Two-Client and Multi-client Functional Encryption for Set Intersection

Tim van de Kamp David Stritzl
Willem Jonker Andreas Peter

University of Twente
{t.r.vandekamp,d.l.stritzl}@alumnus.utwente.nl,
{w.jonker,a.peter}@utwente.nl

Abstract. We propose several functional encryption schemes for set intersection and variants on two or multiple sets. In these schemes, a party may learn the set intersection from the sets of two or more clients, without having to learn the plaintext set of each individual client. For the case of two clients, we construct efficient schemes for determining the set intersection and the cardinality of the intersection. To evaluate the cardinality of the intersection, no overhead is incurred when compared to operating on plaintext data. We also present other functionalities with a scheme for set intersection with data transfer and a threshold scheme that only discloses the intersection if both clients have at least t elements in common. Finally, we consider set intersection and set intersection cardinality schemes for the case of three or more clients from a theoretical perspective. Our proof-of-concept implementations show that the two-client constructions are efficient and scale linearly in the set sizes.

Keywords: multi-client functional encryption · non-interactive · set intersection

1 Introduction

In a functional encryption (FE) scheme, decryption keys are associated with a functionality f and the decryption of an encrypted message m returns the function

This is the full version of the paper appearing in the procedings of ACISP 2019.
The final authenticated version is available online at DOI: 10.1007/978-3-030-21548-4_6.

applied to the message, $f(m)$, instead of the original message m. This concept can be extended to functions with more than one input, resulting in a multi-input functional encryption (MI-FE) scheme. Correspondingly, the decryption algorithm of an MI-FE scheme requires a decryption key, associated with an n-ary function f, and n encrypted values $x_1, \ldots, x_n$ to output $f(x_1, \ldots, x_n)$.

A strict subset of these MI-FE schemes are termed multi-*client* functional encryption (MC-FE) schemes [GGG+14]. In such an MC-FE scheme, the inputs for the n-ary function f are given by n distinct parties, termed *clients*. Each client encrypts their input using their own encryption key and a *time-step* or *session identifier*. This identifier is used to determine which ciphertexts from the various clients belong together. To evaluate a function f using the corresponding decryption key, all inputted ciphertexts need to be associated with the same identifier or otherwise decryption will fail.

In this work, we explore the set intersection functionality and several variants. Inspired by the popularity of private set intersection (PSI) protocols [PSZ14], we define a scheme for determining the set intersection of two clients' sets in a *non-interactive* manner. Additionally, we propose several other non-interactive variants of interactive PSI protocols that were previously proposed in literature. We construct a two-client functional encryption (2C-FE) scheme for determining the cardinality of the intersection (*i.e.*, $|\mathcal{S}_a \cap \mathcal{S}_b|$, where $\mathcal{S}_\gamma$ is the set belonging to client γ), similar to PSI cardinality [KS05]. We also consider a non-interactive 2C-FE version of the less common PSI with data transfer [DT10; JL10], where the common set elements are shared with associated data (*i.e.*, $\{\, (x_j, \varphi_a(x_j), \varphi_b(x_j)) \mid x_j \in \mathcal{S}_a \cap \mathcal{S}_b \,\}$, where $\varphi_\gamma(x_j)$ is the data associated with x_j by client γ). Finally, we construct a threshold scheme where the set intersection is only revealed if two clients have at least t set elements in common.

Following our 2C-FE schemes, we also explore the much harder multi-client case where we propose MC-FE schemes for determining the (cardinality of the) set intersection of more than two sets. While 2C-FE schemes could also be used to determine the intersection of multiple sets, doing so would leak information about the intersection of each pair of sets. To prevent this undesirable leakage and achieve secure MC-FE for set intersection, we require more involved constructions.

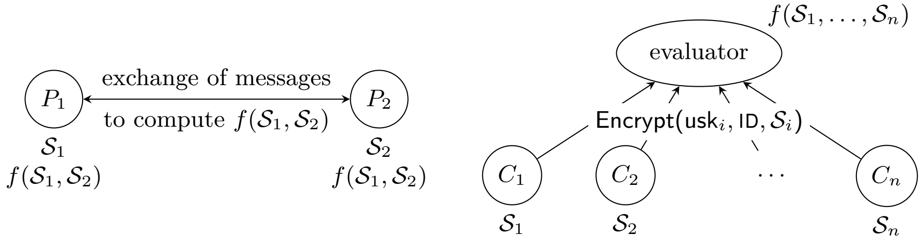
An overview of constructions for MC-FE for set intersection presented in this work is given in Table 1.

Although the functionalities for our MC-FE schemes are inspired by various PSI protocols, the usage scenario differs in a crucial way: We apply our MC-FE schemes in a scenario where a third party, termed the *evaluator*, learns the function outcome. In Section 5.1 we explain why non-interactive 2C-FE cannot be secure if one of the clients also serves as the evaluator. We highlight the difference between PSI and our MC-FE for set intersection in Figure 1.

Using the functionalities provided by our constructions, it is possible to achieve

Table 1: Overview of the presented MC-FE schemes for set operations.

functionality	two-client	multi-client
set intersection	§ 6.2	§ 7.3
set intersection cardinality	§ 6.1	§§ 7.1, 7.2
set intersection with data transfer	§ 6.3	open problem
threshold set intersection	§ 6.4	open problem


P_1
exchange of messages
to compute $f(\mathcal{S}_1, \mathcal{S}_2)$
P_2
$\mathcal{S}_1$
$f(\mathcal{S}_1, \mathcal{S}_2)$
$\mathcal{S}_2$
$f(\mathcal{S}_1, \mathcal{S}_2)$
$f(\mathcal{S}_1, \ldots, \mathcal{S}_n)$
evaluator
$\mathsf{Encrypt}(\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i)$
C_1
C_2
$\cdots$
C_n
$\mathcal{S}_1$
$\mathcal{S}_2$
$\mathcal{S}_n$

(a) A typical scenario of private set intersection (PSI). Both parties learn the output of the function evaluation, but not each other's inputs.

(b) Our scenario of MC-FE for set intersection. The evaluator learns the function evaluation and nothing else about the clients' inputs.

Figure 1: Fundamental difference between a private set intersection (PSI) protocol and our multi-client functional encryption (MC-FE) schemes for set intersection.

privacy-preserving profiling. For example, consider a case where the police is looking for suspects which were both present at a concert and recently received a large sum of money on their bank account. Using a 2C-FE scheme for determining the set intersection, the police will only learn about the suspects matching the two profiles, while learning nothing about the other visitors of the concert or other people that received an unusual amount of money. Another use case is privacy-preserving data mining, such as the computation of various set similarity scores. For example, by determining the cardinality of a set intersection we can compute the Jaccard index (*i.e.*, $|\mathcal{S}_1 \cap \mathcal{S}_2|/|\mathcal{S}_1 \cup \mathcal{S}_2| = |\mathcal{S}_1 \cap \mathcal{S}_2|/(|\mathcal{S}_1| + |\mathcal{S}_2| - |\mathcal{S}_1 \cap \mathcal{S}_2|)$), without requiring the evaluator to learn the clients' sets themselves.

To asses the practicability of our constructions, we implemented several of our proposed schemes. Our 2C-FE constructions are quite efficient: Determining the cardinality of the set intersection of two encrypted sets is *as fast as any plaintext solution* and determining the set intersection of sets of 100 thousand elements in size can be done in just under a second.

2 Preliminaries

A t-OUT-OF-n Shamir's secret sharing scheme (SSSS) uses a t-degree polynomial f over a finite field $\mathbb{F}_p$. To share the secret s, pick a random polynomial with $f(0) = s$ and pick shares $\big(i, f(i)\big)$ for distinct values i. To recover the secret from a set of at least t distinct shares $\big\{\big(i, f(i)\big)\big\}_{i \in \mathcal{S}}$, Lagrange interpolation is used, $f(0) = \sum_{i \in \mathcal{S}} f(i) \cdot \Delta_{\mathcal{S},i}$, where $\Delta_{\mathcal{S},i} = \prod_{j \in \mathcal{S}, j \neq i}\big(j \cdot (j-i)^{-1}\big)$.

A Bloom filter is a data structure that can be used for efficient set membership testing. An (m, k) Bloom filter consists of a bit string bs of length m (indexed using $\mathsf{bs}[\ell]$ for $1 \leq \ell \leq m$) and is associated with k independent hash functions, $h_i \colon \{0,1\}^* \to \{1, \ldots, m\}$ for $1 \leq i \leq k$. The Bloom filter is initialized with the bit string of all zeros. To add an element x to the Bloom filter, we hash the element for each of the k hash functions to obtain $h_i(x)$ and set the $h_i(x)$th position in the bit string bs to 1, *i.e.*, $\mathsf{bs}[h_i(x)] = 1$ for $1 \leq i \leq k$. To test the membership of an element x^*, we simply check if $h_i(x^*) = 1$ for $1 \leq i \leq k$.

Note that Bloom filters have no false negatives for membership testing, but may have false positives. Furthermore, we point out that the hash functions h_i do not necessary need to be cryptographic hash functions.

