PRIVATE SET INTERSECTION: NEW GENERIC CONSTRUCTIONS AND FEASIBILITY RESULTS

PAOLO D’ARCO
Dipartimento di Informatica, University of Salerno
84084 Fisciano (SA), Italy

MARÍA ISABEL GONZÁLEZ VASCO* AND ANGEL L. PÉREZ DEL POZO
MACIMTE, Area de Matemática Aplicada, U. Rey Juan Carlos
c/ Tulipán, s/n, 28933, Móstoles, Madrid, Spain

CLAUDIO SORIENTE
Telefónica Research
Barcelona, Spain

RAINER STEINWANDT
Florida Atlantic University (FAU), 777 Glades Rd
Boca Raton, FL 33431, USA

(Communicated by Alfred Menezes)

Abstract. In this paper we focus on protocols for private set intersection (PSI), through which two parties, each holding a set of inputs drawn from a ground set, jointly compute the intersection of their sets. Ideally, no further information than which elements are actually shared is compromised to the other party, yet the input set sizes are often considered as admissible leakage.

In the unconditional setting we evidence that PSI is impossible to realize and that unconditionally secure size-hiding PSI is possible assuming a set-up authority is present in an set up phase. In the computational setting we give a generic construction using smooth projective hash functions for languages derived from perfectly-binding commitments. Further, we give two size-hiding constructions: the first one is theoretical and evidences the equivalence between PSI, oblivious transfer and the secure computation of the AND function. The second one is a twist on the oblivious polynomial evaluation construction of Freedman et al. from EUROCRYPT 2004. We further sketch a generalization of the latter using algebraic-geometric techniques. Finally, assuming again there is a set-up authority (yet not necessarily trusted) we present very simple and efficient constructions that only hide the size of the client’s set.

2010 Mathematics Subject Classification: Primary: 94A60; Secondary: 68P99.

Key words and phrases: Private set intersection, size-hiding, unconditional security, homomorphic encryption, oblivious pseudorandom evaluation.

The first three and the last author were partially supported by the Spanish Ministerio de Economía y Competitividad through the project grant MTM-2010-15167. This research is also partially supported by the Italian PRIN project GenData 2020.

* Corresponding author.
1. Introduction

The Private Set Intersection (PSI) problem is, in a nutshell, concerned with two parties, each holding a set of inputs drawn from a ground set, that wish to jointly compute the intersection of their sets, without leaking any additional information. Cryptographic solutions to the PSI problem allow interaction between a client $C$ and a server $S$, with respective private input sets $C = \{c_1, \ldots, c_v\}$ and $S = \{s_1, \ldots, s_w\}$, both drawn from the ground set $U$. At the end of the interaction, $C$ learns $S \cap C$ and $|S|$, while $S$ learns nothing beyond $|C|$.

Plenty of potential applications for PSI have been suggested. The Department of Homeland Security wishing to check a list of suspicious persons against the passenger list of a flight operated by a foreign air carrier, the federal tax authority checking if any suspect tax evader has a foreign bank account, and other folklore case scenarios have been the leading examples in many papers (see [13] for a detailed list). Remarkably, in many of these scenarios it is in the interest of the client (and maybe also of the server), that the size of its input set is not leaked through the interaction. Consider for instance the case in which two competing companies wish to find out the set of common clients in a certain geographical area, while they might rather not disclose to what extent they are already established there. If size-hiding is a requirement, many of the techniques proposed for private set intersection fail to do the job.

History of PSI: Results and Techniques. Freedman et al. [20] introduced the first PSI protocol based on oblivious polynomial evaluation (OPE). The key intuition is that elements in the client’s private set can be represented as roots of a polynomial, i.e., $P(x) = \prod_{i=1}^{v}(x - c_i) = \sum_{i=0}^{v} a_i x^i$. Leveraging any additively homomorphic encryption scheme (e.g., [37]), the encrypted polynomial is obliviously evaluated by $S$ on each element of its data set: $S$ computes

$$\{u_j\}_{j=1, \ldots, w} = \{E(r_j P(s_j) + s_j)\}_{j=1, \ldots, w}$$

where $E()$ denotes additively homomorphic encryption and $r_j$ is chosen at random. If $s_j \in C \cap S$, then $C$ learns $s_j$ upon decryption of the corresponding ciphertext $u_j$; otherwise $C$ learns a random value. OPE-based PSI protocols have been extended in [31, 21, 14, 15] to support multiple parties and other set operations such as union, element reduction, etc. Hazay et al. [24] proposed an Oblivious Pseudo-Random Function (OPRF) [19] as alternative primitive to achieve PSI. In [25], given a secret index $k$ to a pseudo-random function family, $S$ evaluates

$$\{u_j\}_{j=1, \ldots, w} = \{f_k(s_j)\}_{j=1, \ldots, w}$$

and sends it to $C$. Later, $C$ and $S$ engage in $v$ executions of the OPRF protocol where $C$ is the receiver with private input $C$ and $S$ is the sender with private input $k$. As a result, $C$ learns $\{f_k(c_i)\}_{i=1, \ldots, v}$ such that $c_i \in C \cap S$ if and only if $f_k(c_i) \in \{u_j\}_{j=1, \ldots, w}$. Improvements using the same approach were provided in [27, 28].

Given $U$ as the ground set where elements of $C$ and $S$ are drawn (i.e., $C, S \subseteq U$), none of the above techniques prevents a client to run a PSI protocol on private input $C = U$ in order to learn the elements in $S$. To this end, Camenisch et al. extended PSI to Certified Sets [5], where a Trusted Third Party (TTP) ensures that private inputs are valid and binds them to each participant. The certification issue was also addressed in [11], where a related problem to PSI was considered. Moreover, the same extension, under a different name, Authorized PSI, was considered in
where protocols that use modular exponentiation, multiplication and hash evaluation were described.

Efficiency of the protocols on large data sets is an important practical issue, and it has been addressed in several papers in the last years. One of the most relevant applications in this respect comes from genomic computing: the user wants to protect the privacy of sensitive information coded in her genomic sequence but at the same time wishes to engage in private computations with other parties, in order to get some advantage, e.g., understand whether she has a predisposition to certain diseases or whether some medication could benefit her state of health. Baldi et al. [4] and De Cristofaro et al. [10] offer an in-depth introduction to this area. In [12], linear-complexity private set intersection protocols were proposed for malicious adversaries. Along the line of [36], to gain efficiency, Bloom filters have been applied in [30, 17]. The protocols proposed in [30] are elegant and one of them is designed for an outsourced scenario. The solutions described in [17, 30] and [38] are currently, with respect to semi-honest adversaries, the most efficient ones available.

Related work. All of the above techniques reveal the size of the participants’ sets. That is, C (resp. S) learns |S| (resp. |C|), even if C ∩ S = ∅. To protect the size of private input sets, Ateniese et al. [3] proposed a size-hiding PSI protocol where C can privately learn C ∩ S and |S| without leaking the size of C. Their scheme is based on RSA accumulators and unpredicatability of the RSA function. The authors proved security against honest-but-curious adversaries in the random oracle model. Very recently, general results on hiding the input-size in two party computation were given [32]; Lindell et al. prove that hiding the input size of both participants involved in a set intersection protocol is impossible in the cryptographic model considering honest-but-curious adversaries, if after the interaction:

- both C and S get C ∩ S and nothing else,
- client only gets C ∩ S, server gets nothing
- client gets C ∩ S, server gets its size, |C ∩ S|.

Further, they are able to evidence that hiding one party’s input size is possible using fully homomorphic encryption; this construction is mainly theoretical, but is the first to encourage searching for new designs without random oracles.

Another (loosely related) work in which the input-size issue has been addressed is [6], where the authors present a simple database commitment functionality where besides the standard security properties the input size of the committed database set is not revealed. They consider malicious users and use as a main tool universal arguments of quasi-knowledge.

Contributions. This paper focuses on PSI protocols between honest-but-curious parties under different security models. It is an improved and revised version of [16]: as the most remarkable changes, we have refined definitions in Section 2, extended the results of Section 3, and added the generic constructions from Section 4. Section 5 has been rewritten, excluding constructions that are outperformed by the ones depicted now in the paper.

We start by considering the size-hiding scenario in Section 3, where both parties hide the size of their input sets. In the unconditional setting, we prove that PSI protocols are not achievable, while size-hiding PSI is possible involving a trusted

---

\[1\] Or computational model, as it is also usually named.
third party which disappears after the setup phase (see Figure 1). Then, in Section 4 we deal with computational security. First we present a new generic construction for PSI from smooth projective hashing, standing on the seminal ideas of Cramer-Shoup [9] and Gennaro-Lindell [22] which have already found applications in several other contexts, such as password-based authenticated key exchange and oblivious transfer. From this generic construction one can derive PSI protocols that can be proved secure in the common reference string model under different number theoretic assumptions, including decisional Diffie-Hellman, quadratic residuosity and Paillier’s decision composite residuosity. The communication complexity of the resulting schemes is quadratic in the input size, thus unfortunately higher than that of other known public-key constructions (like the OPE from [20]).

