text{ for some } i\};$$

(c) additive homomorphically computes $E\left(\sum_{i \in I} t_i\right)$;
(d) samples uniformly random r, r_1, r_2 from a sufficiently large set of positive integers such that $r \gg r_1, r_2$ and $\frac{r_1}{r_2} \geq k$, where $k = |I|$;
(e) computes $r_1 + rk(M^k - 1)$ and additive homomorphically computes

$$E\left(r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1} t'_i\right),$$

where $\{t'_i\}_{i=1}^{k}$ is a permutation of $\{t_i\}_{i \in I}$ s.t. $t'_{i+1} \leq t'_i$.
A sends $E\left(\sum_{i \in I} t_i\right)$, $E\left(r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1} t'_i\right)$ and $r_1 + rk(M^k - 1)$ to B.

5. B**'s decryption phase:** B performs decryptions of the ciphertexts received from A to obtain $\sum_{i\in I} t_i$ and an upper bound for $\sum_{i\in I} \frac{1}{t_i}$ given by

$$\sum_{i\in I} \frac{1}{t_i} \leq \frac{r_1 + rk(M^k - 1)}{r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1} t'_i}.$$

Theorem 4. *Protocol 4 outputs* $\sum_{i\in I} t_i$ *and an upper bound for* $\sum_{i\in I} \frac{1}{t_i}$.

Proof. It is clear from the final step of the protocol that $\sum_{i\in I} t_i$ is an output. Therefore, it remains to show that

$$\sum_{i\in I} \frac{1}{t_i} \leq \frac{r_1 + rk(M^k - 1)}{r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1} t'_i}. \tag{16}$$

Since$\{t'_i\}_{i=1}^k$ is the permutation of $\{t_i\}_{i\in I}$,

$$\sum_{i\in I} \frac{1}{t_i} = \sum_{i=1}^{k} \frac{1}{t'_i}. \tag{17}$$

Note that $\frac{1}{t'_i} \leq \frac{1}{t'_{i+1}}$ and $t'_i \leq M t'_{i+1}$. The latter inequality implies that

$$t'_1 \leq M t'_2 \leq M^2 t'_3 \leq \cdots \leq M^{k-1} t'_k. \tag{18}$$

Hence by Chebyshev's sum inequality,

$$k\left[\sum_{i=1}^{k} M^{i-1} t'_i \left(\frac{1}{t'_i}\right)\right] \geq \left(\sum_{i=1}^{k} M^{i-1} t'_i\right)\left(\sum_{i=1}^{k} \frac{1}{t'_i}\right). \tag{19}$$

After a number of simplifications, it follows that

$$\sum_{i=1}^{k} \frac{1}{t'_i} \leq \frac{k(M^k - 1)}{(M-1)\sum_{i=1}^{k} M^{i-1} t'_i}. \tag{20}$$

Since $\frac{r_1}{r_2} \geq k$ from the choices of r_1, r_2,

$$\frac{r_1}{r_2} \geq k \geq \frac{k(M^k - 1)}{(M-1)\sum_{i=1}^{k} M^{i-1} t'_i}, \tag{21}$$

which yields

$$\frac{r_1}{r_2} \geq \frac{k(M^k-1)}{(M-1)\sum_{i=1}^{k} M^{i-1}t'_i}$$

$$\iff \frac{k(M^k-1)}{(M-1)\sum_{i=1}^{k} M^{i-1}t'_i} \leq \frac{r_1 + rk(M^k-1)}{r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1}t'_i}$$

and the result of (16) follows from (17) and (20). □

6.1 Applicability of Sum Composition

Let the sum composition measure T be denoted to be

$$T = \frac{r_1 + rk(M^k-1)}{r_2 + r(M-1)\sum_{i=1}^{k} M^{i-1}t'_i}. \tag{22}$$

Theorem 5. *There are less than l elements in the intersection with associated integer values of at most $\hat{t}$ where $\hat{t} \in \mathbb{Z}^+$ satisfying $\hat{t} < \frac{l}{T}$.*

Proof. The proof is straightforward. Suppose for a contradiction that there are at least l elements in the intersection with associated integer values of at most $\hat{t}$. Then we have the following chain of inequalities:

$$T < \frac{l}{\hat{t}} \leq \sum_{i \in I} \frac{1}{t_i} \leq T, \tag{23}$$

a contradiction. □

For instance when $l = 1$, it can be established that there are no small associated integer values under $\frac{1}{T}$ contained in the set intersection. In other words, every element in the intersection has an associated value of at least $\frac{1}{T}$.

It should be noted that the applicability of this measure of sum composition is dependent on the distribution of the associated values held by B. Generally, this output functionality is more useful when the spread of associated values is sufficiently large. In practice, B who holds the associated values can decide whether to initiate these two protocols based on her dataset.

7 Intersection-Sum with Approximate Composition (2)

Type 1 enables the transmission of an approximate sum composition without any substantial increase in communication over the intersection-sum. However, that requires B to reveal an upper bound of M to A. In this section, we describe

a protocol where communication cost is not an overriding consideration without the need to disclose an upper bound of M to A . Type 2 also results in a tighter output approximation compared to type 1. A key ingredient involves an injective mapping of the set of reciprocals of positive integers to the set of positive integers which preserves addition. Denote f_c to be such an injective map such that $f_c : t_i \rightarrow \left\lceil \frac{c}{t_i} \right\rceil$ for a suitable large constant c. The detailed selection of c will be discussed later in this section.