3 Related Work

While the term MC-FE [GGG$^+$14] only recently gained traction, a couple of MC-FE schemes have already been proposed several years ago. For example, for the functionality of summing inputs from distinct clients, Shi *et al.* [SCR$^+$11] proposed a construction. Around the same time, Lewko and Waters [LW11] proposed a multi-authority attribute-based encryption scheme. Their construction can also be seen as MC-FE since the evaluated function only outputs a plaintext if the user has the right inputs (*i.e.*, attributes) to the function (*i.e.*, policy). More recently, MC-FE constructions for computing vector equality [KPE$^+$17] and inner products [CDG$^+$18; ABK$^+$19] have been proposed. However, no MC-FE schemes for functionalities related to set operations have been proposed.

Despite being interactive by definition, PSI protocols are functionality-wise the closest related to our constructions. While the concept of PSI dates from the mid-80s [Mea86], renewed interest in PSI protocols started in the beginning of the new millennium [FNP04; KS05]. A comprehensive overview of various PSI constructions and techniques is given by Pinkas, Schneider, and Zohner [PSZ14]. While most PSI constructions achieve their functionality through techniques different from ours, Bloom filters have been used by interactive PSI protocols before [Ker12b; DCW13].

The type of PSI protocols that are most related to our MC-FE schemes are termed *outsourced* PSI [Ker12a; Ker12b; KMR$^+$14; LNZ$^+$14; ZX15; ATD15;

ATD16]. In outsourced PSI, a client may upload its encrypted set to a *service provider*, which will then engage in a PSI protocol on the client's behalf. Hence, in outsourced PSI the other client still learns the outcome of the evaluated set intersection, while in our definition of MC-FE for set intersection we require a dedicated evaluator to learn this outcome. This difference is typified by the difference in homomorphic encryption and FE: While both techniques allow us to compute over encrypted data, with homomorphic encryption we learn the *encrypted* output of the computation while with FE we learn the *plaintext* result. The two-client set intersection protocol by Kerschbaum [Ker12a] is a notable exception to regular outsourced PSI: In that construction the service provider also learns the outcome of the set intersection. However, besides their limited scope of considering only two-client set intersection, they consider a weaker security notion. Their construction is only collusion resistant if the two clients collude against the evaluator, not if the evaluator colludes with one client against the other client (something we show impossible in Section 5.1). As a consequence, their construction cannot be extended to a secure scheme in the multi-client case. Moreover, their proposed construction is malleable and thus does not provide any form of integrity.

4 Multi-client Functional Encryption for Set Operations

An MC-FE [GGG+14] scheme for a specific set operation consists of n parties, termed *clients*. Each of these clients encrypts their own set. Another party, which we term *evaluator*, having a decryption key and receiving these encrypted sets, can evaluate an n-ary set operation f over the clients' inputs.

To run the same functionality f multiple times without the possibility for the evaluator to mix old clients' inputs with newly received inputs, MC-FE schemes associate an identifier ID with every ciphertext. An evaluator is only able to evaluate the function if all ciphertexts use the same identifier ID.

The MC-FE schemes we propose support only a single functionality f (*e.g.*, set intersection). Therefore, our schemes do not need to define a key generation algorithm to create a decryption key for each of the functionalities. Instead, we can suffice with the creation of a decryption key for the single functionality in Setup. This type of FE schemes is commonly referred to as *single key* [KLM+18]. However, to avoid confusion in our multi-client case—where we still have a key for each client—we refer to this setting as *single evaluation key* MC-FE.

Definition 1 (Multi-client Functional Encryption for Set Operations)**.** A single evaluation key MC-FE scheme for set operation f, consists of the following three polynomial time algorithms.

$\mathsf{Setup}(1^\lambda, n) \to (\mathsf{pp}, \mathsf{esk}, \mathsf{usk}_1, \ldots, \mathsf{usk}_n)$. On input of the security parameter λ and the number of clients, the algorithm outputs the public parameters pp, the evaluator's evaluation key esk, and the clients' secret keys usk_i for each client $1 \leq i \leq n$. The public parameters are implicitly used in the other algorithms.

$\mathsf{Encrypt}(\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i) \to \mathsf{ct}_{\mathsf{ID},i}$. For a client i to encrypt a set $\mathcal{S}_i$ for identifier ID, the client uses its secret key usk_i and outputs the ciphertext $\mathsf{ct}_{\mathsf{ID},i}$.

$\mathsf{Eval}(\mathsf{esk}, \mathsf{ct}_{\mathsf{ID},1}, \ldots, \mathsf{ct}_{\mathsf{ID},n}) \to f(\mathcal{S}_1, \ldots, \mathcal{S}_n)$. An evaluator having the evaluation key esk and a ciphertext for identifier ID from every client, outputs the function evaluation $f(\mathcal{S}_1, \ldots, \mathcal{S}_n)$.

4.1 Schemes Without an Evaluator Key

While having schemes with an evaluation secret key might be desirable in some cases, in other cases it is desirable that anyone may learn the outcome of the function, *e.g.*, similar to property-revealing encryption [PR12; BLR+15]. However, observe that we can *always* adapt an MC-FE scheme without an evaluation key to the above defined single evaluation key MC-FE by using public key encryption. Indeed, instead of sending the ciphertexts resulting from the MC-FE scheme directly to the evaluator, we simply require the clients to encrypt these ciphertexts again, but now using the public key of the evaluator. This ensures that only the evaluator with the corresponding private key (used as an evaluation key) can evaluate the functionality f. An alternative solution is to require the clients to send their ciphertexts over a secure channel to the evaluator. This way, no other party has access to the ciphertexts.

We conclude that, since schemes without an evaluation key can be turned into a single evaluation key MC-FE scheme, MC-FE schemes without an evaluation key are at least as powerful as single evaluation key MC-FE. For this reason, *we construct only* MC-FE *schemes without an evaluation key* and stress that our resulting schemes can thus be used both *with* and *without* an evaluation key.

5 Security

We use the indistinguishability-based security notion from Goldwasser *et al.* [GGG+14, § 3.2] for MC-FE. In this notion, the adversary's goal is to decide which of the two, by the adversary chosen, plaintexts is encrypted. The notion allows the adversary to adaptively query for the encryption of plaintext, while it can locally evaluate the received ciphertext using $\mathsf{Eval}(\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$. Additionally, the adversary is allowed to statically corrupt the clients by announcing the corrupted clients before it receives the public parameters.

The adversary can thus be seen as a *malicious evaluator* that tries to learn information about the ciphertexts, other than what it should be allowed according to the functionality of the scheme. In its attempts, the malicious evaluator may collude with the clients in an effort to learn more about other clients' ciphertexts.

Let f be the supported function of the MC-FE scheme for n clients. This function has n inputs, one for every client. For a subset $I \subseteq \{1, \ldots, n\}$, we use the notation $f(\{x_i\}_{i \in I}, \cdot)$ to denote the function that has its inputs x_i, for $i \in I$, hardwired in the function.

Definition 2 (Adaptive IND-security of MC-FE [GGG+14])**.** An MC-FE scheme *without an evaluation key* is *secure* if any probabilistic polynomial time (p.p.t.) adversary $\mathcal{A}$ has at most a negligible advantage in winning the following game.

Corruptions The adversary sends a set of uncorrupted and corrupted clients to the challenger, I and $\bar{I}$, respectively.

Setup The challenger $\mathcal{B}$ picks a bit $b \xleftarrow{R} \{0, 1\}$, and sends the public parameters pp along with the user keys of the corrupted clients $\{\mathsf{usk}_i\}_{i \in \bar{I}}$ to the adversary $\mathcal{A}$.

Query 1 The adversary may query the challenger for the encryption of sets $\mathcal{S}_i$ for uncorrupted clients $i \in I$ associated with an ID that has not been used before. For each uncorrupted client $i \in I$, the challenger returns the encrypted set $\mathsf{ct}_{\mathsf{ID},i} \leftarrow \mathsf{Encrypt}(\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i)$.

Challenge The adversary sends two equally sized sets $\mathcal{S}^*_{i,0}$, $\mathcal{S}^*_{i,1}$, $|\mathcal{S}^*_{i,0}| = |\mathcal{S}^*_{i,1}|$, for every uncorrupted client $i \in I$ together with an ID^* that has not been used before. The challenger checks if the challenge is allowed by checking if $f(\{\mathcal{S}^*_{i,0}\}_{i \in I}, \cdot) = f(\{\mathcal{S}^*_{i,1}\}_{i \in I}, \cdot)$. If this is not the case the challenger aborts the game. Otherwise, it returns the encrypted sets $\mathsf{Encrypt}(\mathsf{usk}_i, \mathsf{ID}^*, \mathcal{S}^*_{i,b})$ for every uncorrupted client $i \in I$.

Query 2 Identical to Query 1.

Guess The adversary outputs its guess b' for the challenger's bit b.

Note that by definition, the ciphertext does not need to hide the set size. This is similar to the semantic security notion where the ciphertext does not need to hide the plaintext size. If this is undesirable, fixed-sized sets can easily be obtained by adding dummy elements to each set.

5.1 Corruptions in Two-Client Functional Encryption

We observe that *any* single evaluation key 2C-FE scheme can never be secure against corruptions for non-trivial functionalities. To see why this is the case, consider a 2C-FE scheme for the functionality $f(x, y)$. Assume, without loss of generality, that the adversary corrupts the client which determines the input y. By definition of the game for adaptive IND-security of MC-FE, the adversary submits two values x_0 and x_1 to the challenger. For the challenge inputs to be allowed, it is required that $f(x_0, \cdot) = f(x_1, \cdot)$, *i.e.*, we require $f_{x_0}(y) = f_{x_1}(y)$ *for all possible* y. So, unless f is a constant function in y, we have to require that $x_0 = x_1$, for which it is trivial to see that the challenge will be indistinguishable.