Further, we establish the existence of a PSI protocol where both parties hide the size of their sets (Figure 3); this does not contradict [32], for our protocol cannot be used unless the universe size \( U \) is polynomial (as computations have to actually run over the full universe). Then, we depict a generalization of the oblivious polynomial evaluation construction from Freedman et al. [20], which hides the input sizes if a bound \( M \) on both the client and server’s set is known, and sketch a generic algebraic-geometric construction that can be seen as an (at this point, only theoretical) generalization of it. The latter poses an algebraic-geometric question that we believe to be of interest in its own right.

Finally, in Section 5 we provide two explicit constructions of unbalanced protocols where only the client hides the size of its input set. These assume the aid of an authority (not necessarily trusted in one of them) which is only present in a setup phase and then disappears.

2. Preliminaries

In this section we provide definitions and tools used in the rest of the paper, essentially following the model from [3]. However, we refine their definitions slightly in order to deal with both computationally and unconditionally secure protocols, and to introduce the size-hiding constraint on both the client and the server side. Moreover, we generalize the underlying scenario by considering a finite set of clients \( \mathbf{C} = \{C_1, \ldots, C_n\} \) and a finite set of servers \( \mathbf{S} = \{S_1, \ldots, S_k\} \) (possibly not disjoint) holding corresponding data sets \( C_i \) for \( i = 1, \ldots, n \) and \( S_j, j = 1, \ldots, k \) whose elements are drawn from a common universe \( U \). Each client \( C_i \) may interact with each server \( S_j \) in order to compute the intersection \( C_i \cap S_j \). Note that as the set of clients and the set of servers may not be disjoint, we capture the situation in which the same participants may be enrolled in various executions playing a different role each time.

In some cases we introduce a setup authority \( \mathbf{SA} \) which is fully trusted and interacts with the parties in a setup phase before the actual protocol execution takes place (so, it is in a sense a Trusted Initializer as introduced by Rivest in [31]). This authority might provide secret information to the parties, and its presence is actually unavoidable if participants cannot be assumed honest-but-curious, to certify the sets of secrets held by the parties in order to prevent any party to execute the protocol with a different set than the one it owns (for instance, the full universe).

**Definition 1.** A party is referred to as honest-but-curious, HBC for short, if it correctly follows the steps of the protocol but eventually tries to get extra-knowledge from the transcript of the execution.
A size-hiding private set intersection protocol (SH-PSI), enabling two participants $C_i \in C = \{C_1, \ldots, C_n\}$ and $S_j \in S = \{S_1, \ldots, S_k\}$ to compute the intersection of their set of inputs, while revealing no further information (in particular, keeping secret all sizes of the involved sets) can be defined as follows:

**Definition 2.** A SH-PSI protocol is a scheme involving $C_i \in C = \{C_1, \ldots, C_n\}$, $S_j \in S = \{S_1, \ldots, S_k\}$ and (possibly) a fully trusted setup authority $SA$, together with two components, Setup and Interaction, where

- Setup is an algorithm that selects all global parameters, run by the server $S_j$ and the client $C_i$, and possibly involving the setup authority $SA$;
- Interaction is a protocol involving only $S_j$ and $C_i$ on respective input sets

$$S_j = \{s_{j1}, \ldots, s_{jw_j}\} \text{ and } C_i = \{c_{i1}, \ldots, c_{iv_i}\},$$

which are subsets of a ground set $U = \{u_1, \ldots, u_{|U|}\}$,

satisfying correctness, client privacy, and server privacy.

Correctness is formalized by:

**Definition 3.** A scheme specified in Definition 2 is **correct** if, when run by HBC parties, $C \in C$ and $S \in S$, at the end of Interaction, run on corresponding inputs $S$ and $C$, with overwhelming probability $S$ outputs $\bot$ and $C$ outputs $S \cap C$ or $\bot$ if the intersection is empty.

Notice that, compared to the definition of correctness provided in [3], we do not require that $|S|$ is part of the client’s output. In Section 5 in which we will apply the size-hiding restriction only on the client’s side, we will stick to the definition of correctness from [3] and require the client $C$ to output $S \cap C$ and $|S|$ or just $|S|$ if the intersection is empty. In the sequel, we will refer to such protocols as **unbalanced size-hiding**, thus coining the term USH-PSI to address schemes fulfilling the definition of correctness from [3]. Unless otherwise specified, the definition of correctness used in the sequel is the one above.

Concerning client privacy, since the server does not get any output from the protocol, it is enough to require that the server, from the interaction, does not distinguish between cases in which the client has different input sets.

**Definition 4.** Let $\text{View}_S(C, S)$ be a random variable representing the view of $S$ during the execution of Interaction with inputs $C$ (from a client in $C$) and $S$. A scheme specified as in Definition 2 guarantees **client privacy** if, for every $S^*$ that plays the role of $S$, for every set $S$, and for any two possible client input sets $C, \hat{C}$ it holds that:

$$\text{View}_{S^*}(C, S) \equiv \text{View}_{S^*}(\hat{C}, S).$$

Notice that in the above definition, when considering the unconditional setting, the parties $S, C$ and $S^*$ are unbounded and indistinguishable means that the two views are **perfectly indistinguishable**, that is, they are identically distributed. On the other hand, in the computational setting, $S, C$ and $S^*$ are probabilistic polynomial time Turing machines. Hence, indistinguishable means that the two views are **computationally indistinguishable**.

Server privacy needs a bit more: the client gets the output of the protocol, and by using its input and the output, by analyzing the transcript of the execution, could
obtain extra-knowledge about the server’s secrets. Nevertheless, if the transcript can be simulated by using only input and output, then server privacy is achieved.

**Definition 5.** Let \( C \in \mathcal{C} \) and View\(_C(C,S)\) be a random variable representing \( C\)'s view during the execution of Interaction with inputs \( C \) and \( S \). Then, a scheme defined as in Definition 2 guarantees server privacy if, for any \( C \in \mathcal{C} \) there exists an algorithm \( C^* \) such that

\[
\{C^*(C, S \cap C)\}_{(C,S)} \equiv \{\text{View}_C(C, S)\}_{(C,S)}.
\]

As before, in the unconditional setting the parties are unbounded and the transcript produced by \( C^* \) and the real view need to be identically distributed. In the computational setting, the parties are probabilistic polynomial time Turing machines and the transcript produced by \( C^* \) and the real view are required to be computationally indistinguishable. We will always consider HBC parties, even though Definitions 4 and 5 are written without this restriction.

3. SH-PSI: the unconditionally secure case

In this section we deal with the unconditionally secure setting and show two results: first, we evidence that unconditionally secure 2-party PSI protocols do not exist. Further, we prove that unconditionally secure size-hiding PSI can indeed be achieved if we allow the involvement of SA in the Setup algorithm.

3.1. Impossibility of unconditionally secure 2-party PSI. For a universe \( \mathcal{U} \) such that \(|\mathcal{U}| > 2\), there is no unconditionally secure 2-party PSI protocol, no matter if one or both players receive the intersection as an output, or if the size of the input sets is revealed. To prove this claim, we first show that from a secure PSI protocol \( \Pi \) that outputs the intersection and the set sizes to both participants, one can construct a protocol that securely computes the AND function. The latter does not exist in the unconditional setting [8, Page 22], so \( \Pi \) cannot exist. The remaining output-variants of PSI protocols can be obtained from a protocol like \( \Pi \), thus completing our argument.

**Theorem 1.** Let \( F = (F_1, F_2) \) be the functionality that, on input a pair of subsets \((C,S)\) of a universe \( \mathcal{U} \) with \(|\mathcal{U}| > 2\), outputs \( F_1(C,S) = (C \cap S, |S|) \) to the first participant and \( F_2(C,S) = (C \cap S, |C|) \) to the second one. Then an unconditionally secure 2-party protocol which computes \( F \) does not exist.

**Proof.** Let us reason by contradiction and assume that a secure 2-party protocol \( \Pi \) which computes \( F \) does exist. From \( \Pi \) we will build a secure private AND protocol \( \Gamma \). A private AND protocol is a two-party protocol, run by users \( A \) and \( B \), at the end of which the players get the logical AND of their input bits and nothing else (i.e., a secure protocol for computing \( a \cdot b \) from private inputs \( a, b \in \{0,1\} \)).

