Multi-clients Verifiable Computation via Conditional Disclosure of Secrets

Rishabh Bhadauria(✉) and Carmit Hazay(✉)

Bar-Ilan University, Ramat-Gan, Israel
{rishabh.bhadauria,carmit.hazay}@biu.ac.il

Abstract. In this paper, we explore the connection between two-party conditional disclosure of secrets (CDS) and verifiable computation. Here, the integrity mechanism underlying CDS is leveraged to ensure two-clients verifiable computation, where the computation is outsourced to an external server by two clients that share the input to the function. Basing integrity on CDS enjoys several significant advantages such as non-interactivity, constant rate communication complexity, a simple verification procedure, easily batched, and more.

In this work, we extend the definition of plain CDS, considering two additional security properties of privacy and obliviousness that respectively capture input and output privacy. We then show that these extended notions of CDS are useful for designing secure two-party protocols in the presence of an untrusted third party.

We complement the above with a sequence of new CDS constructions for a class of predicates of interest, including private set-intersection (PSI) and set-union cardinality, comparison, range predicate, and more. Based on these constructions we design new non-interactive constant-rate protocols for comparing two strings based on symmetric-key cryptography, and without requiring bit-decomposition. We additionally design new protocols for PSI cardinality and PSI based on recent work by Le, Ranellucci, and Gordon (CCS 2019) with similar advantages.

1 Introduction

In this paper, we explore the connection between two-party conditional disclosure of secrets (CDS) [11] and verifiable computation. CDS is a generalization of secret-sharing, where two parties (denoted by Alice and Bob), that share a uniform string r, wish to disclose a secret s to a third party (denoted by Claire) if and only if their respective inputs x_1 and x_2 satisfy some predicate $f : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ (we denote such inputs as 1-inputs). CDS is defined by two encoding algorithms $(\mathsf{Enc}_1, \mathsf{Enc}_2)$ for Alice and Bob to respectively compute their messages to Claire. Based on these encodings, and the knowledge of x_1 and x_2, Claire runs a decoding algorithm Dec to extract s. Note that CDS does

C. Hazay—This work is supported by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office, and by ISF grant No. 1316/18.

C. Galdi and V. Kolesnikov (Eds.): SCN 2020, LNCS 12238, pp. 150–171, 2020.
https://doi.org/10.1007/978-3-030-57990-6_8

not maintain the privacy of Alice and Bob's inputs. In the past two decades, different aspects of CDS have been studied extensively exploring its communication complexity, the complexity of the decoder, and its expressibility; see some recent examples [1–3,17]. Concerning the latter aspect, note that garbled circuits [22] imply CDS for any polynomial function assuming only one-way functions and requiring communication complexity $O(\kappa \cdot |\mathrm{C}|)$ where C is the computed circuit and κ is the security parameter.

Verifiable Computation from CDS. In this work, we explore the observation that for CDS with sufficiently long secrets, the secret can serve as proof for the fact that the output of f equals 1. Namely, at the heart of every CDS construction lies an integrity mechanism that prevents Claire from learning s for 0-inputs. This observation is not new and previously made in the context of attribute-based encryption [20], a public key object analogue to CDS. We exploit this mechanism to demonstrate the usefulness of CDS for designing non-interactive two-clients verifiable protocols in the presence of untrusted server, for a class of predicates that admit CDS. Informally, given two CDS schemes for f and $\bar{f}$, the clients forward the server the encoding of their inputs within these CDS. The server then replies with the decoded messages. From the secrecy property of the underlying CDS, the server should not extract both secrets.

In more detail, our basic two-client model includes three parties; two clients C_0 and C_1 that have an input and access to shared randomness, and a third untrusted server S. The clients wish to delegate the computation of some predicate f to an external unreliable server while performing less work than mutually evaluating the function. Note that this modeling differs from classic verifiable computation [9] by distributing the input between two clients. In this work, we prove that CDS gives rise to verifiable computation on distributed data. The security of our constructions holds in the presence of a malicious server (that follows an arbitrary attack strategy) and the semi-honest corruption of a single client (or any proper subset of the clients in the general setting). Some of our constructions also tolerate the malicious behaviour of the clients.