PROTOCOL 5 (Sum Composition type 2)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$, B outputs $\sum_{i \in I} t_i$ and an upper bound for $\sum_{i \in I} \frac{1}{t_i}$, where I is the set of indices of $b_i \in X \cap Y'$.

1. **Setup:** Identical to Protocol 1.
2. A**'s encryption phase:** Identical to Protocol 1.
3. B**'s encryption phase:** B
(a) selects a random private exponent $k_2 \in G$;
(b) computes $h(a_i)^{k_1 k_2}$;
(c) computes $\{h(b_j)^{k_2}, E(t_j),\ E(f_c(t_j))\}$.
B returns $h(a_i)^{k_1 k_2}$ in shuffled order to A. B sends $\{h(b_j)^{k_2}, E(t_j),\ E(f_c(t_j))\}$ to A.
4. **Matching & homomorphic computations:** A
(a) computes $\{h(b_j)^{k_2 k_1}, E(t_j),\ E(f_c(t_j))\}$;
(b) computes the set I of intersection indices where

$$I = \{j : h(a_i)^{k_1 k_2} = h(b_j)^{k_2 k_1} \text{ for some } i\};$$

(c) additive homomorphically computes

$$E\left(\sum_{i \in I} t_i\right) \quad \text{and} \quad E\left(\sum_{i \in I} f_c(t_i)\right).$$

A sends $E\left(\sum_{i \in I} t_i\right)$ and $E\left(\sum_{i \in I} f_c(t_i)\right)$ to B.
5. B**'s decryption phase:** B performs decryptions of the ciphertexts received from A to obtain $\sum_{i \in I} t_i$ and an upper bound for $\sum_{i \in I} \frac{1}{t_i}$ given by

$$\sum_{i \in I} \frac{1}{t_i} \leq \frac{1}{c} \sum_{i \in I} f_c(t_i).$$

Theorem 6. *Protocol 5 outputs* $\sum_{i \in I} t_i$ *and an upper bound for* $\sum_{i \in I} \frac{1}{t_i}$.

Proof. As before, the protocol correctly executes $\sum_{i \in I} t_i$ as an output. Moreover, $\frac{1}{t_i} = \frac{1}{c}\left(\frac{c}{t_i}\right) \leq \frac{1}{c}\left\lceil \frac{c}{t_i} \right\rceil$. The result follows by summing over all i's in I. □

7.1 On the selection of c

Suppose the values of t_i's are bounded by x bits. We show that taking c to be of size $2x$ bits admits f_c to be injective. Indeed, let t_i, t_j be two distinct associated values. Without loss of generality, assume $t_i < t_j$, Then

$$\left| \frac{c}{t_i} - \frac{c}{t_j} \right| = c\left(\frac{t_j - t_i}{t_i t_j}\right) \geq 1 \tag{24}$$

since c is of size $2x$ bits. It follows that

$$\left| \frac{c}{t_i} - \frac{c}{t_j} \right| \geq 1 \Rightarrow \left\lceil \frac{c}{t_i} \right\rceil \neq \left\lceil \frac{c}{t_j} \right\rceil \tag{25}$$

which proves injectivity.

7.2 Comparisons between Type 1 & Type 2

Type 2 induces a larger communication overhead as compared to type 1. This increase in communication essentially occurs in Step 3 of type 2 where B sends an additional public key encryption $E(f_c(t_i))$ to A for each element in Y.

Table 1: Recommended RSA key length from NIST

security level (κ)	80	112	128	192	256
RSA key length	1024	2048	3072	7680	15360

The NIST report [13] details the recommended RSA key length (in bits) to achieve a κ-bit security as given in Table 1. In the case of E being the Paillier encryption, a 2048-bit length modulus corresponds to a 112-bit security level. This translates to a ciphertext of length 4096-bit for Paillier encryption. Thus, this increase in communication overhead is approximately in the region of $4096n$ bits.

On the other hand for large intersection sizes, type 2 is more practical and has a lower computational cost. An added benefit of type 2 is that the quantity M is not disclosed as compared to type 1.

7.3 Harmonic mean

The harmonic mean output functionality can be achieved using a combination of techniques highlighted in Protocol 1 and Protocol 5. The harmonic mean of a data set $\{t_1, \ldots, t_k\}$ is given by $k\left(\sum_{i=1}^{k} \frac{1}{t_i}\right)^{-1}$. The initial steps of this protocol follows that of Protocol 1 such that all instances of t_i are replaced with $\left\lfloor \frac{c}{t_i} \right\rceil$. In the final step, B obtains the harmonic mean by computing

$$\frac{k}{\left(\sum_{i=1}^{k} \frac{1}{t_i}\right)} \approx \frac{cr}{r_2 + \frac{r-r_1}{k} \sum_{i \in I} \lfloor \frac{c}{t_i} \rceil}. \tag{26}$$

The proofs of correctness and security follows from the corresponding proofs associated with Protocol 1. As with the geometric mean, larger values of c result in higher precision.