Indeed, assume that, for \( i \in \{1,2\} \), \( P_i \) is holding as input for \( \Gamma \) a bit \( b_i \in \{0,1\} \). For the PSI protocol \( \Pi \), choose three different elements \( a_1, a_2, a_3 \in \mathcal{U} \). Each player \( P_i \) constructs its input set \( X_i \) as follows:

- If \( b_i = 0 \), then \( X_i \) is chosen as \( \{a_i\} \).
- If \( b_i = 1 \), then \( X_i \) is chosen as \( \{a_3\} \).

Then \( P_1 \) and \( P_2 \) run the protocol \( \Pi \) for PSI with inputs \( X_1 \) and \( X_2 \) respectively. Each player, depending on the output of \( \Pi \), sets the output of \( \Gamma \) as follows:

- If the intersection \( X_1 \cap X_2 = \emptyset \), then the output of \( \Gamma \) is set to 0.
- If the intersection \( X_1 \cap X_2 = \{a_3\} \), then the output of \( \Gamma \) is set to 1.
It is easy to check that $\Gamma$ is a secure protocol for the computation of $\text{AND}(b_1, b_2)$:

**Correctness.** Note that $X_1 \cap X_2 = \{a_3\}$ if and only if $b_1 = b_2 = 1$ and $X_1 \cap X_2 = \emptyset$ otherwise, thus $\Gamma$ correctly computes $\text{AND}(b_1, b_2)$.

**Security.** First note that all the possibly involved sets have precisely one element, therefore the size of the other player’s set does not provide any information. Let us analyze the information leaked from $X_1 \cap X_2$ to player $P_i$. If the input of player $P_i$ is $b_i = 0$ then $X_1 \cap X_2 = \emptyset$, regardless of the input of the other player. Thus $P_i$ does not gain any information. On the other hand, if $b_i = 1$, the intersection $X_1 \cap X_2$ allows $P_i$ to learn the input of the other player, but this is also what happens in an ideal execution of an AND protocol.

As $\Pi$ is supposed to be an unconditionally secure protocol, players do not learn anything beyond the output, so $\Gamma$ is also an unconditionally secure protocol, which concludes the proof.

**Corollary 1.** Let $G = (G_1, G_2)$ be any functionality that, on input a pair of subsets $(C, S)$ of a universe $U$ with $|U| > 2$, outputs $G(C, S)$ such that

- $G_i$ consists of both, one, or none of the components of $F_i$ for $i \in \{1, 2\}$, where $F$ is the functionality described in Theorem 1.
- $C \cap S$ is present in at least one of the two outputs $G_1(C, S)$ or $G_2(C, S)$.

Then an unconditionally secure 2-party protocol which computes $G$ does not exist.

**Proof.** From a protocol $\Lambda$ which securely computes $G$ we can derive a protocol $\Pi$ which securely computes $F$. Namely, the player holding the intersection $C \cap S$ sends it to the other player, if needed, and each player sends the size of its input to the other player. Theorem 1 states no such protocol $\Pi$ can exist; as a result, neither can $\Lambda$.

### 3.2. Unconditionally secure SH-PSI with a trusted initializer

Allowing the involvement of a setup authority, we can achieve size-hiding in the unconditional setting. Figure 1 shows a construction. In the Setup, the $SA$ chooses two random bijections $f, g : \mathcal{P}(U) \rightarrow \{0, 1\}^{|U|}$, which are kept secret from client and server. After the $SA$ has received the client’s set of secrets, she sends the client an identifier, computed by means of $f$, and a list of sub-identifiers, one for each possible subset of the client’s set of secrets, computed through the second random function $g$. When the $SA$ receives a server’s set of secrets, she constructs a two-column table: in the first column there is, for each possible subset $E$ of the ground set, an identifier of $E$, computed with the first random function; the second column has an identifier, computed with the second random function, of the intersection $E \cap S$ of the subset $E$ and the server’s set of secrets $S$. The table is given to the server. The interaction between a client and a server is a simple two-round protocol: the client sends its set identifier; the server looks up in the table the row with the received identifier, and returns the identifier in the second column. Finally, the client looks up the server’s reply in the list of sub-identifiers and determines the intersection. We emphasize that the goal of this construction is only to show the existence of SH-PSI schemes in the unconditional setting. The scheme we provide is not practical for a universe of large size.

**Theorem 2.** Let $f, g : \mathcal{P}(U) \rightarrow \{0, 1\}^{|U|}$ be two random bijections. The protocol described in Figure 1 is a SHI-PSI protocol achieving correctness, client privacy, and server privacy in the unconditional model.

---

Underline:

**PSI: constructions and feasibility results**

487

---

**Corollary 1.** Let $G = (G_1, G_2)$ be any functionality that, on input a pair of subsets $(C, S)$ of a universe $U$ with $|U| > 2$, outputs $G(C, S)$ such that

- $G_i$ consists of both, one, or none of the components of $F_i$ for $i \in \{1, 2\}$, where $F$ is the functionality described in Theorem 1.
- $C \cap S$ is present in at least one of the two outputs $G_1(C, S)$ or $G_2(C, S)$.

Then an unconditionally secure 2-party protocol which computes $G$ does not exist.

**Proof.** From a protocol $\Lambda$ which securely computes $G$ we can derive a protocol $\Pi$ which securely computes $F$. Namely, the player holding the intersection $C \cap S$ sends it to the other player, if needed, and each player sends the size of its input to the other player. Theorem 1 states no such protocol $\Pi$ can exist; as a result, neither can $\Lambda$.

---

**Theorem 2.** Let $f, g : \mathcal{P}(U) \rightarrow \{0, 1\}^{|U|}$ be two random bijections. The protocol described in Figure 1 is a SHI-PSI protocol achieving correctness, client privacy, and server privacy in the unconditional model.
Figure 1. An unconditionally secure size-hiding set intersection protocol.

Proof. Correctness. The table $T$ contains the pair $(R, g(C \cap S))$, so the client will indeed retrieve $C \cap S$ by looking up in $L$.

Server Privacy. The client only gets a (random) identifier from the interaction, which can be simulated without $S$. Namely, the algorithm $C^*$ mentioned in Definition 2.5. may simply generated values uniformly at random in $\{0, 1\}^{|U|}$ to (perfectly) simulate the clients view.

Client Privacy. The server does not get any information about the correspondence value-subset, since the construction of the table is completely blind to it. Moreover, independently of the client set of secrets, the table the server gets has in the second column exactly $2^{|S|}$ different random values, that is, the number of all possible subsets of $S$. Each of these values appears exactly the same number of times, namely $2^{|U|-|S|}$, as for every $F \subseteq S$,

$$\# \{ E \subseteq U : S \cap E = F \} = \# \{ F \cup E' : E' \subseteq U \setminus S \} = \# \{ E' : E' \subseteq U \setminus S \} = 2^{|U|-|S|}.$$ 

Hence, a request from a client only allows the server to learn the two values $(f(C), g(C \cap S))$ which do not leak any information about the client’s set of secrets nor its size.

4. Cryptographic setting

In this section we move to the cryptographic setting; we start by introducing a new generic construction based on smooth projective hashing.

4.1. PSI from smooth projective hashing. The main tool behind our construction are smooth projective hash functions (SPH) from commitment schemes, standing on the seminal ideas of Cramer–Shoup [9] and Gennaro–Lindell [22], which have already found applications in several other contexts, such as password-based authenticated key exchange and oblivious transfer. We summarize informally the
main definitions and results, and refer to [22, 1] for precise definitions and concrete constructions.

Let \( \text{Com} \) be a non-interactive commitment scheme that is computationally hiding and perfectly binding in the common reference string model. This may be derived from any IND-CPA encryption scheme, like ElGamal encryption. For other applications of smooth projective hashing, non-malleability of the commitments is required, and they are typically derived from IND-CCA encryption schemes (cf. [1] for results on non-interactive commitment schemes for smooth projective hashing constructions). By \( \text{Com}_\rho(u, r) \) we denote a commitment to an element \( u \) using the common reference string \( \rho \) and the random coins \( r \); further, denote by \( \text{Com}_\rho \) the space of all strings that may be output by \( \text{Com} \) when the common reference string is \( \rho \).

Now, define the sets:

\[
\begin{align*}
U_\rho &= U \times \text{Com}_\rho \\
L_\rho &= \{(u, \text{com}) \mid \exists r \text{ such that } \text{com} = \text{Com}_\rho(u, r)\}
\end{align*}
\]

We consider a subset membership problem defined as follows. For each \( \ell \in \mathbb{N} \) a common reference string \( \rho \) (of polynomial size in \( \ell \)) is selected. Further, for each \( u \in U \) define \( D(U_\rho \setminus L_\rho) \) respectively \( D(L_\rho) \) as the distribution induced by choosing random \( r \) and computing \( (C_\rho(0|u, r), u) \) respectively \( (C_\rho(u, r), u) \).