Prior work in the multi-client setting [6,13] showed generic solutions with a preprocessing phase whose running time depends on the complexity of f, which therefore must be amortized away. On the other hand, we only focus on a concrete set of predicates of interest where our solutions are not involved with a preprocessing phase. Moreover, the complexity of the input encoding algorithms of our underlying CDS schemes is strictly smaller than applying a multi-party protocol between the clients; we elaborate on this point more concretely below. We consider two flavours of privacy: input privacy where the server may only conclude the outcome of the predicate, and full privacy in the spirit of the simulation-based definition of secure computation, where only the clients learn the outcome of the predicate.

Applying the CDS abstraction for verifiable computation enjoys some qualitative advantages:

1. Round complexity. First, CDS based constructions are non-interactive and require only a single message in each direction. Reducing the round

complexity of secure protocols is an important goal, where non-interactive protocols are particularly attractive. Here the clients can post the "encoding" of their inputs and go offline, allowing the server to evaluate the function on the inputs and sending a single message to the client.
2. Hardness assumptions. CDS is an information-theoretic object that can potentially be realized only based on one-way functions or even without any assumption. In this work, we additionally require using Σ-protocols.
3. Constant rate communication complexity. The encoding messages sent from the clients imply constant upload rate communication complexity.[1] As demonstrated below, this complexity is much smaller than the complexity achieved in prior work for our particular class of predicates.
4. Simplicity. The verification procedure of our schemes is as simple as checking equality between the untrusted party's response and the secret s.
5. Batching. The verification algorithm can be easily batched, taking the linear combination of many secrets and amortizing away the communication complexity of the server. This batching approach implies (amortized) download rate-1 which is much smaller than the solutions from [6,13] that incur download rate of κ.
6. Point-to-point channels. Our protocols do not employ any broadcast channels and only rely on point-to-point channels. This also means that Claire can launch a selective abort attack.
7. Transparent setup and no preprocessing. Finally, recall that CDS schemes require common randomness between Alice and Bob. This can be viewed as mutual access to a common random string which is simpler to realize than employing a preprocessing phase that requires communication and has a secret trapdoor.

2PC with an Untrusted Helper from CDS. The second model where CDS is useful for is a two-party computation with an untrusted "helper". Namely, the classic notion of two-party computation is extended to the three-party setting, where the third party S performs the computation on the inputs of the other two parties. Security in this model holds as long as at most one of the parties is corrupted. This model has been recently considered in [16] with the aim of designing more practical set operations protocols. In order to use a simulation based definition for our protocols, we extend the security definition of plain CDS to support two additional features: privacy and obliviousness. Loosely speaking, privacy implies that only the output of the predicate may be leaked to Claire whereas obliviousness implies that nothing is leaked. This forms a hierarchy of definitions and captures additional scenarios that require both privacy and integrity.

New CDS Constructions. We design new (private/oblivious) CDS constructions for several important predicates. We then establish our verifiable schemes based on these CDS constructions.

[1] We measure the upload rate as the ratio between the size of the encoded messages and the inputs. We further define the download rate by the ratio between the size of $f(x_1, x_2)$ and s.

Equality/Inequality. We begin with CDS schemes for verifying equality and inequality of sufficiently long strings without requiring bit-decomposition. Protocols for securely comparing private values are fundamental building blocks of secure computation with numerous applications, initiated by the famous millionaires problem by Yao [22]. Nevertheless, designing concretely efficient protocols has remained a challenge. The state of the art concrete (amortized) analysis for semi-honest secure comparison [7] requires $O(\kappa\ell/\log\kappa)$ bits in the setup phase and $O(\ell)$ bits in the online phase, based on oblivious transfer. Another recent work [4] for equality based on function secret sharing requires $\lambda\ell$ bits in the setup phase and ℓ bits in the online phase. Currently, this work implies the best online communication in the semi-honest setting. Using somewhat homomorphic encryption scheme, the (amortized) bit complexity of [10] is $\tilde{O}(\ell+\kappa)$. Both [10] and [7] require non-constant round complexity.

In contrast, our non-interactive protocols achieve small upload rate (e.g., 10) for moderately long strings (e.g., polylogarithmic is κ), and rely on one-way functions and Σ-protocols.[2] Compared with [4], our scheme induces a higher upload rate in the online phase. Nevertheless, we require a simpler and more efficient setup, as our setup only requires a uniform random string while [4] requires correlated randomness (in the form of function secret sharing keys for computing a distributed point function), as well as more bits in the setup phase.