In datasets where $t = \max\{t_i\}$ is relatively small, the function f_c can be simplified to $f_c(t_i) = \frac{c}{t_i}$ where $c = lcm(1, 2, \ldots, t)$. Clearly, f_c remains injective and its image is a subset of the positive integers. The only constraint is a practical one and that is to ensure that the size of c for this simplified function has to be well within the modulus of the Paillier encryption. Define the second Chebyshev function $\psi(t)$ to be the sum over all prime powers not exceeding t i.e. $\psi(t) = \sum_{p^k \leq t} \log p$. Thus c can be expressed as $c = e^{\psi(t)}$. Applying Theorem 12 of [54], we obtain an upper bound $c < e^{1.03883t}$. It can be deduced that c is within the limits of a typical Paillier encryption modulus for relatively small values of t. As an example for $t = 100$,

$$c < e^{103.883} < 2^{150} \tag{27}$$

which is much smaller than the modulus of a Paillier encryption of size well over a thousand bits (e.g. the implementations of [35] as well as ours apply a modulus of 1536 bits). Our computations estimate that $t = 1000$ can be supported for the above given parameters. In settings where the values of t are large, one can simply take $f_c(t_i) = \left\lfloor \frac{c}{t_i} \right\rceil$ for a sufficiently large c as outlined earlier.

8 Reverse Intersection-Statistical Protocols

This section describes protocols for which the destination of the output is reversed (i.e. A obtains this output instead of B). The key differences between our protocols presented in this section compared with [35] are that our protocols enable the hiding of the intersection cardinality from B and also support other statistical outputs apart from the sum. The trade-off is that an additional round of communication is required. We present a general method of how the

output of general statistics (presented in this paper) can be reversed. An explicit example of the reverse intersection-percentile sum protocol is provided as follows.

PROTOCOL 6 (Reverse Intersection-Percentile Sum)

Input: A inputs set X; B inputs set Y.
Output: A outputs $|X \cap Y'|$ and intersection percentile sum. B outputs $\perp$.

1. **Setup:** Identical to Protocol 1.
2. A**'s encryption phase:** Identical to Protocol 1.
3. B**'s encryption phase:** Identical to Protocol 2.
4. **Matching & homomorphic computations:** A
(a) samples a uniformly random 1024-bit value r;
(b) computes $\{h(b_{\sigma(j)})^{k_2 k_1}, E(t_{\sigma(j)})\}$;
(c) computes the set I of intersection indices where

$$I = \{\sigma(j) : h(a_i)^{k_1 k_2} = h(b_{\sigma(j)})^{k_2 k_1} \text{ for some } i\};$$

(d) additive homomorphically computes

$$E\left(r + \sum_{i=\lceil \frac{kP_1}{100} \rceil}^{\lceil \frac{kP_2}{100} \rceil} t'_i\right)$$

where $k = |I|$ and $\{t'_i\}_{i=1}^{k}$ is a permutation of $\{t_i\}_{i \in I}$ such that $t'_i \leq t'_{i+1}$.
A sends $E\left(r + \sum_{i=\lceil \frac{kP_1}{100} \rceil}^{\lceil \frac{kP_2}{100} \rceil} t'_i\right)$ to B.
5. B**'s decryption phase:** B performs decryption of the ciphertext received to obtain the plaintext S. B sends S to A.
6. A**'s output:** A obtains the intersection percentile sum by computing $S - r$.

For a general statistical measure S where $E(S)$ is sent to B (thereby B knowing S as a consequence), the destination output can be reversed with A sending $E(r + S)$ instead for a suitable large random value of r. B who holds the private key can then perform this decryption and sends the resulting $r + S$ back to A. A obtains the desired statistical measure output by subtracting her private value of r from $r + S$. In the special case where the desired output of A is the intersection mean, S can simply be taken to be the intersection sum. Indeed in the reverse protocol, the intersection sum and intersection mean are equivalent since the intersection cardinality is always known to A.

8.1 Security Analysis

Apart from the security based on the Decisional Diffie-Hellman assumption and the hardness of the Decisional Composite Residuosity problem, our reverse protocols also have to take into account that the additional communication round of sending $E(r+S)$ to B (who holds the private key) preserves the privacy of S. We go about this by showing how statistical indistinguishability can be achieved in our protocols with the selection of random large r values.

Let S and S' be any fixed values, where S and S' refer to the same statistical measure of size at most x-bit. Denote R to be the random variable uniformly distributed among the y-bit integer. By taking $X = R + S$ and $Y = R + S'$, it can be computed that $\Delta(X, Y) \approx 2^{x-y}$.

This statistical distance can be made arbitrary small by selecting large values of r (along with corresponding large parameter sizes of E). In practice, it can be assumed that any statistical measure S is at most 80-bit. Our protocol selects a large 1024-bit value of r which provides a statistical security of 2^{-940}.