Now consider the above subset membership problem \( I \), where for each \( \rho \) the set \( U_\rho \) is partitioned by the sets \( \{(u) \times \text{Com}_\rho\}_{u \in U} \). Then similarly as in [22], from the hiding property of the commitment scheme it is easy to see that \( I \) is a hard partitioned subset membership problem.

In such a situation, consider at hand a smooth projective hash family \( F \) for the induced hard partitioned subset membership problem. This essentially means we have a hash family \( F \) indexed by a key space \( K \), that is \( F = \{f_k\}_{k \in K} \), so that for every \( k \in K \), we have

\[
f_k : U_\rho \rightarrow G
\]

for a set \( G \) of superpolynomial size in the security parameter. Let \( P \) be a fixed set and \( \alpha \) an (efficiently computable) projection function defined over \( K \times \text{Com}_\rho \) with range in \( P \). Now given all these ingredients we get the following:

1. **Efficient hashing from \( k \):** there exists an efficient algorithm \( \text{KeyHash} \) which on input a pair \( (u, \text{com}) \) from \( U_\rho \), and a key \( k \in K \) outputs \( f_k(u, \text{com}) \).
2. **Efficient hashing from projection and witness:** there exists an efficient algorithm \( \text{ProjHash} \) which on input:
   - a pair \( (u, \text{com}) \in U_\rho \),
   - a random value \( r \) so that \( \text{com} = \text{Com}_\rho(u, r) \),
   - and a projection \( \alpha(k, \text{com}) \),
outputs \( f_k(u, \text{com}) \).
3. **Smoothness:** for every \( u \in U \), \( \text{com} \in \text{Com}_\rho \), with \( (u, \text{com}) \notin L_\rho \), the distributions
   \[
   \{\text{com}, u, \alpha(k, \text{com}), f_k(u, \text{com})\}
   \]
   and
   \[
   \{\text{com} = \text{Com}_\rho(u, r), u, \alpha(k, \text{com}), g\}
   \]
are computationally indistinguishable, where \( g \) is drawn uniformly at random from \( G \).

Let us now introduce our protocol, building on a non-interactive commitment scheme \( \text{Com} \) and a smooth projective hash family \( F = \{f_k\}_{k \in K} \) as described above. The key ideas in our design are, in a nutshell:
During setup, SA fixes and publishes the necessary information for implementing a non-interactive commitment scheme (which can actually be seen as the common reference string $\rho$). Also the associated projective hash family is made explicit.\footnote{The setup authority SA publishes all necessary information for implementing the commitment scheme as well as the description of the projective hash family. In a way, we can view its role as implementing the common reference string.}

In the first round, the client sends to the server a commitment $\text{com}_i$ to each element $c_i$ in his set.

The server selects uniformly at random a key $k \in K$ and evaluates the corresponding hash function $f_k$ on each pair $(s_j, \text{com}_i)$, for $1 \leq i \leq v, 1 \leq j \leq w$. Further, she sends them to the client together with the projection keys $\alpha(k, \text{com}_i)$ for $i = 1, \ldots, v$.

Finally, the client can identify matching sets $C \setminus \{s\}$ for each $s \in C \cap S$, and, as a result, retrieve the intersection. Note that this is all he learns, as due to the smoothness property he gains no information from the evaluations of $f_k$ in any pair $(u, \text{com})$ for which $\text{com}$ is not a valid commitment on $u$.

A detailed description can be seen in Figure 2; note that evaluations of $f_k$ computed by the server are actually executions of $\text{KeyHash}$, while evaluations by the client are executions of $\text{ProjHash}$.

**Theorem 3.** Let $\text{Com}$ be a non-interactive perfectly binding and computationally hiding commitment scheme, and $F$ a family of smooth projective hash functions. Then, the protocol given in Figure 2 is a PSI protocol achieving correctness, client privacy, and server privacy.
Proof. Correctness. To see that the protocol is correct, let us first argue that if there are indices $i \in \{1, \ldots, v\}$ and $j \in \{1, \ldots, w\}$ such that $c_i = s_j$, the client will set $c_i \in C \cap S$. Indeed, in this case $(s_j, \text{com}_i) \in L_\rho$, and as a result the KeyHash and ProjHash executions carried over by the server and client (respectively) will output the same value $\phi_{ij}$.

Furthermore, if $c_i$ is not in $S$, for each $j = 1, \ldots, w$, due to the smoothness of $F$ we have that with overwhelming probability $\text{test}_i \neq \phi_{ij}$; and as a result there is only negligible probability that the client includes $c_i$ in the intersection.

Client Privacy. As the commitment scheme is perfectly hiding, only $|C|$ is leaked from the interaction.

Server Privacy. To simulate the client’s view, $C^*$ uniformly at random selects a key $k^*$ for the projective hash family $F$, chooses nonces $r_i^*$, sets $\text{com}_i = \text{Com}_\rho(c_i, r_i^*)$, and computes the projections $\alpha(k^*, \text{com}_i)$ for $i = 1, \ldots, w$. Assume w.l.o.g. that $C \cap S = \{c_1, \ldots, c_I\}$, for $0 \leq I \leq v$. Now, for $i = 1, \ldots, I$ set $\phi_{i,j}^* = f_{k^*}(c_i, \text{com}_i)$ and choose all other $\phi_{i,j}$ uniformly at random from $G$. Due to the smoothness property, this view is computationally indistinguishable from the real client’s view.

Remark 1. As efficient smooth projective hashing as above can be derived from different number theoretical assumptions (decisional Diffie-Hellman, quadratic residuosity, and Paillier’s decision composite residuosity), so can the above PSI protocol.

As a result, in scenarios where PSI is to be combined with other (public key) cryptographic tools, it could be of interest to have at hand a SPH construction that allows for reusing global parameters and underlying assumptions.

| Protocol     | Comm. Overhead | Server Exp. | Client Exp. |
|--------------|----------------|-------------|-------------|
| [20]         | $\mathcal{O}(v + w)$ | $\mathcal{O}(w(\log \log v))$ | $\mathcal{O}(w + v)$ |
| [31]         | $\mathcal{O}(w + v)$ | $\mathcal{O}(vw)$ | $\mathcal{O}(v + w)$ |
| SPH-DDH      | $\mathcal{O}(vw)$ | $\mathcal{O}(vw)$ | $\mathcal{O}(v)$ |

Table 1. Performance comparison of SPH-based implementations vs prior public key implementations for PSI.

On the other hand, the computation and communication complexity of this scheme is (globally) higher than that of other known constructions. In Table 1 we give a brief summary of how an implementation of our scheme compares to previous proposals which are also public key, standard model (i.e., without random oracles) and only consider honest-but-curious adversaries ( [20, 31]). The cost for Client and Server is measured in terms of exponentiations. At this, we see that our scheme is outperformed due to its high communication overhead. Recent proposals outside the public key setting are indeed much more efficient, see [39, 58]. The cost of our scheme is evaluated considering a DDH-based construction for SPH depicted in [22], and assuming the corresponding commitments are implemented via CCA2-encryptions under the same assumption. Thus, the scenario considered is the same as in 7.2 of [22].

4.2. The size-hiding case. In this section we show that in the computational case size-hiding private set intersection is possible. We give two constructions. The first one is valid when the size of the ground set is polynomial in the security parameter and allows to state the equivalence between PSI, oblivious transfer (OT),

---

4DDH stands for Decisional Diffie Hellman.
and the secure computation of the AND function. This protocol was independently proposed in [26]. The second one is tailored for relatively small sets of secrets with an upper bound on their size being known a priori.

4.2.1. AND-based SH-PSI protocol. An oblivious transfer protocol is a two party protocol involving a sender and a receiver; the sender has two secrets, \( s_0 \) and \( s_1 \), while the receiver is interested in one of them. Its choice is represented by a bit \( \sigma \). After running the protocol the receiver gets \( s_\sigma \) and nothing else, while the sender does not learn which secret the receiver has recovered. Introduced by Rabin [40], and later on redefined in different equivalent ways, it is a key-tool in secure two-party and multi-party computation. We will denote this primitive by \( \text{OT}(s_0, s_1, \sigma) \).

It is well known folklore that a private AND protocol can be realized by using an \( \text{OT}(b_0, b_1, s) \) protocol: the bit \( b_s \) that the receiver gets can be expressed as \( b_s = (1 \oplus s) b_0 \oplus s b_1 \). As a result, invoking the instance \( \text{OT}(0, a, b) \), the receiver will get \( a \cdot b \).