We introduce two CDS schemes for equality and inequality. One first scheme uses one-way functions (or pseudorandom functions (PRFs)) while the other scheme uses Σ-protocols as well. Specifically, we leverage the special soundness property of the Σ-protocol, which implies an algorithm that extracts the witness given two transcripts with the same first message and distinct challenges. In the semi-honest setting (where the clients are semi-honest and Claire may be malicious), we require one way functions and Σ-protocols. Specifically, we use one way functions to encode the input, and a Σ-protocol which is easily sampleable with respect to witness. This means that given a witness ω, we can generate a statement x. Moving to the malicious setting requires to replace the PRFs with PRPs as well as to utilize Σ-protocols that have an extractability algorithm for extracting the verifier's randomness given the transcript, witness, statement and prover's randomness. For instance, Schnorr's protocol [21] satisfies both of these properties. The verification algorithm provided by the Σ-protocol is also essential in allowing it to be maliciously secure. We note that this mechanism can also be extended to the multi-party case where we would use Σ-protocol with an extended special soundness property. We compare our construction to prior work in Table 1. Finally, we note that our protocols can be extended to a zero test with some tweaks.

[2] Loosely speaking, a Σ-protocol is a 3-round public-coin interactive proof for an NP relation, for which there exists an extractor that extracts the witness upon rewinding the prover. We require an additional transcript verifiability property that is leveraged for achieving correctness against malicious input encoding of Alice and Bob, going beyond semi-honest security.

PSI CARDINALITY. Private set-intersection (PSI) is an important functionality that gained much attention from the cryptographic community due to its broad applicability. It is defined by computing the intersection of two (or more) sets X_1 and X_2. In this work, we show that a non-interactive variant of a recent set-intersection protocol by Le et al. [16] implies CDS for the cardinality of set-intersection based on one-way functions. More concretely, we show that (a variant of one of) their protocols induces two CDS constructions for verifying upper and lower bounds on the intersection size, yielding the exact size of the set-intersection with rate 3 communication complexity. We note that these CDS techniques can be easily extended to address set-union cardinality, set-membership and small domain range predicates. Using the CDS for verifying upper bounds of PSI intersection as well as using our CDS for equality, we can construct a two-party PSI with untrusted server and constant rate (4 in the passive setting and 8 in the active setting). As above, we can extend the security of our schemes to the malicious setting. Finally, we note that in order to use our CDS schemes for verifiable computation, the soundness error probability $1/|\mathbb{F}|$ must be negligible. Therefore, $|\mathbb{F}|$ must be at least of size polylogarithmic in κ.

Table 1. A comparison of our equality protocol with prior work where λ is the security parameter, the inputs are of size ℓ bits and OT is oblivious transfer.

Construction	Setup	Round complexity	Online comm.	Offline comm.	Hardness assump.	Security
[7]	Correlated[a]	≥ 3	$\mathcal{O}(\lambda\ell)$	$3\ell + o(\ell)$	OWF +OT	Passive
[18]	Uniform	3	$\mathcal{O}(\lambda\ell)$	$\mathcal{O}(\lambda\ell)$	OWF+OT	Active
[4]	Correlated[b]	2	ℓ	$\lambda\ell$	OWF	Passive
Fig. 7	Uniform	2	3ℓ	6ℓ	OWF	Passive
Fig. 7	Uniform	2	10ℓ	7ℓ	OWF+Σ-protocol[c]	Active

[a] This work uses two types of correlated randomness that are generated using OT for XOR and AND shares.
[b] This correlation requires keys for computing distributed point functions.
[c] We concretely rely here on the hardness of discrete logarithm in groups.

Prior work on verifiable set operations for the single client setting [5,19] relied on stronger hardness assumptions such as q-string Diffie-Hellman and extractable collision-resistant hash functions, but also achieved stronger properties such as public verifiability and dynamically changing sets. [8] was the first work which introduced the concept of server-aided two-party PSI where the server is used as a helper party to resolve a dispute under stringer assumptions. While the communication complexity is linear and the number of rounds is constant, the rate is higher (at least 16) and the number of rounds is at least 7 (excluding the extra rounds occurred due to zero-knowledge proofs). In a followup work [15], Kamara et al. constructed a 3-round server-aided two-party PSI with a malicious server where the communication complexity is inflated by some statistical parameter γ.