9 Implementation & Performance

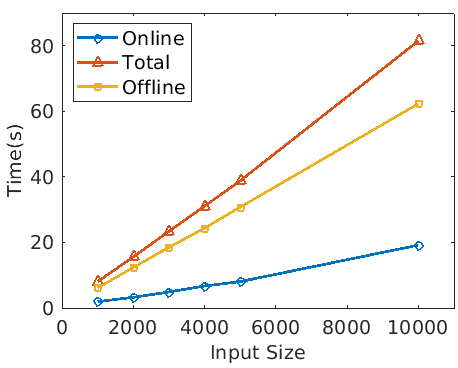
We implemented *PSI-Stats* in C++. The benchmark machine is desktop workstation running on a single-thread with an Intel Core i7-7700 CPU @ 3.60GHz and 28GB RAM. The bandwidth is 4867 Mbps with round-trip time of 0.02 ms. As a measure of comparisons, we also run the intersection-sum implementation of [35] with the results recorded in Table 2 under the "Sum" column. The respective input sizes m, n are equal and the comparisons are based on the running time (in seconds) as well as communication cost (in MB). All intersection sizes are standardized at 200 unless otherwise stated. We set the value of $c = 10^9$ for the geometric mean protocol. In running Protocol 1, we also provide the readers with the results of the actual generalized means alongside the outputs that it obtained from the execution of the protocol. The output generalized means values given in the tables are all rounded down to the nearest whole number.

We apply similar parameters as with the experiments conducted in [35]. The elements/identifiers in the generated datasets are 128-bit strings, with associated values being at most 32 bits long (the specific testing range is set between 1 to 100). The sum of the associated values is also bounded by 32 bits. The input set sizes range from 1000 to 100000. An elliptic curve with 224 bit group elements constitute the group G. The hash function SHA-256 is utilized for h. The Paillier encryption involves the product of two 768 bit primes which yields a plaintext space of 1536 bits and ciphertext of length 3072 bits.

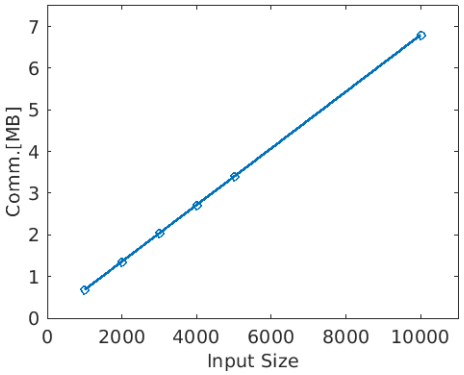
In addition, we have segmented the total time into disjoint durations of the offline and online phases. The offline phase refers to the pre-computation process where the parties can perform offline computations of their respective individual dataset even before the initial round of communication commences. The online phase begins from the initial round of communication to the end of the protocol. In practice, the online phase duration generally provides a more relevant indicator of practical performance as opposed to the total time taken. The results are

Table 2: Performance of Intersection-Range/Sum protocol.

	Sum [35]		Sum & Range	
Input Size	Time(s)	Comm.[MB]	Time(s)	Comm.[MB]
1000	7.88	0.6798	7.87	0.6809
2000	15.54	1.3588	15.50	1.3599
3000	23.29	2.0377	23.24	2.0389
4000	30.98	2.7167	30.97	2.7179
5000	38.78	3.3957	38.52	3.3969
10000	79.69	6.79	78.99	6.80
50000	387.88	33.95	385.53	33.96
100000	796.96	67.9	789.64	68

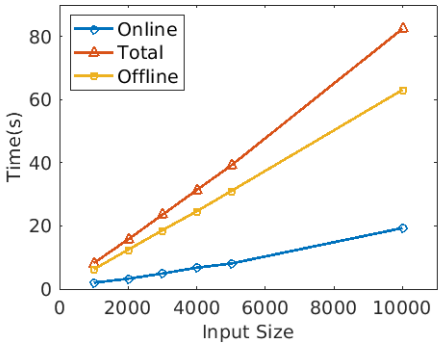


(a) Time

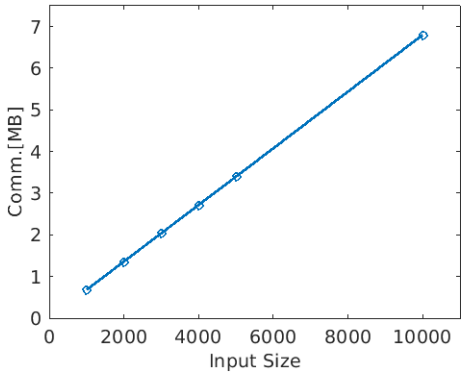


(b) Communication

Fig. 1: Performance on Arithmetic Mean



(a) Time



(b) Communication

Fig. 2: Performance on Geometric Mean

Table 3: Performance of Private Intersection-arithmetic mean protocol.

	Time(s)			Comm.[MB]	Actual Value	Output (rounded down)
Input Size	Offline	Online	Total			
1000	6.11	1.94	8.05	0.6799	51.31	51
2000	12.32	3.25	15.57	1.3589	48.91	48
3000	18.52	4.87	23.39	2.0379	51.76	51
4000	24.34	6.71	31.05	2.7169	44.525	44
5000	30.76	8.04	38.8	3.3958	48.835	48
10000	62.3	19.18	81.48	6.80	51.64	51
50000	311.5	76.57	388.07	33.96	48.45	48
100000	627	186.65	813.65	68	50.83	50

Table 4: Performance of Private Intersection-geometric mean protocol.