Now, the key idea underlying our construction is that, if the set of secrets of \( C \) and \( S \) are represented by means of two characteristic vectors \( I_C \) and \( I_S \) of elements of \( U \) then, by running an AND\((I_{c_i}, I_{s_i})\) protocol for each bit of the vectors, \( C \) and \( S \) get the intersection and nothing else. Each \( \text{AND}(I_{c_i}, I_{s_i}) = 1 \) means that they share the \( i \)-th element of the ground set \( U \). Details are given in Figure 3.

Let \( \ell \) be a security parameter and let the universe \( U = \{u_1, \ldots , u_{|U|}\} \) be a ground set of size \( \text{poly}(\ell) \) (both known and fixed at the setup phase). Assume that \( C \) (resp. \( S \)) can be encoded in a characteristic vector \( I_C \) (resp. \( I_S \)), such that \( I_C[j] = 1 \) (resp. \( I_S[j] = 1 \)) iff the \( j \)-th element of \( U \) is in \( C \) (resp. \( S \)). It is easy to check that the protocol is correct. Moreover, it is secure as long as the AND protocol is. Thus, if we use the OT construction proposed in [18], which relies on the existence of enhanced trapdoor permutations (see [23, Definition C.1.1.,]) we obtain the following.

**Theorem 4.** The protocol given in Figure 3 when instantiated with the OT protocol of Figure 4.2.1. realizes SH-PSI, achieving correctness, client privacy and server privacy.

---

5For a practical instantiation one would use OT extension with a constant number of public-key operations, as it was done in [39].
Protocol steps (simplified description):

Let \( b_0, b_1 \in \{0, 1\} \) be the sender’s secret bits, and let \( i \in \{0, 1\} \) be the receiver’s choice.
Moreover, let \( \mathcal{F} = \{ f_\alpha : D_\alpha \rightarrow D_\alpha \}_{\alpha \in \{0, 1\}^{\text{poly}(t)}} \) be a family of enhanced trapdoor permutations.

- The Sender uniformly selects a trapdoor pair \((\alpha, t)\) and sends \(\alpha\) to the Receiver.
- The Receiver, uniformly and independently,
  - selects \(e_0, e_1 \in D_\alpha\);
  - sets \(y_i = f_\alpha(e_i)\) and \(y_{1-i} = e_{1-i}\);
  - sends \(y_0, y_1\) to the Sender.
- Using the trapdoor \(t\) and the inverting algorithm, for \(j = 0, 1\), the Sender computes
  \[x_j = f_\alpha^{-1}(y_j)\text{ and }c_j = b_j \oplus b(x_j),\]
where \(b(\cdot)\) is a hardcore predicate for \(f_\alpha\). Then, it sends \(c_0, c_1\) to the Receiver.
- The Receiver computes \(b_i = c_i \oplus b(e_i)\).

**Figure 4.** OT protocol based on trapdoor permutations.

**Proof. Correctness.** The SH-PSI protocol from Figure 3 is correct as long as the OT protocol is. Just note that

\[c_i \oplus b(e_i) = (b_i \oplus b(x_i)) \oplus b(e_i) = (b_i \oplus b(f_\alpha^{-1}(f_\alpha(e_i)))) \oplus b(e_i) = b_i.\]

**Client Privacy.** Intuitively, client privacy is guaranteed because in the OT protocol the Sender, independently of the bit \(i\) held by the Receiver, gets uniformly distributed values of \(D_\alpha\). Indeed, \(y_{1-i} = c_{1-i}\) is chosen uniformly at random in \(D_\alpha\) and \(y_i = f_\alpha(e_i)\) is still uniformly distributed in \(D_\alpha\) since \(e_i\) is chosen uniformly at random and \(f_\alpha\) is a permutation on \(D_\alpha\).

More precisely, a simulator \(\text{Sim}_S\) for the sender in the OT protocol will choose \(y_0, y_1 \in D_\alpha\) uniformly at random. From this one can construct a simulator for the server just running independently \(\text{Sim}_S\) for each instance of the AND computation. Note that the view produced by this simulator is identically distributed to the view of the sender in a real execution of the protocol and it is independent of the set of secrets held by the client. Then, it easily follows that the server does not distinguish between two executions in which the client has two different sets of secrets.

**Server Privacy.** On the other hand, in the OT protocol the Receiver gains no knowledge on the bit \(b_{1-i}\) because \(b(\cdot)\) is a hardcore predicate for \(f_\alpha\); as a result, from the triplet \((\alpha, e_{1-i}, c_{1-i} = b_{1-i} \oplus b(x_{1-i}))\) it is infeasible to predict \(b_{1-i}\) better than at random.

From this, we may construct a simulator \(\text{Sim}_R\) which acts as the receiver in the real protocol OT except that it chooses \(c_{i-1} \in \{0, 1\}\) uniformly at random. This in turn yields a simulator for the client of the SH-PSI protocol, by running \(\text{Sim}_R\) for each execution of the AND protocol. \(\square\)

**Remark 2.** In this section we have described a reduction from PSI to AND. Given the known reductions from AND to OT and from OT to PSI (the latter constructed in [20]), it is stated that OT, PSI and AND are equivalent.
4.2.2. Threshold-based protocol. Assuming some a priori information on the sizes of the input sets is known, more efficient protocols may be achieved. Here we assume that a known value $M$ upper-bounds the sizes of all clients’ and servers’ sets. The smaller $M$ is with respect to $|\mathcal{U}|$, the greater the interest of this construction (we need $M$ to be polynomial size, but $|\mathcal{U}|$ can be larger).

As our scheme is a simple twist of [20], the main tool used is also additively homomorphic encryption. Informally, a public-key encryption scheme is (additively) homomorphic if, for any two encryptions $\text{Enc}(m_1)$ and $\text{Enc}(m_2)$ of any two messages $m_1$ and $m_2$, it holds that $\text{Enc}(m_1) \cdot \text{Enc}(m_2) = \text{Enc}(m_1 + m_2)$, where $\cdot$ is a group operation on ciphertexts. By repeated application of the property, for any integer $c$, it follows that $\text{Enc}(m_1)^c = \text{Enc}(cm_1)$; as a result, given encryptions $\text{Enc}(a_0), \ldots, \text{Enc}(a_k)$ of the coefficients $a_0, \ldots, a_k$ of a polynomial $P$ of degree $k$, and a plaintext value $y$, it is possible to compute $\text{Enc}(P(y))$, i.e., an encryption of $P(y)$. Paillier’s cryptosystem [37] exhibits such properties and achieves semantic security under chosen plaintext attacks under the decisional composite residuosity assumption (see, for instance, [29, Section 11.3]).

The proposed scheme is depicted in Figure 5. At this, $\text{Enc}$ and $\text{Dec}$ denote the Paillier encryption and decryption algorithms, respectively. As the client is assumed to be honest, he will execute in the setup phase the key generation algorithm for Paillier encryption, at which, in particular two $\ell$-bit primes $p, q$ are fixed. At this point, he specifies an encoding of $\mathcal{U}$ into $\mathbb{Z}_n \setminus \{0\}$ with $|\mathcal{U}|/n$ being negligible. For the sake of readability, in Figure 5 elements of $\mathcal{U}$ are assumed to belong to $\mathbb{Z}_n \setminus \{0\}$—cf. Remark 3 below for exclusion of 0.

---

**Figure 5.** Polynomial-based construction for $|\mathcal{C}|, |\mathcal{S}| \leq M$. 

| **Setup** | **Client** | $\mathcal{C} = \{c_1, \ldots, c_v\}$ | **Server** | $\mathcal{S} = \{s_1, \ldots, s_w\}$ |
| --- | --- | --- | --- | --- |
| **Interaction** | $pk, sk \leftarrow \text{KeyGen}(1^\ell)$ | $\mathcal{P}(x) = x^{M-v} \sum_{j=1}^v (x - c_j)$ | $\{\text{Enc}(a_0), \ldots, \text{Enc}(a_M)\}$ | $pk$ |
| | $P(x) = \sum_{j=0}^M a_j x^j$ | | for $1 \leq i \leq w$ : | |
| | $r_i \leftarrow \mathbb{Z}_n$ | | $p_i = r_i \cdot P(s_i) + s_i$ | |
| | $e_i = \text{Enc}(p_i)$ | | for $w + 1 \leq i \leq M$ : | |
| | $\tilde{e}_i \leftarrow \mathbb{Z}_n$ | | $e_i = \text{Enc}(\tilde{e}_i)$ | |
| | $(e_1, \ldots, e_M)$ | | for $1 \leq j \leq v$ : | |
| | if $(\exists \ j : \text{Dec}(e_j) = c_j \land c_j \in \mathcal{C})$ | | output $c_j$ | |
Theorem 5. The protocol given in Figure 7 is an SH-PSI protocol achieving correctness, client privacy, and server privacy, assuming that the Paillier encryption scheme is secure in the sense of IND-CPA.