A recent work on threshold PSI [12] shows semi-honest protocols for comparing the union size of two sets when excluded with the intersection against some threshold parameter t. These protocols introduce communication complexity $\tilde{O}(t)$ (resp. $O(t^2)$ and $O(t^3)$) assuming fully homomorphic encryption (resp. additively homomorphic encryption and oblivious transfer). It is an interesting problem to extend these techniques to the non-interactive setting (where the second solution is not constant round).

2 Preliminaries

2.1 Σ-Protocols

A Σ-protocol is a 3-round primitive that implies zero-knowledge for honest verifiers with special soundness. In this work, we require the following additional properties:

- **Efficiently sampleable.** We require from the underlying Σ-protocol to be efficiently sampleable with respect to the witness. Formally, there exists an efficient algorithm $x \leftarrow \text{Gen}(\omega, 1^\kappa)$ such that $(x, \omega) \in \mathcal{R}$.
- **Randomness extraction.** We require from the underlying Σ-protocol to have an extractability property where the extractor extracts the prover's randomness given the protocol's transcript, witness and statement of the Σ-protocol. More formally, there exist a PPT algorithm $\mathsf{RandExt}$ such that $p_r = \mathsf{RandExt}(x, \omega, T)$ which can extract the randomness r_p associated with the prover's algorithm in Σ-protocol, $(x, \omega) \in \mathcal{R}$ and T correspond to the transcript of Σ-protocol.

In this work, we employ Schnorr's Protocol [21] that satisfies these two properties. We also use the verification algorithm of the verifier to determine whether a transcript is accepting (or valid).

2.2 Conditional Disclosure of Secrets

We begin with the basic definition of CDS as given in [11]. We require computational privacy and thus implicitly assume that all our algorithms receive the security parameter κ as part of their input. Furthermore, some of our constructions will evaluate a predicate over a field $\mathbb{F}$ rather than on bit strings. We note that the input sizes will always be polynomially related to κ.

Definition 1 (CDS). *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a predicate. Let Enc_1 and Enc_2 be two* PPT *encoding algorithms and Dec be deterministic decoding algorithm. Let $s \in \{0,1\}^\kappa$ be a secret and r joint randomness drawn from the uniform distribution. Then the following conditions must hold:*

– **Correctness:** *For every input* (x_1, x_2) *which satisfies the condition* $f(x_1, x_2) = 1$ *and a secret* s, $\Pr[\mathsf{Dec}(x_1, x_2, \mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)) \neq s] \leq \mathsf{negl}(\kappa)$
– **Secrecy:** *There exists a* PPT *simulator* Sim *such that for every input* (x_1, x_2) *which satisfies the condition* $f(x_1, x_2) = 0$ *and a secret* $s \in \{0,1\}^\kappa$,

$$\left\{\mathsf{Sim}(x_1, x_2)\right\}_{x_1,x_2 \in \{0,1\}^n} \overset{c}{\approx} \left\{\mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)\right\}_{x_1,x_2 \in \{0,1\}^n}$$

2.3 Multi-clients Verifiable Computation

In this model, a set of clients outsource the computation of a function f over their distributed inputs to an untrusted server. We are interested in a non-interactive multi-client verifiable computation where the clients do not interact with each other after the setup phase (which is used to generate common randomness r and is independent of the function and the client's inputs). Note that this phase can be realized either via a one-time coin-tossing protocol or by accessing a public source of randomness, such as a random oracle applied on a fixed value. As in [13], we consider two security flavours for this setting, where the clients are either semi-honest or malicious whereas the servers are always malicious. We will continue with the syntax.

Syntax. An t-party multi-clients verifiable computation (MVC) scheme with semi-honest clients consists of the following algorithms:

– $r \leftarrow \mathsf{Setup}$: All clients receive a uniform string r.
– $(\tilde{x}_j, \tau_j) \leftarrow \mathsf{Input}(x_j, r, 1^\kappa)$. Each client C_j will run this *input encoding algorithm* on its input x_j and randomness r. The output of this algorithm is an encoded input $\tilde{x}_j$, which will be sent to the server, and the input decoding secret τ_j which will be kept private by the client.
– $(\alpha_1, \ldots, \alpha_t) \leftarrow \mathsf{Compute}(\tilde{x}_1, \ldots \tilde{x}_t, f)$. Given the encoded inputs $\{\tilde{x}_j\}_j$ and the function description, this *computation algorithm* computes an encoded output α_j.
– $y \cup \{\perp\} \leftarrow \mathsf{Verify}(\tau_j, \alpha_j)$. Each client C_j runs this *verification algorithm* with the decoding secret τ_j, and the encoded output α_j. The algorithm outputs either a value y (that is supposed to be $f(x_1, \ldots, x_t)$), or $\perp$ indicating that the server attempted to cheat.