Generalizing the result, we see that in an MC-FE scheme for n clients, at least two clients need to remain uncorrupted. Phrased differently, this means that for MC-FE with n clients, we can allow for at most $n - 2$ corruptions.

6 Two-Client Constructions for Set Intersections

We propose several 2C-FE schemes for various set operations: computing the cardinality of the set intersection, computing the set intersection itself, computing the set intersection with data transfer or projection, and computing the set intersection only if a threshold is reached. We discuss constructions supporting more than two clients in Section 7.

6.1 Two-Client Set Intersection Cardinality

To compute the cardinality of a set intersection from two clients, we can suffice with a simple scheme using a pseudorandom function (PRF) (*e.g.*, see [Sma16, § 11.2]). The two clients encrypt each set element individually using a PRF under the same key. Since a PRF has a deterministic output, the evaluator can now use *any* algorithm for determining the cardinality of the intersection, even algorithms that only operate on plaintext data (*e.g.*, see [DK11] for an overview).

Setup(1^λ) $\rightarrow$ $(\mathsf{pp}, \mathsf{usk}_1, \mathsf{usk}_2)$. Let $\Phi = \{\phi_\kappa\}$ be a PRF ensemble for functions $\phi_\kappa \colon \mathcal{ID} \times \{0,1\}^* \rightarrow \{0,1\}^{\geq\lambda}$. Pick a PRF ϕ_{msk}. The public parameters are $\mathsf{pp} = (\Phi)$ and the clients' keys $\mathsf{usk}_1 = \mathsf{usk}_2 = (\phi_{\mathsf{msk}})$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) $\rightarrow$ $\mathsf{ct}_{\mathsf{ID},i}$. For a client i to encrypt its set $\mathcal{S}_i$ for an identifier $\mathsf{ID} \in \mathcal{ID}$, the client computes the PRF for each set element $x_j \in \mathcal{S}_i$. It outputs the set $\mathsf{ct}_{\mathsf{ID},i} = \{\, \phi_{\mathsf{msk}}(\mathsf{ID}, x_j) \mid x_j \in \mathcal{S}_i \,\}$.

Eval($\mathsf{ct}_{\mathsf{ID},1}, \mathsf{ct}_{\mathsf{ID},2}$) $\rightarrow$ $|\mathcal{S}_1 \cap \mathcal{S}_2|$. To evaluate the cardinality of the set intersection, output $|\mathsf{ct}_{\mathsf{ID},1} \cap \mathsf{ct}_{\mathsf{ID},2}|$.

We can use a block cipher, keyed-hash function, hash-based message authentication code, or a similar function as the PRF.

Theorem 1. *The two-client set intersection cardinality scheme defined above is secure under the assumption that the* PRF *is indistinguishable from a random function.*

Proof. This directly follows from the security of the PRF. Note that the evaluator only learns whether two set elements $x_{1,j} \in \mathcal{S}_1$ and $x_{2,j'} \in \mathcal{S}_2$ equal or not. Nothing else is revealed about the set elements $x_{1,j}$ and $x_{2,j'}$. $\square$

6.2 Two-Client Set Intersection

In case of two-client set intersection, we need not only to determine whether two encrypted set elements are the same, but also learn the plaintext set element if they are the same. We achieve this by adapting our construction for two-client set intersection cardinality with a combination of convergent encryption [DAB+02] (*cf.* message-locked encryption [BKR13]) and secret sharing: We encrypt the set element under a key derived from the message itself and secret share the encryption key. If both clients encrypted the same message, the decryption key can be recovered from the secret shares and the ciphertext can be decrypted. To encrypt the set element itself, we use an authenticated encryption (AE) scheme [BN00].

Setup(1^λ) $\to$ ($\mathsf{pp}, \mathsf{usk}_1, \mathsf{usk}_2$). Let $\langle g \rangle = \mathbb{G}$ be a group of prime order p and let $\Phi = \{\phi_\kappa\}$ be a PRF ensemble for functions $\phi_\kappa \colon \mathcal{ID} \times \{0,1\}^* \to \mathbb{G}$ and AE an AE scheme. Define a mapping from the group to the key space of the AE scheme, $H \colon \mathbb{G} \to \mathcal{K}_{\text{AE}}$. Pick a PRF ϕ_{msk} and pick $\sigma_1 \xleftarrow{R} \mathbb{Z}_p$ to set $\sigma_2 = 1 - \sigma_1 \pmod p$. The public parameters are $\mathsf{pp} = (\mathbb{G}, \Phi, H, \text{AE})$ and the clients' keys $\mathsf{usk}_1 = (\phi_{\mathsf{msk}}, \sigma_1)$ and $\mathsf{usk}_2 = (\phi_{\mathsf{msk}}, \sigma_2)$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) $\to$ $\mathsf{ct}_{\mathsf{ID},i}$. For a client i to encrypt its set $\mathcal{S}_i$ for an identifier $\mathsf{ID} \in \mathcal{ID}$, the client computes the PRF for each set element $x_j \in \mathcal{S}_i$. It outputs the set of tuples $\{(\mathsf{ct}_{\mathsf{ID},i,j,1}, \mathsf{ct}_{\mathsf{ID},i,j,2})\}_{1 \le j \le |\mathcal{S}_i|}$,

$$\mathsf{ct}_{\mathsf{ID},i} = \left\{ \left(k_{\mathsf{ID},j}^{\sigma_i}, \text{AE}.\mathsf{Enc}_{H(k_{\mathsf{ID},j})}(x_j)\right) \mid k_{\mathsf{ID},j} = \phi_{\mathsf{msk}}(\mathsf{ID}, x_j), x_j \in \mathcal{S}_i \right\}.$$

Eval($\mathsf{ct}_{\mathsf{ID},1}, \mathsf{ct}_{\mathsf{ID},2}$) $\to$ $\mathcal{S}_1 \cap \mathcal{S}_2$. For all $\mathsf{ct}_{\mathsf{ID},1,j,2} = \mathsf{ct}_{\mathsf{ID},2,k,2}$ (and hence $x = x_{1,j} = x_{2,k}$), determine

$$\begin{aligned} k_{\mathsf{ID},x} &= \mathsf{ct}_{\mathsf{ID},1,j,1} \cdot \mathsf{ct}_{\mathsf{ID},2,k,1} \\ &= \phi_{\mathsf{msk}}(\mathsf{ID}, x)^{\sigma_1} \cdot \phi_{\mathsf{msk}}(\mathsf{ID}, x)^{\sigma_2} = \phi_{\mathsf{msk}}(\mathsf{ID}, x)^{\sigma_1 + \sigma_2} = \phi_{\mathsf{msk}}(\mathsf{ID}, x), \end{aligned}$$

to decrypt $\mathsf{ct}_{\mathsf{ID},i,j,2}$ using $\text{AE}.\mathsf{Dec}_{H(k_{\mathsf{ID},x})}(\mathsf{ct}_{\mathsf{ID},i,j,2})$ for $i = 1$ or, equivalently, $i = 2$.

Theorem 2. *The two-client set intersection scheme defined above is secure under the decisional Diffie–Hellman (*DDH*) assumption, a secure* PRF*, and a secure* AE *scheme.*

Proof. We construct an algorithm that is able to break the DDH problem if a p.p.t. adversary $\mathcal{A}$ has a non-negligible advantage in winning the game.

Setup The challenger $\mathcal{B}$ receives the DDH tuple (g, g^a, g^b, T) from the group $\mathbb{G}$ of prime order p. It defines a PRF ensemble $\Phi = \{\phi_\kappa\}$ and mapping $H \colon \mathbb{G} \to \mathcal{K}_{\text{AE}}$ according to the scheme. The public parameters $\mathsf{pp} = (\mathbb{G}, \Phi, H, \text{AE})$ are sent to the adversary. The challenger indirectly sets $\sigma_1 = a$ and $\sigma_2 = 1 - a$, *i.e.*, $g^{\sigma_1} = g^a$ and $g^{\sigma_2} = g \cdot (g^a)^{-1}$.

Query Upon receiving an allowed encryption query for $(i, \mathsf{ID}, \mathcal{S})$, the challenger encrypts the elements of the set $\mathcal{S}$ as follows. It models the PRF as follows: On input (ID, x_j), output $g^{r_{\mathsf{ID},x_j}}$, where, if the input has not been queried before, $r_{\mathsf{ID},x_j} \xleftarrow{R} \mathbb{Z}_p$. The challenger encrypts an element $x_j \in \mathcal{S}$ as

$$\mathsf{ct}_{\mathsf{ID},i,j} = \begin{cases} \left((g^a)^{r_{\mathsf{ID},x_j}}, \text{AE}.\mathsf{Enc}_k(x_j)\right) & \text{if } i = 1; \\ \left((g \cdot (g^a)^{-1})^{r_{\mathsf{ID},x_j}}, \text{AE}.\mathsf{Enc}_k(x_j)\right) & \text{if } i = 2, \end{cases} \quad \text{where } k = H(g^{r_{\mathsf{ID},x_j}}).$$

It outputs the encrypted set $\mathsf{ct}_{\mathsf{ID},i}$ to the adversary.