Proof. In the sequel, we set the notation $I := |C \cap S|$ and $L := w - I$.

Correctness. The client’s output is constructed by comparing his set $C$ with the one consisting of $S \cap C$ plus the decryption of $M - I$ encryptions of random values. Because of $|U|/n$ being negligible, the probability that the latter actually encode an element in $U$ (disrupting thus the computation of the intersection) is negligible.

Client Privacy. Due to the semantic security of Enc the distribution of

$$\{\text{Enc}(a_0), \ldots, \text{Enc}(a_M)\}$$

is indistinguishable of that induced by selecting $M + 1$ plaintexts independently and uniformly at random from $\mathbb{Z}_n$.

Server Privacy. In order to argue the existence of a probabilistic polynomial algorithm $C^*$ that is able to simulate the client’s view on input $C$ and $C \cap S$, we modify $C$’s view replacing the true input values from the server, constructed as encryptions involving values $s \in S \setminus C$ with encryptions of elements chosen uniformly and independently at random from $\mathbb{Z}_n \setminus \{0\}$. Consider thus the true distribution

$$D_0 := \{\{\rho_j\}_{j=0}^M; \{\text{Enc}(r_s P(s_j) + s_j)\}_{j=1}^w; \{\xi_j\}_{j=1}^{M-w}\}$$

where for $i = 0 \ldots M$ each $\rho_i$ denotes the random value involved in the Paillier encryption yielding $\text{Enc}(a_i)$, namely, they are values chosen uniformly and independently at random from $\mathbb{Z}_n^*$, and $\text{Enc}(r_s P(s_j) + s_j)$ for $j = 1, \ldots, w$ are computed as in the protocol description of Figure 7. Further w.l.o.g., we assume that $C \cap S = \{s_1, \ldots, s_l\}$ and so the elements $\xi_1, \ldots, \xi_{M-w}$ are chosen as encryptions of uniformly at random chosen plaintexts in $\mathbb{Z}_n$. Now, consider the distribution

$$D_L := \{\{\rho_j\}_{j=0}^M; \alpha_j; \{\nu_j\}_{j=1}^L; \{\xi_j\}_{j=1}^{M-w}\}$$

where $\alpha_j = \text{Enc}(r_s P(s_j) + s_j)$ for $j = 1, \ldots, I$, and the elements $\nu_1, \ldots, \nu_L$ are encryptions of elements chosen uniformly and independently at random from $\mathbb{Z}_n \setminus \{0\}$. As above, $\xi_1, \ldots, \xi_{M-w}$ are encryptions of elements chosen uniformly and independently at random from $\mathbb{Z}_n \setminus \{0\}$ as well. From the semantic security of Paillier encryption, it follows that these two distributions are computationally indistinguishable.

\[\square\]

Remark 3. Having Paillier encryption in mind, we have designed the polynomial $P$ in the interaction to have a small support, i.e., to have many 0-coefficients (as encryptions of 0 with Paillier are cheap). That is the reason for excluding 0 from the domain when defining the encoding of $U$ into $\mathbb{Z}_n$. Different refinements of this step may suit better if another encryption scheme is used, always ensuring that the resulting polynomial has no roots that may correspond to an encoding of an element outside $C$ and yet in $U$. In [20], abstracting from the concrete homomorphic encryption scheme used, several modifications of their basic protocol are proposed in order to boost efficiency, all of them geared toward reducing the number of multiplications and exponentiations performed by the server. These ideas would also yield a significant speed up versus a naïve implementation of the above protocol.
4.2.3. Algebraic PSI. In this section we give a theoretical generalization of the polynomial-based construction depicted in the previous section. Let us assume we are in the same scenario as above, namely, there exists a known value $M$ such that $|C|, |S| \leq M$. We summarize some facts about algebraic curves and their function fields used in the sequel. This type of tools was used by Chen and Cramer for secret sharing schemes and multi-party computation constructions in [7], which can be used as a more extensive reference.

Consider $\Upsilon$ a smooth, projective, absolutely irreducible plane curve defined over a prime field $K = \mathbb{F}_q$ and let $\overline{K}$ be the algebraic closure of $K$. Such a curve can be represented by a polynomial $F \in K[X, Y]$ that is irreducible over $\overline{K}[X, Y]$, where the affine part of the curve is the set of points $P \in \overline{K}^2$ such that $F(P) = 0$. We will only consider the rational points of the curve, that is, the points with coordinates in $K$. A divisor over $\Upsilon$ is a formal sum $D = \sum_{P \in \Upsilon} m_P \cdot P$, with only finitely many non-zero integer coefficients $m_P$. The sum of this finite number of integers is called the degree of the divisor and denoted by $\deg D$. Let $\overline{K}(\Upsilon)$ be the function field of the curve. Given a function $f \in \overline{K}(\Upsilon)$, the divisor of $f$ is defined as $\text{div}(f) = \sum_{P \in \Upsilon} \text{mul}_f(P) \cdot P$, where $\text{mul}_f(P)$ is the multiplicity of $f$ at $P$ (positive if $P$ is a zero of $f$, negative if it is a pole) and zero otherwise. The divisor of a function always has degree 0, that is, the sum of the multiplicities of zeroes of $f$ equals the sum of the multiplicities of the poles.

The Riemann-Roch space associated to a divisor $D$ is the set
\[
\mathcal{L}(D) = \{ f \in \overline{K}(\Upsilon) \mid \text{div}(f) + D \geq 0 \} \cup \{0\},
\]
where the partial order relation between divisors is determined by a divisor being $\geq 0$ if all their coefficients are non-negative integers. $\mathcal{L}(D)$ is a finite dimensional vector space over $\overline{K}$ and the Riemann-Roch theorem provides its dimension $\ell(D)$. In particular, if $\deg D \geq 2g - 1$ then $\ell(D) = \deg(D) - g + 1$, where $g$ is the genus of $\Upsilon$. A divisor is rational if it is invariant under the Galois group $\text{Gal}(\overline{K}/K)$. This includes divisors whose support consists only of rational points. Moreover, if $D$ is rational then $\mathcal{L}(D)$ has a basis over $K$. Therefore we will restrict to the subspace of $\mathcal{L}(D)$ which is fixed under $\text{Gal}(\overline{K}/K)$ and consists of $q^\ell(D)$ different functions. Additionally, $f(P) \in K$ if $P$ is rational.

Let us fix an algebraic curve $\Upsilon$ of genus $g$ as described above, denote by $R_\Upsilon$ the set of rational points of the curve, and fix $Q \in R_\Upsilon$. Choose $N = M + g - 1$, where $M \geq |C|, |S|, g$, and fix the divisor $D = N \cdot Q$. For our protocol to work we need the following

**Assumption 1.** Given a set $\mathcal{C} \subseteq R_\Upsilon \setminus \{Q\}$ satisfying $|\mathcal{C}| \leq M$, there exists a function $f \in \mathcal{L}(D)$ such that $\mathcal{Z}(f) = \{ P \in R_\Upsilon \mid f(P) = 0 \} = \mathcal{C}$.

Given a concrete curve $\Upsilon$, it is not obvious how realistic this assumption is. We believe this question to be of interest in its own right, beyond the cryptographic application depicted here, but with this assumption, it is relatively straightforward to generalize the protocol from Section 4.2.2. By definition, the space $\mathcal{L}(D)$ consists of functions with no poles other than $Q$ and such that the multiplicity of $Q$ as a pole is less than or equal to $\ell(D)$. As $\deg D = N = M + g - 1 \geq 2g - 1$, by the Riemann-Roch theorem, $\ell(D) = \deg(D) - g + 1 = N - g + 1 = M$, thus it is possible to compute a basis $\mathcal{B} = \{f_1, f_2, \ldots, f_M\}$ of $\mathcal{L}(D)$ over $K$.

Assume the universe $\mathcal{U}$ is encoded publicly as a subset of $R_\Upsilon \setminus \{Q\}$ and that $|\mathcal{U}|/|K|$ is negligible. We use a collision resistant hash function $H : R_\Upsilon \rightarrow K$. The
public parameters for the scheme are \( \text{params} = (\Upsilon, Q, M, B, H) \). In the setup phase, the client executes the key generation algorithm for a homomorphic IND-CPA secure encryption scheme defined over \( K \) (see, e.g., [2]) and sends the public key to the server. The interaction phase is analogous to the protocol in Section 4.2.2. A description is provided in Figure 6. Note that given a point \( s \in S \), from \( \{\text{Enc}(a_i)\}_{i=1}^M \), sent by the client, and \( \{\text{Enc}(f_i(s))\}_{i=1}^M \), which can be evaluated by the server, she can compute \( \text{Enc}(f(s)) \).