Note that the setup can also be made reusable using a pseudorandom function. In contrast, prior work requires a more complicated setup phase. Efficiency wise, we would like the time it takes for encoding the input and for verifying the output of the server to be less than computing f. Moreover, correctness can be defined naturally by requiring that the outcome of the computation algorithm passes the verification algorithm with overwhelming probability when correctly generated. We continue with a soundness definition.

Definition 2 (Soundness of MVC). *For a multi-client verifiable computation scheme MVC, consider the following experiment with respect to an adversarial server $\mathcal{A}$:*

$$\Pr\left[y \neq f(x_1,\ldots,x_t) \;\middle|\; \begin{array}{c} \text{for all } j \in [t],\ r \leftarrow \mathsf{Setup}, (\tilde{x}_j, \tau_j) \leftarrow \mathsf{Input}(x_j, r, 1^\kappa), \\ (\alpha_1,\ldots,\alpha_t) \leftarrow \mathcal{A}(\tilde{x}_1,\ldots\tilde{x}_t, f)\ ,\ y \leftarrow \mathsf{Verify}(\tau_j, \alpha_j) \end{array} \right] \leq \mathsf{negl}(\kappa)$$

Security Against Malicious Clients. The above definition holds for semi-honest clients (that follow the protocol faithfully) as long as they do not collude with the server. A stronger notion considers security in the presence of malicious clients (that follow an arbitrary attack strategy). In the two-client setting, our constructions are secure in the presence of a single malicious corruption of one of the clients.

Achieving Privacy. Our definition does not guarantee input or output privacy. These properties will be derived directly from the underlying CDS construction. Namely, if the CDS will be private or oblivious then the MVC will respectively maintain input or output privacy. For the constructions that achieve both privacy and correctness, we will use a simulation-based definition to prove security.

3 New Variants of CDS

In this section, we define two new variants of CDS: - private CDS and oblivious CDS. While private CDS hides the input of the clients achieving input privacy, oblivious CDS achieves input privacy as well as output privacy.

We extend Definition 1 by not giving the inputs x_1 and x_2 to both the decoder and the simulator Sim. This implies that Claire does not gain any information about x_1 and x_2 from the encoded messages of Alice and Bob, but may still conclude the outcome of the predicate. This definition will be useful for achieving input privacy in our multi-client verifiable computation constructions.

Definition 3 (Private CDS). *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a predicate. Let Enc_1 and Enc_2 be* PPT *encoding algorithms and Dec be deterministic decoding algorithm. Let $s \in \{0,1\}^\kappa$ be a secret and r joint randomness drawn from the uniform distribution. Then the following conditions must hold:*

- **Correctness:** *For every input (x_1, x_2) which satisfies the condition $f(x_1, x_2) = 1$ and a secret s,* $\Pr[\mathsf{Dec}(\mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)) \neq s] \leq \mathsf{negl}(\kappa)$.
- **Privacy:** *There exists a* PPT *simulator* Sim *such that for every input (x_1, x_2) and a secret s,*

$$\left\{\mathsf{Sim}(1^{|x_1|}, 1^{|x_2|}, y)\right\}_{x_1,x_2\in\{0,1\}^n} \overset{c}{\approx} \left\{\mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)\right\}_{x_1,x_2\in\{0,1\}^n}$$

$$\left\{\mathsf{Sim}(1^{|x_1|}, 1^{|x_2|}, y, s)\right\}_{x_1,x_2\in\{0,1\}^n} \overset{c}{\approx} \left\{\mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)\right\}_{x_1,x_2\in\{0,1\}^n}$$

where the first equation holds for $y = 0$ and the second equation holds for $y = 1$, and $y = f(x_1, x_2)$.

Finally, we consider a simulation-based definition, where we require that Claire cannot conclude any information about the secret. This definition is formalized by requiring that the encoded messages can be simulated without the knowledge of both the inputs nor the output (namely, the secret).