	Time(s)			Comm.[MB]	Actual Value	Output (rounded down)
Input Size	Offline	Online	Total			
1000	6.23	1.99	8.22	0.6799	37.84	37
2000	12.45	3.27	15.72	1.3589	38.07	38
3000	18.61	4.91	23.52	2.0379	36.71	36
4000	24.61	6.75	31.36	2.7169	36.58	36
5000	31.02	8.08	39.1	3.3958	42.288	42
10000	63.1	19.33	82.43	6.80	44.28	44
50000	312.8	77.07	389.87	33.96	43.04	43
100000	629	187.68	816.68	68	37.13	37

presented in Tables 3, 4 and illustrated in Figures 1 and 2, which highlight that both the running time and communication cost in our protocols are linear with respect to the input size.

From the tabulated results, we note that our *PSI-Stats* protocols have comparable running time and communication cost in relation to [35] which solely computes the intersection sum. In fact, our protocol which outputs both the sum and range functionality is observed to have a slightly lower running time compared to [35] which only outputs the sum. This is attributed to the reason that in the implementation of [35], an intended random shuffle is replaced with a sort based on hash digests of associated values. Due to the properties of hash functions, this does not affect the protocol of [35]. However, that incurs a slight difference in their overall running time which explains the discrepancy.

We also select a couple of actual datasets to conduct our experiments. The dataset [41] taken from the UCI ML repository [21] relates to marketing campaigns of a Portuguese banking institution. This dataset consists of a pair of sets: one of which is a proper subset of the other. We extract the attribute of interest corresponding to the yearly bank balance of bank's clients. A holds the smaller

Table 5: Performance of PSI-Stats on a UCI repository dataset.

	Time(s)			Comm.[MB]	Actual Mean	Output (rounded down)
Input Size	Offline	Online	Total			
45211	281.6	68.77	350.37	30.74	1422.65	1422

Table 6: Performance of PSI-Stats on a Kaggle dataset.

	Sum & Range		Mean		Actual Mean	Output Mean (rounded down)
Input Size	Total time(s)	Comm.[MB]	Total time(s)	Comm.[MB]		
30000	232	20.37	239	20.4	6317.035	6317

set of size 4521 while B holds the larger set of size 45211. To simulate a practical setting, each input (representing a client) is assigned a *kojin bangō* which is a unique 12-digit ID number issued to residents in Japan for taxation purpose. Other identifiers such as the Social Security Number issued in the United States can also be similarly assigned. The set size of A is then increased to 45211 to match that of B by generating a distinct *kojin bangō* for each new client. Consequently, the set size is 45211 for both parties and the intersection corresponds to the original smaller set of size 4521. Related figures in the computation of mean yearly balance of common clients between these two parties via *PSI-Stats* are presented in Table 7. The second dataset involves spending from a mall taken from Kaggle [6] which has an input size of 30000. We use the column labelled "payment 2" as the set of corresponding associated values. The performance of *PSI-Stats* of selected functionalities on the Kaggle dataset is recorded in Table 6.

9.1 Comparisons with circuit-based PSI protocols

The main advantage of *PSI-Stats* as compared to all existing circuit based approaches is that it incurs the lowest communication overhead. This is particular crucial in low bandwidth settings or where communication cost is at a premium. In this regard, we demonstrate this by providing a comparison of the running time of *PSI-Stats* with the state-of-the-art circuit based PSI protocol in low network bandwidth settings.

The current most efficient state-of-the-art circuit-based PSI protocol is the recent work of Pinkas *et al.* [47]. As such, we shall use [47] to serve as a benchmark in the comparisons of *PSI-Stats* with circuit-based PSI approaches. We run the "no stash" protocol of [47] along with the arithmetic mean protocol of *PSI-Stats* in a network bandwidth of 1 Mbps with a round-trip time of 0.02 ms. The results are reflected in Table 7. It should be emphasized that the "no stash" protocol of [47] outputs the set intersection (without payload) as compared to the arithmetic mean of the set intersection given in the running times of

PSI-Stats. Substantial modifications have to be incorporated to the "no stash" protocol to support secure post-processing of the output of the set intersection (e.g. statistical functions) which incur additional communication and computational overheads. Nevertheless, the results of [47] in Table 7 serve well as a lower bound reference for the output functionality of arithmetic mean. Moreover, to optimize the efficiency when running the protocol of [47], we compute in Matlab the minimal number of mega-bins B required such that each mega-bin contains at most $max_b \leq 1024$ elements with probability under 2^{-40} for various set sizes n. The probability that there exists a bin with at least max_b elements given in [47] is bounded above by

$$B \sum_{i=max_b}^{3n} B^{-i} \binom{3n}{i} \left(1 - \frac{1}{B}\right)^{3n-i}.$$

Our computed minimum values of B are 19, 38, 75, 113 for $n = 5000, 10000, 20000, 30000$ respectively.

Table 7: Comparisons with state-of-the-art circuit-based PSI at network bandwidth setting of 1 Mbps.