Challenge An allowed challenge request from the adversary for the sets $\mathcal{S}^*_{1,0}$, $\mathcal{S}^*_{1,1}$, $\mathcal{S}^*_{2,0}$, and $\mathcal{S}^*_{2,1}$ with identifier ID^*, is answered by the challenger by sending the encrypted sets $\mathcal{S}^*_{1,b}$ and $\mathcal{S}^*_{2,b}$ back to the adversary. An element $x_j \notin (\mathcal{S}_{1,b} \cap \mathcal{S}_{2,b})$ is encrypted as

$$\mathsf{ct}_{\mathsf{ID},i,j} = \begin{cases} \left(T^{r_{\mathsf{ID}^*,x_j}}, \text{AE}.\mathsf{Enc}_k(x_j)\right) & \text{if } i = 1; \\ \left((g^b \cdot T^{-1})^{r_{\mathsf{ID}^*,x_j}}, \text{AE}.\mathsf{Enc}_k(x_j)\right) & \text{if } i = 2, \end{cases} \quad \text{where } k = H\left((g^b)^{r_{\mathsf{ID},x_j}}\right).$$

Note that this indirectly sets the output of the PRF to $g^{br_{\mathsf{ID}^*,x_j}}$ for $x_j \notin (\mathcal{S}_{1,b} \cap \mathcal{S}_{2,b})$. The elements $x_j \in (\mathcal{S}_{1,b} \cap \mathcal{S}_{2,b})$ are encrypted as in the query phase.

If the adversary $\mathcal{A}$ outputs a correct guess $b' = b$, the challenger outputs the guess that $T = g^{ab}$, otherwise, it outputs its guess $T \in_R \mathbb{G}$. □

6.3 Two-Client Set Intersection with Data Transfer or Projection

The two-client set intersection scheme described above can be extended into a two-client set intersection scheme with data transfer (analogous to PSI *with data transfer* [DT10; JL10]). Instead of only encrypting the set element x_j itself, $\mathsf{ct}_{\mathsf{ID},i,j,2} = \text{AE}.\mathsf{Enc}_k(x_j)$, we can also choose to encrypt both the element itself and the data associated to the set element $\rho(x_j)$. The security of the scheme is the same as before since we rely on the security of the AE scheme.

Moreover, the proposed scheme also allows for a two-client set intersection projection scheme (analogous to PSI *with projection* [CFS+17]). We construct such a scheme by encrypting only the associated data $\rho(x_j)$, $\mathsf{ct}_{\mathsf{ID},i,j,2} = \text{AE}.\mathsf{Enc}_k(\rho(x_j))$, not the set element x_j itself. Security follows from the fact that the AE decryption key $k = H(\phi_{\mathsf{msk}}(\mathsf{ID}, x_j))$ does not reveal any information about the set element x_j, assuming the security of the used PRF. However, the evaluator does learn that the projections of both clients correspond to the same set element.

6.4 Two-Client Threshold Set Intersection

To allow the evaluator to learn the cardinality of the intersection, but only the set elements in the intersection if the clients have at least t set elements in common, we propose a two-client threshold set intersection scheme. We achieve this by encrypting the share of the decryption key for the AE ciphertext $k_{\mathsf{ID},j}^{\sigma_i}$ using another encryption key. This newly added encryption key can only be obtained by the evaluator if the clients have at least t set elements in common.

Although the construction is based on the previous scheme, the precise construction is quite technical. We therefore state the complete scheme below.

Setup($1^\lambda, t$) $\rightarrow$ ($\mathsf{pp}, \mathsf{usk}_1, \mathsf{usk}_2$). Let AE an AE scheme and $\langle g \rangle = \mathbb{G}$ be a group and $\mathbb{F}_p$ be a field, both of prime order p. Let $\Phi = \{\phi_\kappa\}$ and $\Psi = \{\psi_\kappa\}$ be PRF ensembles for functions $\phi_\kappa \colon \mathcal{ID} \times \{0,1\}^* \rightarrow \mathbb{G}$ and $\psi_\kappa \colon \mathcal{ID} \times \{0,1\}^* \rightarrow \mathbb{F}_p$, respectively. Define a mapping from the group to the key space of the AE scheme, $H \colon \mathbb{G} \rightarrow \mathcal{K}_{\text{AE}}$. Pick three PRFs $\phi \in \Phi$, $\psi_1, \psi_2 \in \Psi$ and $\sigma_1 \xleftarrow{R} \mathbb{Z}_p, \rho_1 \xleftarrow{R} \mathbb{Z}_{p-1}$, setting $\sigma_2 = 1 - \sigma_1 \pmod p$ and $\rho_2 = 1 - \rho_1 \pmod{p-1}$.

The public parameters are $\mathsf{pp} = (\mathbb{G}, \Phi, \Psi, H, \text{AE}, t)$ and the clients' keys $\mathsf{usk}_1 = (\phi, \psi_1, \psi_2, \sigma_1, \rho_1)$ and $\mathsf{usk}_2 = (\phi, \psi_1, \psi_2, \sigma_2, \rho_2)$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) $\rightarrow$ $\mathsf{ct}_{\mathsf{ID},i}$. For a client i to encrypt its set $\mathcal{S}_i$ for an identifier $\mathsf{ID} \in \mathcal{ID}$, the client computes the PRF for each set element $x_j \in \mathcal{S}_i$. It defines the $(t-1)$th degree polynomial f_{ID} by setting the coefficients $c_i = \psi_2(\mathsf{ID}, i)$, for $0 \leq i < t$, to obtain the polynomial $f_{\mathsf{ID}}(x) = c_{t-1}x^{t-1} + \cdots + c_1 x + c_0$.

The client outputs the set

$$\mathsf{ct}_{\mathsf{ID},i} = \Big\{ \big(k_{\mathsf{ID},j,2}, f(k_{\mathsf{ID},j,2})^{\rho_i}, \text{AE}.\mathsf{Enc}_{H(c_0)}(k_{\mathsf{ID},j,1}^{\sigma_i}), \text{AE}.\mathsf{Enc}_{H(k_{\mathsf{ID},j,1})}(x_j)\big) \\ \mid k_{\mathsf{ID},j,1} = \phi(\mathsf{ID}, x_j), k_{\mathsf{ID},j,2} = \psi_1(\mathsf{ID}, x_j), x_j \in \mathcal{S}_i \Big\} .$$

Eval($\mathsf{ct}_{\mathsf{ID},1}, \mathsf{ct}_{\mathsf{ID},2}$) $\rightarrow$ ($|\mathcal{S}_1 \cap \mathcal{S}_2|, \{ x_j \mid x_j \in \mathcal{S}_1 \cap \mathcal{S}_2, |\mathcal{S}_1 \cap \mathcal{S}_2| \geq t \}$). The evaluation algorithm consists of two stages; the second stage is only executed if $|\mathcal{S}_1 \cap \mathcal{S}_2| \geq t$.

1. To determine the cardinality of the set intersection $|\mathcal{S}_1 \cap \mathcal{S}_2|$, the evaluator counts the number of times a value $k_{\mathsf{ID},j,2}$ occurs both in $\mathsf{ct}_{\mathsf{ID},1}$ and $\mathsf{ct}_{\mathsf{ID},2}$.
2. If $|\mathcal{S}_1 \cap \mathcal{S}_2| \geq t$, the evaluator uses Lagrange interpolation to compute the value $c_0 = f(0)$. It can do so by taking t distinct tuples $\big(k_{\mathsf{ID},j,2}, f(k_{\mathsf{ID},j,2})\big)$, where $f(k_{\mathsf{ID},j,2}) = f(k_{\mathsf{ID},j,2})^{\rho_1} \cdot f(k_{\mathsf{ID},j,2})^{\rho_2}$. Now, when the secret c_0 has been recovered from the shares, the evaluator can use it to decrypt the values $\text{AE}.\mathsf{Enc}_{H(c_0)}(k_{\mathsf{ID},j,1}^{\sigma_i})$. So, the evaluator obtains $k_{\mathsf{ID},j,1}^{\sigma_i}$ for *every* set element in $x_j \in \mathcal{S}_i$ if $|\mathcal{S}_1 \cap \mathcal{S}_2| \geq t$. Observe that for the elements in the intersection, the evaluator has both $k_{\mathsf{ID},j,1}^{\sigma_1}$ and $k_{\mathsf{ID},j,1}^{\sigma_2}$, and can compute $k_{\mathsf{ID},j,1} = k_{\mathsf{ID},j,1}^{\sigma_1} \cdot k_{\mathsf{ID},j,1}^{\sigma_2}$. Finally, using $H(k_{\mathsf{ID},j,1})$, it can decrypt $\text{AE}.\mathsf{Enc}_{H(k_{\mathsf{ID},j,1})}(x_j)$ to obtain $x_j \in \mathcal{S}_1 \cap \mathcal{S}_2$.

Since the construction above builds upon the set intersection scheme, which can be modified into a set intersection with data transfer scheme or a set intersection with projection scheme, we similarly obtain both threshold set intersection with data transfer and projection.