**Remark 4.** If there is no need to hide the size of the server’s set, no “fake” encryptions need to be sent in the last round of the protocol (saving communication complexity) and also the condition \( |S| \leq M \) can be dropped. With these two modifications we would obtain a protocol which is unbalanced size-hiding in the sense defined in the next section.

**Remark 5.** For the above construction we would need to be able to compute rational functions from certain divisors. Already, the problem of deciding if a divisor \( D \) of degree 0 is actually the divisor of a rational function seems to be highly nontrivial. If the curve \( \Upsilon \) is defined over the field \( C \), that is, \( \Upsilon \) is a Riemann surface, Abel’s theorem (see [33], chapter VIII, section 2) provides a characterization: \( D \) is the divisor of a meromorphic function on \( \Upsilon \) if and only if \( A_0(D) = 0 \) in the Jacobian of \( X \), where \( A_0 \) is the Abel-Jacobi map. For elliptic curves over \( C \), this translates to the following result: \( D \) is the divisor of a meromorphic function if and only if \( D = 0 \) (when the formal sum which defines the divisor is seen as an actual sum in the elliptic curve). As far as we know, this problem has not been studied for algebraic curves over finite fields (we refer here to [34] as a complete introduction to this field).
5. Unbalanced size-hiding aided by a setup authority

In this section, we follow the spirit of [3] and try to provide unbalanced private set intersection protocols, i.e., protocols in which the client learns $|S|$ from the interaction, while keeping $|C|$ secret. We follow the definitions of correctness, client privacy, and server privacy from [3].

We propose two different versions of essentially the same protocol, which is based on the ideas in [25], where a function chosen from a PRF family is obliviously evaluated by the server on each of the client’s elements (as roughly described in Section 1). In our constructions, the function evaluation is delegated to the setup authority $SA$ in the setup phase. Depending on the degree of trust the participants put in $SA$, the function evaluation is carried out in a different way, leading to the two versions of our protocol. Namely, if $SA$ is fully trusted, each participant sends its input set to $SA$ for direct function evaluation (in the sequel we will refer as the protocol using this setup as the PRF-PSI-protocol, see Figure 7). While, if participants do not wish to reveal their sets, a two-party protocol between $SA$ and each participant $P$ is run, resulting in $SA$ obliviously evaluating the chosen function on $P$’s inputs (resulting in our so-called OPRF-PSI-protocol, see Figure 8). Later on, in the interaction phase (where $SA$ is no longer available), the two participants (playing the roles of client $C$ and server $S$ respectively) engage in a one-round protocol where $S$ sends the values she obtained from $SA$ to $C$, which, by comparison with his own evaluated values, obtains the intersection. Note that USH is trivially achieved as the client sends no message to the server.

It is worth mentioning that our PRF protocol is conceptually very close to the protocol by Hayay et al. [24], where $SA$ is implemented by a trusted smart card which is first used by the server to compute the evaluations of his items and then sent to the client to do the same. When we originally proposed this protocol in [16] we were not aware of the existence of this other protocol.

Remark 6. Note that for correctness of our protocols, the range of the functions from the family $\{f_r\}_{r \in K}$ should be chosen such that the probability of collision for every $f_r$ is negligible in the security parameter.

Theorem 6. The protocol given in Figure 7 is a USH-PSI achieving correctness, client privacy, and server privacy under the assumption that $\{f_r\}_{r \in K}$ is a PRF family.

Proof. Correctness. Every $c_j \in C \cap S$ is output by the protocol. An element $c_j \notin C \cap S$ is output if and only if $f_r(c_j) = f_r(s_i)$ with $c_j \neq s_i$, and this happens only with negligible probability if $\{f_r\}_{r \in K}$ is chosen as in Remark 6.

Client Privacy. This is trivially achieved as the client sends nothing to the server.

Server Privacy. This follows from the the pseudorandomness property of the function family $\{f_r\}_{r \in K}$. At this, a probabilistic polynomial algorithm $C^*$ can simulate the client’s view on input $C, C \cap S$, and $|S|$, by constructing a sequence $\{R^*_1, \ldots, R^*_|S|\}$ so that, for each $u_i \in C \cap S$, a corresponding $R^*_i$ is defined as $f_r(u_i)$, while the rest are values chosen independently and uniformly at random in the range of $f_r$. \qed

Correctness is defined including $|S|$ as part of the client’s output, the definition of client privacy coincides with the one we have given here for SH-PSI protocols, while server privacy ensures the client’s view is polynomial-time simulatable on input $C, S \cap C$, and $|S|$ (the latter definition does not coincide with the one from [3], but $|S|$ is needed in the simulation included in their proof).
Participant $P \in C \cup S$

\[ P = \{p_1, \ldots, p_u\} \]

\[ P' = \{f_r(p_1), \ldots, f_r(p_u)\} \]

\[ P = \{p_1, \ldots, p_u\} \]

\[ P' = \{f_r(p_1), \ldots, f_r(p_u)\} \]

\[ \text{For } j = 1, \ldots, u \]

\[ p_j \rightarrow f_r(p_j) \]

\[ r \rightarrow f_r \]

\[ \perp \]

\[ \text{For } j = 1, \ldots, u \]

\[ p_j \rightarrow f_r(p_j) \]

\[ r \rightarrow f_r \]

\[ \perp \]

\[ \text{for } 1 \leq j \leq v \]

if $f_r(c_j) \in S'$

Output $c_j$

**Figure 7.** The PRF-PSI-protocol.

Now let us briefly describe our OPRF protocol. The setup phase is depicted in Figure 8 while the interaction is the same as in Figure 7. The only difference between the two protocols is that in the OPRF one SA does not learn anything about the participant set (beyond its size) as long the OPRF two-party protocol used in the setup is secure. Therefore we obtain a security result for this protocol similar to Theorem 6.

**Theorem 7.** The protocol with setup given in Figure 8 and interaction from Figure 7 is a USH-PSI achieving correctness, client privacy, and server privacy under the assumption that \( \{f_r\}_{r \in K} \) is a PRF family.

**Proof.** Same as Theorem 6.

In order to implement the PRF-PSI-protocol in a very efficient way, the PRF family could be instantiated by using a block cipher like AES. For the OPRF-PSI-protocol, the efficient proposal from [35] can be used. Let us briefly describe this pseudorandom function family: let $G$ be a cyclic group of prime order $q$ and $g$ a generator of $G$ for which the DDH assumption holds. We will denote by $K = (\mathbb{Z}_q)^n$, a set of keys for a function family. Elements in $K$ have the form $r = (r_1, \ldots, r_n)$. The family of functions $\{f_r\}_{r \in K}$, defined as follows, is proven to be pseudorandom. 

\[ \mathbb{G} \]
under the DDH assumption (see [35] for details).

\[
f_r : \{0,1\}^n \rightarrow G \\
\left(x_1, \ldots, x_n\right) \mapsto g^{\prod_{i=1}^n r_i x_i} = g^{\prod_{i=1}^n r_i}.
\]

Therefore, we will additionally need an encoding of the ground set \(U\) into the set \(\{0,1\}^n\) for big enough \(n\). The protocol proposed in [19] can be used to evaluate \(f_r\) in an oblivious way, suitable for the protocol in Figure 8.

Finally let us compare our two protocols with [25]. Our main motivation to introduce our protocols is to obtain the size-hiding property, which [25] does not provide (at the expense of introducing a third party, the SA). For the OPRF protocol, the efficiency and communication complexity remains equal than those in [25], as the same messages and computations are made, the only difference is that they are split between the server and the SA. When we consider the PRF protocol, again the communication is the same and also the number of function evaluations; but, in this case these can be direct evaluations (do not need to be oblivious). The performance of this protocol is the same as the one from [24].

6. Conclusion

We have explored the private set intersection problem when hiding the sizes of the input sets is relevant; for SH-PSI, we provide a (conceptual) protocol in the unconditional setting (with the help of a trusted party in a setup phase). Furthermore, we proved that PSI is impossible in the unconditional setting, making explicit its relation to AND. In addition, in the computational scenario, we have both given a theoretical construction (only applicable for polynomial universes) and a practical one which is a simple twist of the well-known polynomial scheme of Freedman et al. [20].

For USH-PSI we have proposed two practical and efficient one-round protocols based on the ideas from [25]. The additional size-hiding property is achieved thanks to the introduction of a trusted (or semi-trusted) setup authority which interacts independently once with each participant in a setup phase and whose intervention is not needed any more for later PSI computations between the participants.