	Pinkas *et al.* [47]		*PSI-Stats*	
Input Size	Time(s)	Comm.[MB]	Time(s)	Comm.[MB]
5000	113.48	13.6	65.58	3.4
10000	222.60	26.3	135.56	6.8
20000	444.01	52.9	271.76	13.6
30000	666.23	79.2	407.89	20.4

From the experimental results of Table 7, *PSI-Stats* incurs a lower communication overhead by an average factor of 3.9× as compared to [47]. *PSI-Stats* also has a lower run time in a network bandwidth setting of 1 Mbps. Moreover, when the round-trip time is increased from 0.02 ms to 100 ms, the run time of [47] increases by 3.2 seconds and 3.84 seconds for set sizes of 2^{12} and 2^{16} respectively. In contrast, the run time increase for *PSI-Stats* is merely 0.3 seconds. Based on a report published by Akamai Technologies, Inc. [1], the average internet connection speed of Venezuela and Paraguay is 1.8 Mbps and 1.4 Mbps respectively. We also run experiments at these bandwidths for set size of 2^{16}. The run time of [47] and *PSI-Stats* is 808.90 seconds and 731.06 seconds respectively at 1.8 Mbps while the corresponding run time is 1038.85 seconds and 787.65 seconds respectively at 1.4 Mbps.

10 Conclusion

We present *PSI-Stats* which supports the secure computations of various statistical functions in a privacy-preserving manner. Tables 8 and 9 provide a summarized comparison of this paper with that of [35]. A tick denotes a party who learns the output of the specified statistical functions or intersection cardinality at the end of the protocols while a cross denotes otherwise. A dash denotes non-applicability.

Table 8: Functionality of Protocols

	Ion *et al.*'[35]		*PSI-Stats*	
	A	B	A	B
Sum	×	✓	–	–
Sum & Range	–	–	×	✓
Mean (inc. variance, s.d, etc.)	–	–	×	✓
Geometric Mean	–	–	×	✓
Harmonic Mean	–	–	×	✓
Percentile Sum	–	–	×	✓
Sum Composition Type 1 & 2	–	–	×	✓
Intersection Cardinality	✓	×	✓	×
Communication Rounds	3		3	

Table 9: Functionality of Reverse Protocols

	Ion *et al.*'[35]		*PSI-Stats*	
	A	B	A	B
Sum	✓	×	✓	×
Sum & Range	–	–	✓	×
Mean (inc. variance, s.d, etc.)	–	–	✓	×
Geometric Mean	–	–	✓	×
Harmonic Mean	–	–	✓	×
Percentile Sum	–	–	✓	×
Sum Composition Type 1 & 2	–	–	✓	×
Intersection Cardinality	✓	✓	✓	×
Communication Rounds	3		4	

As shown in Table 8, our work here excludes a protocol for the sole output of intersection sum as this was already presented in [35]. Instead, this paper presents protocols which enable various statistical output functionalities in a secure and privacy-preserving manner. With reference to Table 9, our *reverse protocols* enable the hiding of the intersection cardinality from the party who holds the associated values at the expense of an additional round of communication. This is achieved for all statistical functions discussed in this paper.

Our experimental results show that the running time and communication cost of our protocols in the secure computations of various generalized means are comparable to the efficiency of the recently deployed intersection-sum protocol of [35]. The benefit of *PSI-Stats* having a substantially lower communication cost compared to circuit based PSI approaches is desirable in many business applications. This is also relevant in environments of low network bandwidths.

We have noted from our experiments that the run time of all of our protocols is dominated by the time taken to perform encryption of each associated value. As such, *PSI-Stats* is highly parallelizable and the performance can be further enhanced when multi-threading is enabled. In addition, *PSI-Stats* can easily be extended to enable statistical outputs only if the size of the intersection set

exceeds a pre-defined threshold value. In instances where the intersection set of the identifiers between the two parties is null or under a specified threshold size, party A can call for an early abort of the protocol.

Acknowledgements

We thank Benny Pinkas, Thomas Schneider, Oleksandr Tkachenko and Avishay Yanai for kindly providing us with their codes of the implementation in [47].

References

1. Akamai Technologies, Inc., `www.akamai.com`
2. Anonos, `https://www.anonos.com/`
3. Baffle, `https://baffle.io/`
4. Data Republic, `https://www.datarepublic.com/`
5. IEEE 754-2019 - IEEE Standard for Floating-Point Arithmetic, `standards.ieee.org`
6. Kaggle, `https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset`
7. Privitar, `https://www.privitar.com/`
8. Sepior, `https://sepior.com/`
9. Sharemind, `https://sharemind.cyber.ee/`
10. Unbound Tech, `https://www.unboundtech.com/`
11. R. Agrawal, A. Evfimievski and R. Srikant. Information sharing across private databases. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 86-97, 2003.
12. P. Baldi, R. Baronio, E. D. Cristofaro, P. Gasti, and G. Tsudik. Countering GATTACA: Efficient and secure testing of fully-sequenced human genomes. In *ACM Conference on Computer and Communications Security*, pages 691-702, 2011.
13. E. Barker. Recommendation for key management part 1: General (revision 4). NIST special publication, 800(57):1-147, 2016.
14. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1-36, 2014.
15. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing*, 43(2):831-871, 2014.
16. H. Chen, Z. Huang, K. Laine, and P. Rindal. Labeled PSI from Fully Homomorphic Encryption with Malicious Security. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1223-1237, 2018.
17. H. Chen, K. Laine, and P. Rindal. Fast private set intersection from homomorphic encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1243-1255, 2017.
18. M. Ciampi and C. Orlandi. Combining private set-intersection with secure two-party computation. In *International Conference on Security and Cryptography for Networks*, pages 464-482, 2018.
19. E. D. Cristofaro and G. Tsudik. Practical private set intersection protocols with linear complexity. In *Financial Cryptography and Data Security (FC'10)*, vol. 6052, pages 143-159, 2010.