Theorem 3. *The two-client threshold set intersection scheme defined above is secure under the* DDH *assumption, a secure* PRF*, and a secure* AE *scheme.*

Proof. We only have to prove that the values $k_{\mathsf{ID},j,1}^{\sigma_i}$ can only be obtained if $|\mathcal{S}_1 \cap \mathcal{S}_2| \geq t$, as the rest of the proof directly follows from Theorem 2. Since the values $k_{\mathsf{ID},j,1}^{\sigma_i}$ are encrypted using an AE scheme using the key $H(c_0)$, the values are only know to the evaluator if it has the key $H(c_0)$ (under the assumption of a secure AE scheme). The fact that c_0 (and hence $H(c_0)$) can only be obtained from the secret shares follows from the information-theoretic security of SSSS if a random polynomial f_{ID} was used. Note that the $(t-1)$th degree polynomial is random under the assumption of a secure PRF. Finally, using a similar argument as in Theorem 2, we can show that, under the DDH assumption, $f(k_{\mathsf{ID},j,2})^{\rho_1}$ or $f(k_{\mathsf{ID},j,2})^{\rho_2}$ does not reveal any information about $f(k_{\mathsf{ID},j,2})$ if $f(k_{\mathsf{ID},j,2})^{\rho_2}$ or $f(k_{\mathsf{ID},j,2})^{\rho_1}$, respectively, is unknown. □

7 Multi-client Constructions for Set Intersections

While the 2C-FE constructions from Section 6 could be used in a multi-client case, this would leak information about each pair of sets. For the same reason, deterministic encryption cannot be used in secure MC-FE constructions, which makes it much harder to develop efficient MC-FE schemes.

7.1 Multi-client Set Intersection Cardinality

We construct an MC-FE scheme for testing the set intersection using only a hash function and secret sharing. The proposed scheme incurs no additional leakage and is proven adaptive IND-secure. While our scheme has an evaluation algorithm which does not rely on heavy cryptographic machinery and runs in polynomial time (for a fixed number of clients n), it is not very efficient. The running time of the evaluation algorithm grows in the product of the cardinality of the individual clients' set size. However, for relatively small sets or a small number of clients this scheme might still be efficient enough to use in practice.

Setup($1^\lambda, n$) $\to$ ($\mathsf{pp}, \mathsf{usk}_1, \ldots, \mathsf{usk}_n$). Let $\langle g \rangle = \mathbb{G}$ be a group of prime order p and let $H\colon \mathcal{ID} \times \{0,1\}^* \to \mathbb{G}$ be a hash function. Create random shares of 0 by picking $\sigma_i \overset{R}{\leftarrow} \mathbb{Z}_p$, for all $2 \leq i \leq n$, and setting $\sigma_1 = -\sum_{i=2}^{n} \sigma_i \pmod{p}$. The public parameters are $\mathsf{pp} = (H)$ and the clients' keys $\mathsf{usk}_i = (\sigma_i)$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) $\to \mathsf{ct}_{\mathsf{ID},i}$. For a client i to encrypt its set $\mathcal{S}_i$ using an identifier $\mathsf{ID} \in \mathcal{ID}$, the client encrypts each set element $x_j \in \mathcal{S}_i$ individually. It outputs the set $\mathsf{ct}_{\mathsf{ID},i} = \{\, H(\mathsf{ID}, x_j)^{\sigma_i} \mid x_j \in \mathcal{S}_i \,\}$.

Eval($\mathsf{ct}_{\mathsf{ID},1}, \ldots, \mathsf{ct}_{\mathsf{ID},n}$) $\to |\bigcap_{i=1}^{n} \mathcal{S}_i|$. For each n-tuple $(c_{\mathsf{ID},1}, \ldots, c_{\mathsf{ID},n}) \in \mathsf{ct}_{\mathsf{ID},1} \times \cdots \times \mathsf{ct}_{\mathsf{ID},n}$, the evaluator evaluates $\prod_{i=1}^{n} c_{\mathsf{ID},i} \stackrel{?}{=} 1$. The evaluator outputs the count for the number of times the expression above evaluates to TRUE.

We will prove the construction secure under selective corruptions, but we note that it is also possible to achieve a proof under dynamic corruptions (although less tight) by adapting the proofs from [SCR+11].

Theorem 4. *The improved multi-client set intersection cardinality scheme defined above is secure up to $(n-2)$ corruptions under the* DDH *assumption in the random oracle model (*ROM*).*

Proof. Let $\mathcal{A}$ be a p.p.t. adversary playing the adaptive IND-security game for MC-FE. We will show how to use $\mathcal{A}$ as a distinguisher for a DDH tuple, winning with a non-negligible advantage if $\mathcal{A}$ has a non-negligible advantage in winning the security game.

Random Oracle On input of a tuple (ID, x_j) the oracle checks if it has answered the query before. If not, it picks a value $\beta_{\mathsf{ID},x_j} \xleftarrow{R} \mathbb{Z}_p$. Next, the challenger $\mathcal{B}$ guesses whether the query is for the challenge ID. If so, the oracle outputs $(g^b)^{\beta_{\mathsf{ID},x_j}}$, otherwise, it outputs $g^{\beta_{\mathsf{ID},x_j}}$. If the guess turns out to be wrong later, $\mathcal{B}$ can simply abort the game.

Corruptions The adversary $\mathcal{A}$ announces the set of uncorrupted and corrupted clients, I and $\bar{I}$, respectively.

Setup For $i \in \bar{I}$, the challenger $\mathcal{B}$ picks $\sigma_i \xleftarrow{R} \mathbb{Z}_p$ and sends the values to the adversary $\mathcal{A}$. Let $i' \in I$, for $i \in I \setminus \{i'\}$, $\mathcal{B}$ indirectly sets $\sigma_i = a \cdot \alpha_i$, where $\alpha_i \xleftarrow{R} \mathbb{Z}_p$, by setting $g^{\sigma_i} = (g^a)^{\alpha_i}$. For i', it indirectly sets $\sigma_{i'} = -\sum_{i \neq i'} \sigma_i$,

$$g^{\sigma_{i'}} = \prod_{i \in \bar{I}} g^{-\sigma_i} \cdot \prod_{i \in I, i \neq i'} (g^a)^{-\alpha_i}.$$

Query To answer an encryption query $\mathcal{S}_i$ for an uncorrupted client $i \in I$, the challenger uses the oracle to obtain $\{\, \beta_{\mathsf{ID},x_j} \mid x_j \in \mathcal{S}_i \,\}$ and construct the ciphertext as $\mathsf{ct}_{\mathsf{ID},i} = \left\{\, (g^{\sigma_i})^{\beta_{\mathsf{ID},x_j}} \mid x_j \in \mathcal{S}_i \,\right\}$.

Challenge Upon receiving the challenge sets $\{\, (\mathcal{S}^*_{i,0}, \mathcal{S}^*_{i,1}) \mid i \in I \,\}$ and an ID^* from the adversary, the challenger picks $b \xleftarrow{R} \{0,1\}$. The challenger returns the ciphertexts

$$\mathsf{ct}_{\mathsf{ID}^*,i'} = \left\{ \prod_{i \in \bar{I}} (g^b)^{-\sigma_i \cdot \beta_{\mathsf{ID}^*,x_j}} \cdot \prod_{i \in I, i \neq i'} T^{-\alpha_i \cdot \beta_{\mathsf{ID}^*,x_j}} \mid x_j \in \mathcal{S}^*_{i,b} \right\} \qquad \text{and}$$

$$\mathsf{ct}_{\mathsf{ID}^*,i} = \left\{ T^{\alpha_i \beta_{\mathsf{ID}^*,x_j}} \mid x_j \in \mathcal{S}^*_{i,b} \right\} \qquad \text{for } i \neq i'.$$

Note that if $T = g^{ab}$, the ciphertext is distributed properly according the scheme. If $T \in_R \mathbb{G}$, the challenger returns a ciphertext of a randomly distributed set element. So, the challenger $\mathcal{B}$ guesses that $T = g^{ab}$ if $\mathcal{A}$ correctly guessed $b' = b$ and otherwise, $\mathcal{B}$ guesses that $T \in_R \mathbb{G}$. □

We remark that while the security of the two-client schemes could be proven in the standard model, our multi-client constructions can only be proven in the ROM. The difference in the constructions is that in the two-client case, no corruptions are taken place, and thus we can use a programmable PRF instead of a programmable random oracle.

7.2 Efficient Multi-client Set Intersection Cardinality

A drawback of the multi-client set intersection cardinality scheme might be that the computational complexity for the evaluator grows quickly in the total number of set elements (*i.e.*, $\prod_{i=1}^{n} |\mathcal{S}_i|$). To address this problem, we propose an alternative scheme using Bloom filters. In this scheme, we first combine the Bloom filter representation of every client's set in the encrypted domain, resulting in an encrypted Bloom filter representing the intersection of all clients' sets. Next, the evaluator uses the encrypted set elements of *any* client to determine the cardinality of the intersection. This method used by the evaluator to determine the cardinality of the intersection can be seen as computing $|\mathcal{S}_i \cap (\bigcap_{i=1}^{n} \mathcal{S}_i)| = |\bigcap_{i=1}^{n} \mathcal{S}_i|$. The theoretical efficiency of $\mathcal{O}(n + \min_{i=1}^{n} |\mathcal{S}_i|)$ ciphertext operations is much better than the other scheme. However, the proposed scheme is only secure if no corruptions are taking place.