Acknowledgments

The authors are indebted to Vicente Muñoz for his helpful suggestions regarding the algebraic geometric construction from subsection 4.2.3. We also thank the anonymous referees for their valuable comments and links to recent literature.

References

[1] M. Abdalla, F. Benhamouda, O. Blazy, C. Chevalier and D. Pointcheval, SPHF-friendly non-interactive commitments, in Adv. Crypt. – ASIACRYPT 2013, Springer, 2013, 214–234.
[2] F. Armknecht, D. Augot, L. Perret and A.-R. Sadeghi, On constructing homomorphic encryption schemes from coding theory, in Crypt. Coding (ed. L. Chen), Springer, 2011, 23–40.
[3] G. Ateniese, E. De Cristofaro and G. Tsudik, (If) size matters: size-hiding private set intersection, in Publ. Key Crypt. – PKC 2011 (eds. D. Catalano et al), Springer, 2011, 156–173.
[4] P. Baldi, R. Baronio, E. De 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 – CCS 2011 (eds. Y. Chen et al), ACM, 2011, 691–702.
[5] J. Camenisch and G. M. Zaverucha, Private intersection of certified sets, in Financial Crypt. Data Sec. – FC 2009 (eds. D. R. Dingledine et al), IFCA, Springer, 2009, 108–127.
[6] M. Chase and I. Visconti, Secure database commitments and universal arguments of quasi knowledge in Advances in Cryptology – CRYPTO 2012 (eds. R. Safavi-Naini et al), Springer, 2012, 236–254.

[7] H. Chen and R. Cramer, Algebraic geometric secret sharing schemes and secure multi-party computations over small fields in Adv. Crypt. – CRYPTO 2006 (ed. C. Dwork), Springer, 2006, 521–536.

[8] R. Cramer, Introduction to secure computation in Lect. Data Sec. (ed. I. Damgård), Springer, 1999, 16–62.

[9] R. Cramer and V. Shoup, Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption in Adv. Crypt. – EUROCRYPT 2002 (ed. L.R. Knudsen), Springer, 2002, 45–64.

[10] E. De Cristofaro, S. Faber and G. Tsudik, Secure genomic testing with size- and position-hiding private substring matching, in ACM Workshop Priv. Electr. Soc. – WPES’13 (eds. A.-R. Sadeghi et al), ACM, 2013, 107–118.

[11] E. De Cristofaro, S. Jarecki, J. Kim and G. Tsudik, Privacy-preserving policy-based information transfer, in Priv. Enhanc. Techn. – PETS 2009 (eds. I. Goldberg et al), Springer, 2009, 164–184.

[12] E. De Cristofaro, J. Kim and G. Tsudik, Linear-complexity private set intersection protocols secure in malicious model, in Adv. Crypt. – ASIACRYPT 2010 (ed. M. Abe), Springer, 2010, 213–231.

[13] E. De Cristofaro and G. Tsudik, Practical private set intersection protocols with linear complexity, in Financial Crypt. Data Sec. – FC 2010 (ed. R. Sion), IFCA, Springer, 2010, 143–159.

[14] D. Dachman-Soled, T. Malkin, M. Raykova and M. Yung, Efficient robust private set intersection in Appl. Crypt. Netw. Sec. – ACNS 2009 (eds. M. Abdalla et al), Springer, 2009, 125–142.

[15] D. Dachman-Soled, T. Malkin, M. Raykova and M. Yung, Secure efficient multiparty computing of multivariate polynomials and applications, in Applied Cryptography and Network Security – ACNS 2011 (eds. J. Lopez et al), Springer, 2011, 130–146.

[16] P. D’Arco, M. I. González Vasco, A. L. Pérez del Pozo and C. Soriente, Size-hiding in private set intersection: existential results and constructions in Progr. Crypt. – AFRICACRYPT 2012 (eds. A. Mitrokotsa et al), Springer, 2012, 378–394.

[17] C. Dong, L. Chen and Z. Wen, When private set intersection meets big data: an efficient and scalable protocol, in ACM SIGSAC Conf. Comp. Commun. Sec. – CCS 2013 (eds. A.-R. Sadeghi et al), ACM, 2013, 789–800.

[18] S. Even, O. Goldreich and A. Lempel, A randomized protocol for signing contracts Commun. ACM, 28 (1985), 637–647.

[19] M. J. Freedman, Y. Ishai, B. Pinkas and O. Reingold, Keyword search and oblivious pseudorandom functions in Theory Crypt. – TCC 2005 (ed. J. Kilian), Springer, 2005, 303–324.

[20] M. J. Freedman, K. Nissim and B. Pinkas, Efficient private matching and set intersection in Adv. Crypt. – EUROCRYPT 2004 (eds. C. Cachin et al), Springer, 2004, 1–19.

[21] K. Frikken, Privacy-preserving set union, in Applied Crypt. Netw. Sec. – ACNS 2007 (eds. J. Katz et al), Springer, 2007, 237–252.

[22] R. Gennaro and Y. Lindell, A framework for password-based authenticated key exchange (extended abstract) in Adv. Crypt. – EUROCRYPT 2003 (ed. E. Bham), Springer, 2003, 524–543.

[23] O. Goldreich, Foundations of Cryptography, Volume II. Basic Applications, Cambridge Press, 2004.

[24] C. Hazay and Y. Lindell, Constructions of truly practical secure protocols using standard smartcards, in Proc. 15th ACM Conf. Comp. Commun. Sec., ACM, 2008, 491–500.

[25] C. Hazay and Y. Lindell, Efficient protocols for set intersection and pattern matching with security against malicious an covert adversaries in Theory Crypt. – TCC 2008 (ed. R. Canetti), Springer, 2008, 155–175.

[26] Y. Huang, D. Evans and J. Katz, Private set intersection: Are garbled circuits better than custom protocols?, in Network and Distributed System Security Symposium (NDSS). The Internet Soc., 2012.

[27] S. Jarecki and X. Liu, Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection in Theory Crypt. – TCC 2009 (ed. O. Reingold), Springer, 2009, 577–594.
[28] S. Jarecki and X. Liu, Fast secure computation of set intersection, in Sec. Crypt. Netw. – SCN 2010 (eds. J.A. Garay et al), Springer, 2010, 418–435.

[29] J. Katz and Y. Lindell, Introduction to modern cryptography, in Cryptography and Network Security Series, Chapman & Hall/CRC, 2007.

[30] F. Kerschbaum, Outsourced private set intersection using homomorphic encryption, in ACM Symp. Inf. Comp. Commun. Sec. – ASIACCS 2012, ACM, 2012, 85–86.

[31] L. Kissner and D. Song, Privacy-preserving set operations in Adv. Crypt. – CRYPTO 2005 (ed. V. Shoup), Springer, 2005, 241–257.

[32] Y. Lindell, K. Nissim and C. Orlandi, Hiding the input-size in secure two-party computation in Adv. Crypt. – ASIACRYPT 2013 (eds. K. Sako et al), Springer, 2013, 421–440.

[33] R. Miranda, Algebraic Curves and Riemann Surfaces volume 5, 1995.

[34] C. Moreno, Algebraic Curves over Finite Fields Cambridge Univ. Press, 1991.

[35] M. Naor and O. Reingold, Number-theoretic constructions of efficient pseudo-random functions J. ACM, 51 (2004), 231–262.

[36] R. Nojima and Y. Kadobayashi, Cryptographically secure bloom-filters, Trans. Data Privacy, 2 (2009), 131–139.

[37] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes in Adv. Crypt. – EUROCRYPT’99 (ed. J. Stern), Springer, 1999, 223–238.

[38] B. Pinkas, T. Schneider, G. Segev and M. Zohner, Phasing: private set intersection using permutation-based hashing, in 24rd USENIX Sec. Symp., USENIX Assoc., 2015, 515–530.

[39] B. Pinkas, T. Schneider and M. Zohner, Faster private set intersection based on OT extension, in 23rd USENIX Sec. Symp., USENIX Assoc., 2014, 797–812.

[40] M. Rabin, How to Exchange Secrets by Oblivious Transfer, Technical Report TR-81, Aiken Comput. Lab., Harvard Univ., 1981.

[41] R. Rivest, Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer, 1999, available at http://people.csail.mit.edu/rivest/publications.html

Received July 2015; revised December 2015.

E-mail address: pdarco@unisa.it
E-mail address: mariaisabel.vasco@urjc.es
E-mail address: angel.perez@urjc.es
E-mail address: claudio.soriente@telefonica.com
E-mail address: rsteinwa@fau.edu