Definition 4 (Oblivious CDS). *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a predicate. Let Enc_1 and Enc_2 be two* PPT *encoding algorithms and* Dec *be deterministic decoding algorithm. Let $s \in \{0,1\}^\kappa$ be a secret and r a joint randomness drawn from the uniform distribution. Then the following conditions must hold:*

- **Correctness:** *For every input (x_1, x_2) which satisfies the condition $f(x_1, x_2) = 1$ and a secret s,* $\Pr[\mathsf{Dec}(\mathsf{Enc}_1(x_1, s, r), \mathsf{Enc}_2(x_2, s, r)) \neq s] \leq \mathsf{negl}(\kappa)$.
- **Indistinguishability:** *For every* PPT *active adversary $\mathcal{A}$ in real model corrupting Claire, there exists a* PPT *algorithm* Sim *in ideal world such that:*

$$\{\mathrm{REAL}_{f,A(z)}(x_1, x_2, s, n)\}_{x_1,x_2,s,n} \overset{c}{\approx} \{\mathrm{IDEAL}_{f,\mathsf{Sim}(z)}(x_1, x_2, s, n)\}_{x_1,x_2,s,n}$$

where f is the computed predicate and x_1 and x_2 are the inputs of Alice and Bob, respectively.

4 New CDS Constructions

In what follows, we discuss our CDS constructions for a class of predicates. In Sect. 4.1 we present private and oblivious CDS schemes for the equality predicate. In Sect. 4.2 we present two private CDS schemes for the inequality predicate. In Sect. 4.3 we present a CDS scheme for verifying lower and upper bounds of PSI cardinality. As a general note, we remark that our analysis relies on the fact that the secret is unpredictable (or uniformly random). This is sufficient for our applications, as the parties choose the secret by themselves. For applications where the parties have no control in choosing the secret, they can run our protocols with $F_k(s)$, where F is a PRF.

4.1 CDS for Equality

In this section, we present two CDS constructions for verifying the equality of two strings. Our first construction, presented in Fig. 1, shows a simple oblivious CDS scheme with information-theoretic security. Whereas our second CDS scheme, presented in Fig. 2, shows a private CDS scheme assuming one-way functions. The latter construction uses Σ-protocols as an underlying building block and can be extended to ensure the correctness of the encoding computations of Alice and Bob by relying on the verifiability property of the Σ-protocol, thus enhancing the security of the scheme.

The high level idea of our first construction is by generating two linearly independent equations such that extracting the secret is possible only if the inputs are equal. Due to the perfect secrecy nature of the two equations, the scheme preserves obliviousness, as Claire cannot detect whether she learned the correct secret or not. Specifically, our CDS achieves information-theoretic security and is computationally lightweight. This idea is similar to the multi-party CDS construction from [14].

$\mathsf{CDS}_{\mathrm{EQ}_1}$

- **Inputs.** The CDS protocol for equality is invoked by the interface $\mathsf{CDS}_{\mathrm{EQ}_1}(x_1, x_2, s, (r_1, r_2))$ where
  - x_1 and x_2 is input of Alice and Bob respectively over a field $\mathbb{F}$.
  - r_1 and r_2 are the shared randomness.
  - $s \in \mathbb{F}$ is a secret value which is shared between Alice and Bob.

  The computed function is $f(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2 \\ 0 & \text{otherwise} \end{cases}$
- **Output.** Claire outputs s if $f(x_1, x_2) = 1$ and an independent string $s' \neq s$ otherwise.
- **Algorithms.**
  - $\mathsf{Enc}_1(x_1, s, (r_1, r_2)) = r_1 \cdot x_1 + r_2 + s$.
  - $\mathsf{Enc}_2(x_2, s, (r_1, r_2)) = r_1 \cdot x_2 + r_2$.
  - $\mathsf{Dec}(\tilde{x}_1, \tilde{x}_2) = \tilde{x}_1 - \tilde{x}_2$.

Fig. 1. Oblivious CDS for equality.

Theorem 4.1. *Protocol $\mathsf{CDS}_{\mathrm{EQ}_1}$ from Fig. 1 is an oblivious CDS (cf. Definition 4) for the equality predicate.*

Our second CDS construction for the equality predicate uses Σ-protocols as a platform to compare between two strings in a private manner. Namely, we leverage the (PRF evaluations of the) parties' inputs as the randomness sources to compute the prover's first message of the Σ-protocol and fix the secret as the witness for the corresponding relation. Then, only if equality holds we ensure that Claire can extract the secret due to the special soundness property of the protocol. Our detailed construction is given in Fig. 2. Note that this scheme achieves private CDS as Claire cannot conclude any information about the parties' inputs due to the privacy of the PRF. Nevertheless, it can conclude the outcome of the predicate.