20. S. K. Debnath and R. Dutta. Secure and efficient private set intersection cardinality using bloom filter. In *International Conference on Information Security (ISC'15)*, pages 209-226, 2015.
21. D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017.
22. C. Dong, L. Chen, and Z. Wen. When private set intersection meets big data: An efficient and scalable protocol. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS'13)*, pages 789-800, 2013.
23. S. N. Durlauf and L. Blume, L. *The new Palgrave dictionary of economics (Vol. 6)*, 2008
24. R. Egert, M. Fischlin, D. Gens, S. Jacob, M. Senke, and J. Tillmanns. Privately computing set-union and set-intersection cardinality via bloom filters. In *Australasian Conference on Information Security and Privacy*, pages 413-430, 2015.
25. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469-472, 1985.
26. B. H. Falk, D. Noble, and R. Ostrovsky. Private set intersection with linear communication from general assumptions. In *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*, pages 14-25, 2019.
27. J. Fan and F. Vercauteren. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
28. M. J. Freedman, C. Hazay, K. Nissim, and B. Pinkas. Efficient set intersection with simulation-based security. *J. Cryptology*, 29(1):115-155, 2016.
29. M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology–EUROCRYPT'04*, vol. 3027, pages 1-19, 2004.
30. C. Gentry, S. Halevi, and N. P. Smart. Fully homomorphic encryption with polylog overhead. In *Annual International Conference on the Theory and Applications of Cryptographic Technique*, pages 465-482, 2012.
31. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218-229, 1987.
32. P. Hallgren, C. Orlandi, and A. Sabelfeld. PrivatePool: privacy-preserving ridesharing. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 276-291, 2017
33. Y. Huang, D. Evans, and J. Katz. Private set intersection: Are garbled circuits better than custom protocols? In *Network and Distributed System Security (NDSS'12)*, 2012.
34. B. A. Huberman, M. Franklin, and T. Hogg. Enhancing privacy and trust in electronic communities. In *ACM Conference on Electronic Commerce (EC'99)*, pages 78-86, 1999.
35. M. Ion, B. Kreuter, A. E. Nergiz, S. Patel, M. Raykova, S. Saxena, K. Seth, D. Shanahan, and M. Yung. On deploying secure computing commercially: Private intersection-sum protocols and their business applications. *IACR Cryptology ePrint Archive*, 2019:723, 2019. (To appear in *EuroS&P'20*)
36. Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. In *Advances in Cryptology–CRYPTO'03*, vol. 2729, pages 145-161, 2003.
37. D. Kales, C. Rechberger, T. Schneider, M. Senker, and C. Weinert. Mobile private contact discovery at scale. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1447-1464, 2019.
38. V. Kolesnikov and R. Kumaresan. Improved OT extension for transferring short secrets. In *Advances in Cryptology–CRYPTO'13 (2)*, vol. 8043, pages 54-70, 2013.

39. V. Kolesnikov, R. Kumaresan, M. Rosulek, and N. Trieu. Efficient batched oblivious PRF with applications to private set intersection. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS'16)*, pages 818-829, 2016.
40. C. Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *Symposium on Security and Privacy (S&P'86)*, pages 134-137, 1986.
41. S. Moro, R. Laureano, and P. Cortez. Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. In *Proceedings of the European Simulation and Modelling Conference - ESM'2011*, pages 117-121, 2011.
42. S. Nagaraja, P. Mittal, C. Hong, M. Caesar, and N. Borisov. BotGrep: Finding P2P bots with structured graph analysis. In *USENIX Security Symposium*, pages 95-110, 2010.
43. A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In *NDSS*, vol 11, 2011.
44. M. Orrù, E. Orsini, and P. Scholl. Actively secure 1-out-of-N OT extension with applications to private set intersection. In *Topics in Cryptology–CT-RSA'17*, vol. 10159, pages 381-396, 2017.
45. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology–EUROCRYPT'99*, pages 223-238, 1999.
46. B. Pinkas, T. Schneider, G. Segev, and M. Zohner. Phasing: Private set intersection using permutation-based hashing. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 515-530, 2015.
47. B. Pinkas, T. Schneider, O. Tkachenko, and A. Yanai. Efficient circuit-based PSI with linear communication. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'19)*, pages 122-153, 2019.
48. B. Pinkas, T. Schneider, C. Weinert, and U. Weider. Efficient circuit-based psi via cuckoo hashing. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'18)*, pages 125-157, 2018.
49. B. Pinkas, T. Schneider, and M. Zohner. Faster private set intersection based on OT extension. In *USENIX Security Symposium*, vol. 14, pages 797-812, 2014.
50. B. Pinkas, T. Schneider, and M. Zohner. Scalable private set intersection based on OT extension. *ACM Transactions on Privacy and Security (TOPS)*, 21(2):1-35, 2018.
51. A. C. D. Resende and D. F. Aranha. Faster unbalanced private set intersection. In *International Conference on Financial Cryptography and Data Security*, pages 203-221, 2018.
52. P. Rindal and M. Rosulek. Improved private set intersection against malicious adversaries. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'17)*, pages 235-259, 2017.
53. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120-126, 1978.
54. J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6(1):64-94, 1962.
55. T. Schneider and M. Zohner. GMW vs. Yao? Efficient secure two-party computation with low depth circuits. In *International Conference on Financial Cryptography and Data Security*, pages 275-292, 2013.
56. A. Shamir. On the power of commutativity in cryptography. In *International Colloquium on Automata, Languages and Programming (ICALP'80)*, vol 85, pages 582-595, 1980.