Setup($1^\lambda, n, m, k$) → ($\mathsf{pp}, \mathsf{usk}_1, \ldots, \mathsf{usk}_n$). Let $\langle g \rangle = \mathbb{G}$ be a group of prime order p and let BF be a specification for an (m, k) Bloom filter. Let $\Phi = \{\phi_\kappa\}$ be a PRF ensemble for functions $\phi_\kappa \colon \{0,1\}^* \to \{0,1\}^{\geq \lambda}$ and let $H \colon \mathcal{ID} \times \{0,1\}^* \to \mathbb{G}$ be a hash function. Pick a PRF $\phi \in \Phi$. Additionally, pick for $1 \leq i \leq n$, values $c_i \xleftarrow{R} \mathbb{Z}_p$ and define the n-degree polynomial $f(x) = c_n x^n + \cdots + c_1 x$ over the field $\mathbb{F}_p$. The public parameters are $\mathsf{pp} = (\mathsf{BF}, \Phi, H)$ and the clients' secret keys are $\mathsf{usk}_i = \big(\phi, f(i), f(n+i)\big)$ for $1 \leq i \leq n$. Note that every client receives the same PRF ϕ, but different secret shares $f(i)$ and $f(n+i)$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) → ($\mathsf{ct}_{\mathsf{ID},i,\mathsf{bs}_\mathcal{S}}, \mathsf{ct}_{\mathsf{ID},i,\mathcal{S}}$). First, the client initializes the Bloom filter to obtain $\mathsf{bs}_\mathcal{S}$. Next, it adds its encrypted set elements $\{\, \phi(x_j) \mid x_j \in \mathcal{S}_i \,\}$ to the Bloom filter. For each $1 \leq \ell \leq m$, the client sets $r_{i,\ell} \xleftarrow{R} \mathbb{Z}_p$, if $\mathsf{bs}_{\mathcal{S}}[\ell] = 0$, and $r_{i,\ell} = 0$, otherwise. The client encrypts the Bloom filter for $\mathsf{bs}_{\mathcal{S}}$ as the ordered set

$$\mathsf{ct}_{\mathsf{bs}_{\mathcal{S}}} = \left\{ H(\mathsf{ID}, \ell)^{f(i)} \cdot g^{r_{i,\ell}} \mid 1 \leq \ell \leq m \right\}.$$

Additionally, the client initializes a new bit string bs_j for every set element $x_j \in \mathcal{S}_i$. It encrypts each element x_j and adds $\phi(x_j)$ to the Bloom filter for bs_j. Let t_j denote the Hamming weight (*i.e.*, the number of 1s) of the resulting bit string bs_j. For the resulting bit string bs_j pick $r_{i,j,\ell} \xleftarrow{R} \mathbb{Z}_p$ for $1 \leq \ell \leq m$. Additionally, set $\rho_{i,j,\ell} \xleftarrow{R} \mathbb{Z}_p$ if $\mathsf{bs}_j[\ell] = 0$, and $\rho_{i,j,\ell} = t_j \cdot r_{i,j,\ell}$, otherwise. It encrypts the Bloom filter for bs_j as

$$\mathsf{ct}_{\mathsf{bs}_j} = \left(\left\{ H(\mathsf{ID}, \ell)^{f(n+i)} \cdot g^{\rho_{i,j,\ell}}, g^{r_{i,j,\ell}} \mid 1 \leq \ell \leq m \right\} \right).$$

Finally, the client outputs the ciphertext $\left(\mathsf{ct}_{\mathsf{bs}_{\mathcal{S}}}, \left\{ \mathsf{ct}_{\mathsf{bs}_j} \mid x_j \in \mathcal{S}_i \right\}\right)$.

Eval($\mathsf{ct}_{\mathsf{ID},1}, \ldots, \mathsf{ct}_{\mathsf{ID},n}$) $\to |\bigcap_{i=1}^{n} \mathcal{S}_i|$. Since the clients' ciphertext are encryptions of the individual set elements, we can determine a client with the smallest (encrypted) set. Let γ be such a client. Now, for $1 \leq \ell \leq m$, compute the partial Lagrange interpolation

$$a_\ell = \prod_{i=1}^{n} \left(\mathsf{ct}_{\mathsf{ID},i,\mathsf{bs}_{\mathcal{S}}[\ell]}\right)^{\Delta_{\{1,\ldots,n,n+\gamma\},i}}.$$

Set $d = 0$. Next, to determine if an encrypted set element $x_j \in \mathcal{S}_\gamma$ (represented by a tuple $(\mathsf{ct}_{\mathsf{ID},\gamma,\mathsf{bs}_j}, g^{r_{\gamma,j,\ell}}) \in \mathsf{ct}_{\mathsf{ID},\gamma,\mathcal{S}}$) is in the intersection of all sets, check for each $1 \leq \ell \leq m$, if

$$\left(\mathsf{ct}_{\mathsf{ID},\gamma,\mathsf{bs}_j[\ell]}\right)^{\Delta_{\{1,\ldots,n,n+\gamma\},n+\gamma}} \cdot a_\ell \stackrel{?}{=} (g^{r_{\gamma,j,\ell}})^{t_{j,\ell} \cdot \Delta_{\{1,\ldots,n,n+\gamma\},n+\gamma}}$$

for values $1 \leq t_{j,\ell} \leq k$. If the value $t_{j,\ell}$ occurs $t_{j,\ell}$ times for the values $1 \leq \ell \leq m$, increase the value d by one.

After all encrypted set element $x_j \in \mathcal{S}_\gamma$ have been checked, output the cardinality of the set intersection d.

Correctness To see that the above defined scheme is correct, observe that if a set element $x_j \in \mathcal{S}_i$ is in the intersection of all clients' sets, the values $r_{i,j,\ell}$ equal 0 for the same values of ℓ in the encrypted Bloom filters $\mathsf{ct}_{\mathsf{ID},i,\mathsf{bs}_{\mathcal{S}}}$. Hence, by using the Lagrange interpolation on these elements (corresponding to a_ℓ) together with an encrypted Bloom filter for a single set element $x_j \in \mathcal{S}_\gamma$ (corresponding to $\mathsf{ct}_{\mathsf{ID},\gamma,\mathsf{bs}_j}$), we obtain $H(\mathsf{ID}, \ell)^{f(0)} \cdot g^{r_{i,j,\ell} \cdot \Delta_{\{1,\ldots,n,n+\gamma\},n+\gamma}} = g^{r_{i,j,\ell} \cdot \Delta_{\{1,\ldots,n,n+\gamma\},n+\gamma}}$. Now, note that we set $\rho_{i,j,\ell} = t_j \cdot r_{i,j,\ell}$ if the bit string value $\mathsf{bs}_j[\ell] = 1$. So, if exactly t_j bit string values in the set intersection are set to 1, we know that the element is a member of the set intersection.

Theorem 5. *The improved multi-client set intersection cardinality scheme defined above is secure without corruptions under the* DDH *assumption in the* ROM.

Proof. We construct an algorithm that is able to break the DDH problem if a p.p.t. adversary $\mathcal{A}$ has a non-negligible advantage in winning the game.

Random Oracle On input of a tuple (ID, ℓ) the oracle checks if it has answered the query before. If not, it picks a value $\beta_{\mathsf{ID},\ell} \xleftarrow{R} \mathbb{Z}_p$. Next, the challenger $\mathcal{B}$ guesses whether the query is for the challenge ID. If so, the oracle outputs $(g^b)^{\beta_{\mathsf{ID},\ell}}$, otherwise, it outputs $g^{\beta_{\mathsf{ID},\ell}}$. If the guess turns out to be wrong later, $\mathcal{B}$ can simply abort the game.

Setup The challenger $\mathcal{B}$ receives the DDH tuple (g, g^a, g^b, T) from the group $\mathbb{G}$ of prime order p. It defines a PRF ensemble $\Phi = \{\phi_\kappa\}$ and the Bloom filter BF according to the scheme. Pick for $1 \le i \le n$, values $c_i \xleftarrow{R} \mathbb{Z}_p$ and define the n-degree polynomial $f'(x) = c_n x^n + \cdots + c_1 x$ over the field $\mathbb{F}_p$. The challenger uses $f(x) = a \cdot f'(x)$ to indirectly define the secret shares. Note that this still allows $\mathcal{B}$ to compute $g^{f(x)} = (g^a)^{f'(x)}$ for all values of x.

Query To answer an encryption query $\mathcal{S}_i$ for a client i, the challenger uses the oracle to obtain $\{ \beta_{\mathsf{ID},\ell} \mid x_j \in \mathcal{S}_i \}$ and construct the ciphertext as in the scheme, but using $H(\mathsf{ID}, \ell)^{f(x)} = (g^a)^{\beta_{\mathsf{ID},\ell} f'(x)}$.

Challenge Upon receiving the challenge sets $(\mathcal{S}^*_{i,0}, \mathcal{S}^*_{i,1})$ for $1 \le i \le n$ and an ID^* from the adversary, the challenger picks $b \xleftarrow{R} \{0, 1\}$. The challenger returns the encryptions of the sets $\mathcal{S}^*_{i,b}$ using the scheme's encrypt algorithm, but replacing $H(\mathsf{ID}^*, \ell)^{f(x)}$ by $T^{\beta_{\mathsf{ID}^*,\ell} f'(x)}$. Note that if $T = g^{ab}$, the ciphertext is distributed properly according the scheme. If $T \in_R \mathbb{G}$, the challenger returns a ciphertext of a randomly distributed set element. So, the challenger $\mathcal{B}$ guesses that $T = g^{ab}$ if $\mathcal{A}$ correctly guessed $b' = b$ and otherwise, $\mathcal{B}$ guesses that $T \in_R \mathbb{G}$. □

To construct efficient multi-client functional encryption schemes for set operations that resist corruptions, we need to be able to check the membership of an encrypted set element against the encrypted intersection of the clients' sets. The above construction fails to be secure against corruptions as it (partially) reveals the individual bits in the bit string of a Bloom filter for a set element, *i.e.*, the adversary learns (part of) the bit string representation of the set element.