57. N. Trieu, K. Shehata, P. Saxena, R. Shokri, and D. Song. Epione: Lightweight contact tracing with strong privacy. *arXiv preprint arXiv:2004.13293*, 2020.
58. B. Willemsen. Hype Cycle for Privacy. Gartner, 2019.
59. A.C.C Yao. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162-167, 1986.

A Semi-honest Security Model

We provide a formal description of the semi-honest security model and the corresponding simulation based security proof of *PSI-Stats*. The simulation based proof of security for our arithmetic mean output functionality is given in Theorem 7. Our proof is similar to that provided by [11, 35]. The result for other generalized means can be similarly derived from Theorem 7.

Semi-honest security. Let $\texttt{view}_A{}^{\Pi}(X, Y)$ and $\texttt{view}_B{}^{\Pi}(X, Y)$ be the view of A and B in the protocol Π, respectively. The protocol Π is secure in the semi-honest model if there exists probabilistic polynomial time (PPT) simulators $\texttt{SIM}_A$ and $\texttt{SIM}_B$ such that for all inputs X, Y,

$$\texttt{view}_A^{\Pi}(X, Y) \approx \texttt{SIM}_A(1^{\lambda}, X, n, \texttt{output});$$

$$\texttt{view}_B^{\Pi}(X, Y) \approx \texttt{SIM}_B(1^{\lambda}, Y, m, \texttt{output}).$$

Theorem 7. *There exist PPT simulators $\texttt{SIM}_A$ and $\texttt{SIM}_B$ such that for all security parameters λ,*

$$\texttt{view}_A^{\Pi}\left(\{a_i\}_{i=1}^m, \{(b_i, t_i)\}_{i=1}^n\right) \approx \texttt{SIM}_A(1^{\lambda}, \{a_i\}_{i=1}^m, n, k);$$

$$\texttt{view}_B^{\Pi}\left(\{a_i\}_{i=1}^m, \{(b_i, t_i)\}_{i=1}^n\right) \approx \texttt{SIM}_B(1^{\lambda}, \{(b_i, t_i)\}_{i=1}^n, m, M').$$

Proof. The simulator for A is constructed in the following steps.

1. Generate a key $k_1 \in G$ and public, private key pairs for the Paillier encryption scheme.

2. Honestly generate and send $\{h(a_i)^{k_1}\}_{i=1}^m$ as message of A in the first communication round.

3. Generate a dummy set $D = \{g_i\}_{i=1}^m$, where each g_i is randomly selected from G. Send $\{g_i^{k_1}\}_{i=1}^m$ as message of B in Step 3 of Protocol 1.

4. Generate a dummy set $D' = \{h_j\}_{j=1}^n$ for B by setting $h_j = g_j$ for $j \in [k]$, and each h_j for $j \in \{k, \ldots, n\}$ is randomly selected from G.

5. Send $\{(h_j, E(0))\}_{j=1}^n$ in shuffled order as the message of B in Step 3 of Protocol 1, such that each $E(0)$ is freshly generated.

6. Honestly generate the message of A in Step 4 of Protocol 1 using the dummy messages of B from the previous step.

7.Output the view of A in this simulated execution.

By applying a multi-step hybrid argument,

$$\texttt{view}_A^{\Pi}\left(\{a_i\}_{i=1}^m, \{(b_i, t_i)\}_{i=1}^n\right) \approx \texttt{SIM}_A(1^{\lambda}, \{a_i\}_{i=1}^m, n, k).$$

Define $\texttt{SIM}_B$ to perform the Setup step honestly as well honestly performing the operations corresponding to B. $\texttt{SIM}_B$ simulates the messages sent by A in the following manner. $\texttt{SIM}_B$ sends m random chosen elements of G instead of $\{h(a_i)^{k_1}\}_{i=1}^m$ in Step 2 of Protocol 1. In Step 4 of Protocol 1, $\texttt{SIM}_B$ sends r and a fresh Paillier encryption of $\lfloor rM' \rceil$ where r is a randomly chosen 1024-bit value. By a hybrid argument,

$$\texttt{view}_B^{\Pi}\left(\{a_i\}_{i=1}^m, \{(b_i, t_i)\}_{i=1}^n\right) \approx \texttt{SIM}_B(1^\lambda, \{(b_i, t_i)\}_{i=1}^n, m, M').$$

□