7.3 Multi-client Set Intersection

The set intersection can be computed using a notion similar to non-interactive distributed encryption (DE) schemes [GH11; LHK14]. A DE scheme is characterized by two distinctive features. Firstly, we have that multiple clients can encrypt a plaintext under their own secret key. Secondly, if enough clients have encrypted the same plaintext, anyone can recover this plaintext from the clients' ciphertexts.

We construct an MC-FE scheme for set intersection from a DE scheme.

Setup($1^\lambda, n$) $\to$ ($\mathsf{pp}, \mathsf{usk}_1, \ldots, \mathsf{usk}_n$). Run DE.$\mathsf{Gen}(1^\lambda, n, n)$ to generate an n-OUT-OF-n DE scheme defined by pp and obtain the encryption keys $(\mathsf{usk}_1, \ldots, \mathsf{usk}_n)$.

Encrypt($\mathsf{usk}_i, \mathsf{ID}, \mathcal{S}_i$) $\to$ $\mathsf{ct}_{\mathsf{ID},i}$. To encrypt the set $\mathcal{S}_i$, encrypt the identifier ID together with each set element $x_j \in \mathcal{S}_i$ individually,

$$\mathsf{ct}_{\mathsf{ID},i} = \{\, \text{DE}.\mathsf{Enc}(\mathsf{usk}_i, \mathsf{ID} \parallel x_j) \mid x_j \in \mathcal{S}_i \,\},$$

where ID has a fixed length (*e.g.*, by applying padding). The algorithm's output is a random ordering of the set $\mathsf{ct}_{\mathsf{ID},i}$.

Eval($\mathsf{ct}_{\mathsf{ID},1}, \ldots, \mathsf{ct}_{\mathsf{ID},n}$) $\to$ $\bigcap_{i=1}^n \mathcal{S}_i$. For each n-tuple $(c_{\mathsf{ID},1}, \ldots, c_{\mathsf{ID},n}) \in \mathsf{ct}_{\mathsf{ID},1} \times \cdots \times \mathsf{ct}_{\mathsf{ID},n}$, the evaluator uses DE.$\mathsf{Comb}(c_{\mathsf{ID},1}, \ldots, c_{\mathsf{ID},n})$ to obtain either the message $\mathsf{ID} \parallel x_j$ or $\perp$. If the message starts with the expected ID, it adds x_j to the initially empty set $\mathcal{R}$.

After evaluating all tuples, the evaluator outputs the set $\mathcal{R}$.

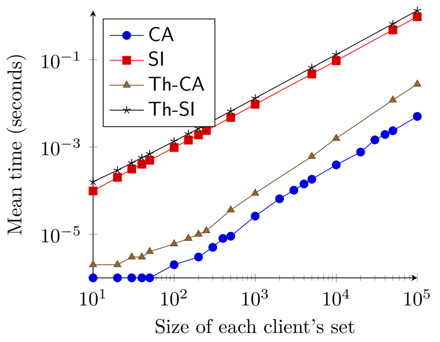
Theorem 6. *The multi-client set intersection scheme defined above is secure under the security of the* DE *scheme.*

Proof. For $b \in \{0, 1\}$, we consider for every set element $x_{j,b} \in \bigcup_{i \in I} \mathcal{S}^*_{i,b}$ two cases:

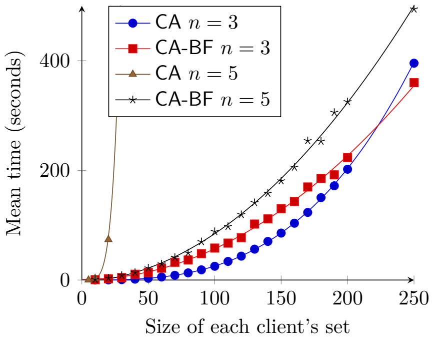
- if $x_{j,b} \in \bigcap_{i \in I} \mathcal{S}^*_{i,b}$, $x_{j,b}$ is also contained in every client i's set $\mathcal{S}^*_{i,1-b}$;
- if $x_{j,b} \notin \bigcap_{i \in I} \mathcal{S}^*_{i,b}$, there is at least one set $\mathcal{S}^*_{k,1-b}$ which does not contain $x_{j,b}$, but an element $x_{j,1-b} \notin \bigcap_{i \in I} \mathcal{S}^*_{i,1-b}$ (and hence $x_{j,1-b} \notin \bigcap_{i \in I} \mathcal{S}^*_{i,b}$) instead.

For the elements x_j satisfying the first case, the adversary does not learn anything about b since for every client i we have that $x_j \in \mathcal{S}^*_{i,b}$ and $x_j \in \mathcal{S}^*_{i,1-b}$, while $|\mathcal{S}^*_{i,b}| = |\mathcal{S}^*_{i,1-b}|$ (remember that the set elements are randomly ordered).

For the elements $x_{j,b}$ satisfying the second case, we claim that the adversary does not learn anything about b by the security of the DE scheme. To see this, note that there exist at least two uncorrupted clients, with at least one client which did not encrypt the plaintext $\mathsf{ID}^* \parallel x_{j,b}$. Observe that the security of the DE scheme gives us that one cannot distinguish an encryption of a plaintext m_0 from an encryption of a plaintext m_1 as long as at most $t-1$ uncorrupted clients have encrypted the same plaintext. Combined with the fact that in our scheme we have set $t = n$ and the fact that we know that at least one uncorrupted client did not encrypt the message $\mathsf{ID}^* \parallel x_{j,b}$ and also that at least one uncorrupted client did not encrypt the message $\mathsf{ID}^* \parallel x_{j,1-b}$, we know that the encryption of the message $\mathsf{ID}^* \parallel x_{j,b}$ is indistinguishable from the encryption of the message $\mathsf{ID}^* \parallel x_{j,1-b}$. $\square$



(a) 2C-FE timings; for the threshold scheme $t = 5$ is used.



(b) MC-FE timings, with interpolation on the domain [0, 250].

Figure 2: Evaluations for determining the cardinality (CA); set intersection (SI); and cardinality (Th-CA) and set intersection (Th-SI) in the threshold scheme.

To improve efficiency, we can combine the above multi-client set intersection scheme with the efficient multi-client set intersection cardinality scheme. The construction for determining the cardinality can be used first to identify which ciphertext elements correspond to set elements that are in the set intersection. Next, we only have to use the evaluation algorithm of the multi-client set intersection scheme on these elements from which we know that they belong to the set intersection.

8 Evaluation

We created proof-of-concept implementations[1] of the proposed 2C-FE schemes and the two MC-FE schemes for determining the cardinality of the intersection. The implementations are done in Python using the Charm library at a 128 bit security level. The evaluations are done on a commodity laptop (i5-4210U@1.7GHz, 8 GB RAM) using only a single core. In Figure 2 we show the time it took to run Eval on encrypted sets of varying sizes. Each client encrypted a set of the same size and had 10% of their set in common with the other clients.

We see that the 2C-FE constructions can be evaluated in under a second, even for sets of 100 thousand elements in size. A lower bound of the timings is given by the 2C-FE cardinality scheme, CA, since it uses the same built-in Python algorithm that is used on plaintext data. The MC-FE constructions are polynomial in the set sizes. We evaluated the Bloom filter (BF) construction with a worst-case false positive rate of 0.001. While it scales linear for fixed Bloom

[1] Available at `https://github.com/CRIPTIM/nipsi`.

filter sizes, the length of the bit strings have to increase linearly for larger sets, resulting in quadratic efficiency of the Eval algorithm.

9 Conclusion

We initiated the study of non-interactive two-client functional encryption (2C-FE) and multi-client functional encryption (MC-FE) schemes for set intersection. We show that very efficient 2C-FE schemes can be constructed for set intersection and related set operations. Additionally, the problem of constructing non-interactive set intersection schemes for three or more clients is addressed by our MC-FE schemes from a theoretical perspective. Finally, we show the practicability of the proposed schemes using proof-of-concept implementations.

Acknowledgments. This work was supported by the Netherlands Organisation for Scientific Research (Nederlandse Organisatie voor Wetenschappelijk Onderzoek, NWO) in the context of the CRIPTIM project.

References

[ABK+19] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. "Decentralizing Inner-Product Functional Encryption." In: *Public-Key Cryptography* (PKC). Ed. by Dongdai Lin and Kazue Sako. Vol. 11443.II. LNCS. Springer International Publishing, 2019, pp. 128–157. DOI: 10.1007/978-3-030-17259-6_5.

[ATD15] Aydin Abadi, Sotirios Terzis, and Changyu Dong. "O-PSI: Delegated Private Set Intersection on Outsourced Datasets." In: *ICT Systems Security and Privacy Protection* (SEC). Ed. by Hannes Federrath and Dieter Gollmann. Vol. 455. IFIPAICT. Springer International Publishing, 2015, pp. 3–17. DOI: 10.1007/978-3-319-18467-8_1.

[ATD16] Aydin Abadi, Sotirios Terzis, and Changyu Dong. "VD-PSI: Verifiable Delegated Private Set Intersection on Outsourced Private Datasets." In: *Financial Cryptography and Data Security* (FC). Ed. by Jens Grossklags and Bart Preneel. Vol. 9603. LNCS. Springer, 2016, pp. 149–168. DOI: 10.1007/978-3-662-54970-4_9.

[BKR13] Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. "Message-Locked Encryption and Secure Deduplication." In: EUROCRYPT. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. LNCS. Springer, 2013, pp. 296–312. DOI: 10.1007/978-3-642-38348-9_18.

[BLR$^+$15] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. “Semantically Secure Order-Revealing Encryption: Multi-input Functional Encryption Without Obfuscation.” In: EUROCRYPT. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9057.II. LNCS. Springer, 2015, pp. 563–594. DOI: 10.1007/978-3-662-46803-6_19.

[BN00] Mihir Bellare and Chanathip Namprempre. “Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm.” In: ASIACRYPT. Ed. by Tatsuaki Okamoto. Vol. 1976. LNCS. Springer, 2000, pp. 531–545. DOI: 10.1007/3-540-44448-3_41.

[CDG$^+$18] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. “Decentralized Multi-Client Functional Encryption for Inner Product.” In: ASIACRYPT. Ed. by Thomas Peyrin and Steven Galbraith. Vol. 11273. LNCS. Springer International Publishing, 2018, pp. 703–732. DOI: 10.1007/978-3-030-03329-3_24.

[CFS$^+$17] Xavier Carpent, Sky Faber, Tomas Sander, and Gene Tsudik. “Private Set Projections & Variants.” In: *Workshop on Privacy in the Electronic Society* (WPES). Ed. by Adam J. Lee. ACM, 2017, pp. 87–98. DOI: 10.1145/3139550.3139554.

[DAB$^+$02] John R. Douceur, Atul Adya, William J. Bolosky, Dan Simon, and Marvin Theimer. “Reclaiming space from duplicate files in a serverless distributed file system.” In: *International Conference on Distributed Computing Systems* (ICDCS). Ed. by Luis E. T. Rodrigues *et al.* IEEE, 2002, pp. 617–624. DOI: 10.1109/ICDCS.2002.1022312.

[DCW13] Changyu Dong, Liqun Chen, and Zikai Wen. “When Private Set Intersection Meets Big Data: An Efficient and Scalable Protocol.” In: *Computer & Communications Security* (CCS). Ed. by Ahmad-Reza Sadeghi *et al.* ACM, 2013, pp. 789–800. DOI: 10.1145/2508859.2516701.

[DK11] Bolin Ding and Arnd Christian König. “Fast Set Intersection in Memory.” In: *Proceedings of the VLDB Endowment* (PVLDB) 4.4 (Jan. 2011). Ed. by H. V. Jagadish and Nick Koudas, pp. 255–266. DOI: 10.14778/1938545.1938550.

[DT10] Emiliano De Cristofaro and Gene Tsudik. “Practical Private Set Intersection Protocols with Linear Complexity.” In: *Financial Cryptography and Data Security* (FC). Ed. by Radu Sion. Vol. 6052. LNCS. Springer, 2010, pp. 143–159. DOI: 10.1007/978-3-642-14577-3_13.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. “Efficient Private Matching and Set Intersection.” In: EUROCRYPT. Ed. by Christian Cachin and Jan L. Camenisch. Vol. 3027. LNCS. Springer, 2004, pp. 1–19. DOI: 10.1007/978-3-540-24676-3_1.

[GGG+14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. “Multi-input Functional Encryption.” In: EUROCRYPT. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Vol. 8441. LNCS. Springer, 2014, pp. 578–602. DOI: `10.1007/978-3-642-55220-5_32`.

[GH11] David Galindo and Jaap-Henk Hoepman. “Non-interactive Distributed Encryption: A New Primitive for Revocable Privacy.” In: *Workshop on Privacy in the Electronic Society* (WPES). Ed. by Yan Chen and Jaideep Vaidya. ACM, 2011, pp. 81–92. DOI: `10.1145/2046556.2046567`.

[JL10] Stanisław Jarecki and Xiaomin Liu. “Fast Secure Computation of Set Intersection.” In: *Security and Cryptography for Networks* (SCN). Ed. by Juan A. Garay and Roberto De Prisco. Vol. 6280. LNCS. Springer, 2010, pp. 418–435. DOI: `10.1007/978-3-642-15317-4_26`.

[Ker12a] Florian Kerschbaum. “Collusion-resistant Outsourcing of Private Set Intersection.” In: *Symposium on Applied Computing* (SAC). Ed. by Sascha Ossowski and Paola Lecca. ACM, 2012, pp. 1451–1456. DOI: `10.1145/2245276.2232008`.

[Ker12b] Florian Kerschbaum. “Outsourced Private Set Intersection Using Homomorphic Encryption.” In: *Information, Computer and Communications Security* (ASIACCS). Ed. by Heung Youl Youm and Yoojae Won. ACM, 2012. DOI: `10.1145/2414456.2414506`.

[KLM+18] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J. Wu. “Function-Hiding Inner Product Encryption Is Practical.” In: *Security and Cryptography for Networks* (SCN). Ed. by Dario Catalano and Roberto De Prisco. Vol. 11035. LNCS. Springer International Publishing, 2018, pp. 544–562. DOI: `10.1007/978-3-319-98113-0_29`.

[KMR+14] Seny Kamara, Payman Mohassel, Mariana Raykova, and Saeed Sadeghian. “Scaling Private Set Intersection to Billion-Element Sets.” In: *Financial Cryptography and Data Security* (FC). Ed. by Nicolas Christin and Reihaneh Safavi-Naini. Vol. 8437. LNCS. Springer, 2014, pp. 195–215. DOI: `10.1007/978-3-662-45472-5_13`.

[KPE+17] Tim R. van de Kamp, Andreas Peter, Maarten H. Everts, and Willem Jonker. “Multi-client Predicate-Only Encryption for Conjunctive Equality Tests.” In: *Cryptology And Network Security* (CANS). Ed. by S. Capkun and Sherman S. M. Chow. Vol. 11261. LNCS. Springer International Publishing, 2017, pp. 135–157. DOI: `10.1007/978-3-030-02641-7_7`.

[KS05] Lea Kissner and Dawn Song. “Privacy-Preserving Set Operations.” In: CRYPTO. Ed. by Victor Shoup. Vol. 3621. LNCS. Springer, 2005, pp. 241–257. DOI: `10.1007/11535218_15`.

[LHK14] Wouter Lueks, Jaap-Henk Hoepman, and Klaus Kursawe. “Forward-Secure Distributed Encryption.” In: *Privacy Enhancing Technologies Symposium* (PETS). Ed. by Emiliano De Cristofaro and Steven J. Murdoch. Vol. 8555. LNCS. Springer International Publishing, 2014, pp. 123–142. DOI: `10.1007/978-3-319-08506-7_7`.

[LNZ$^+$14] Fang Liu, Wee Keong Ng, Wei Zhang, Do Hoang Giang, and Shuguo Han. “Encrypted Set Intersection Protocol for Outsourced Datasets.” In: *International Conference on Cloud Engineering* (IC2E). Ed. by Azer Bestavros *et al.* IEEE, 2014, pp. 135–140. DOI: `10.1109/IC2E.2014.18`.

[LW11] Allison Lewko and Brent Waters. “Decentralizing Attribute-Based Encryption.” In: EUROCRYPT. Ed. by Kenneth G. Paterson. Vol. 6632. LNCS. Springer, 2011, pp. 568–588. DOI: `10.1007/978-3-642-20465-4_31`.

[Mea86] Catherine Meadows. “A More Efficient Cryptographic Matchmaking Protocol for Use in the Absence of a Continuously Available Third Party.” In: *Security and Privacy* (S&P). Ed. by Clark Weissman *et al.* IEEE, 1986, pp. 134–134. DOI: `10.1109/SP.1986.10022`.

[PR12] Omkant Pandey and Yannis Rouselakis. “Property Preserving Symmetric Encryption.” In: EUROCRYPT. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. LNCS. Springer, 2012, pp. 375–391. DOI: `10.1007/978-3-642-29011-4_23`.

[PSZ14] Benny Pinkas, Thomas Schneider, and Michael Zohner. “Faster Private Set Intersection Based on OT Extension.” In: USENIX SECURITY. Ed. by Kevin Fu and Jaeyeon Jung. USENIX Association, 2014, pp. 797–812. URL: `https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/pinkas` (visited on 2019-01-30).

[SCR$^+$11] Elaine Shi, T.-H. Hubert Chan, Eleanor G. Rieffel, Richard Chow, and Dawn Song. “Privacy-Preserving Aggregation of Time-Series Data.” In: *Network and Distributed System Security Symposium* (NDSS). The Internet Society, 2011. URL: `https://www.ndss-symposium.org/ndss2011/privacy-preserving-aggregation-of-time-series-data/` (visited on 2019-01-30).

[Sma16] Nigel P. Smart. *Cryptography Made Simple*. Ed. by David Basin and Kenny Paterson. Information Security and Cryptography. Cham: Springer International Publishing, 2016. DOI: `10.1007/978-3-319-21936-3`.

[ZX15] Qingji Zheng and Shouhuai Xu. “Verifiable Delegated Set Intersection Operations on Outsourced Encrypted Data.” In: *International Conference on Cloud Engineering* (IC2E). Ed. by K. Selçuk Candan and Kyung Dong Ryu. IEEE, 2015, pp. 175–184. DOI: `10.1109/IC2E.2015.38`.