Theorem 4.2. *Assume the existence of pseudorandom functions and a Σ-protocol for some predefined NP relation $\mathcal{R}$, then protocol $\mathsf{CDS}_{\mathrm{EQ}_2}$ from Fig. 2 is a private CDS (cf. Definition 3) for the equality predicate.*

$\mathsf{CDS}_{\mathrm{EQ}_2}$

- **Inputs.** The CDS protocol for equality is invoked by the interface $\mathsf{CDS}_{\mathrm{EQ}_2}(x_1, x_2, s, (r_1, r_2, r_{\mathrm{PRF}}))$ where
  - x_1 and x_2 are the respective inputs of Alice and Bob from $\{0,1\}^\kappa$.
  - r_1, r_2 and r_{PRF} are the shared randomness.
  - $s \in \{0,1\}^\kappa$ is a secret value which is shared between Alice and Bob.

  The computed function is $f(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2 \\ 0 & \text{otherwise} \end{cases}$
- **Notations.** We require that the fractions of r_1 and r_2 from the joint randomness should not be equal (namely, $r_1 \neq r_2$). This is necessary for the extractability of secret s. The protocol is parameterized by an NP relation $(x, \omega) \in \mathcal{R}$ and a Σ-protocol with an extractor $\mathcal{E}$, such that s serves as the witness and x is the corresponding public statement (we assume one can compute x from ω). The Σ-protocol transcript is generated by emulating $\mathcal{P}$ and $\mathcal{V}$ yielding $(a, e, z) = \langle \mathcal{P}(x, \omega; r), \mathcal{V}(x) \rangle$ where r is the prover's randomness.
- **Output.** Claire outputs s if $f(x_1, x_2) = 1$ and $\perp$ otherwise.
- **Algorithms.**
  - $\mathsf{Enc}_1(x_1, s, (r_1, r_2, r_{\mathrm{PRF}}))$:
    - A public statement x is generated as explained above using $\omega = s$ for the Σ-protocol.
    - A PRF key k is generated using r_{PRF}.
    - The input x_1 is encoded as $\mathbf{x}_1 = F_k(x_1)$.
    - The output is generated as

$$\tilde{x}_1 = (x, (a_1, e_1, z_1)) = (x, \langle \mathcal{P}(x, \omega; \mathbf{x}_1), \mathcal{V}(x, r_1) \rangle).$$

  - $\mathsf{Enc}_2(x_2, s, (r_1, r_2, r_{\mathrm{PRF}}))$:
    - A public statement x is generated as explained above using $\omega = s$ for the Σ-protocol.
    - A PRF key k is generated using r_{PRF}.
    - The input x_2 is encoded as $\mathbf{x}_2 = F_k(x_2)$.
    - The output is generated as

$$\tilde{x}_2 = (x, (a_2, e_2, z_2)) = (x, \langle \mathcal{P}(x, \omega; \mathbf{x}_2), \mathcal{V}(x, r_2) \rangle).$$

  - $\mathsf{Dec}(\tilde{x}_1, \tilde{x}_2)$:
    - The message $\tilde{x}_1$ is broken down into $(x, (a_1, e_1, z_1))$ and the message $\tilde{x}_2$ is broken down into $(x, (a_2, e_2, z_2))$.
    - $s' = \begin{cases} \mathcal{E}(\tilde{x}_1, \tilde{x}_2) & \text{if } a_1 = a_2 \\ \perp & \text{otherwise} \end{cases}$

Fig. 2. Private CDS for equality.

The correctness property of Protocol $\mathsf{CDS}_{\mathrm{EQ}_2}$ relies on the special soundness property. The messages sent by Alice and Bob to Claire consist of a statement of Σ-protocol along with the transcripts of Σ-protocol. In the case of the two input values of Alice and Bob are equal, this will result in the first message of the transcript being same. As we have two different transcripts with the same first message, the special soundness property of Σ-protocol always allows us to extract the secret s. The privacy property of the inputs in case of 0-output relies on the special honest verifier zero-knowledge which allows the Sim to generate valid transcripts of a given statement x based on inputting x and the random value e which acts as the second message in Σ protocol. We show that two transcripts generated through the above algorithm is indistinguishable to the message sent in the real protocol. The privacy property of the inputs in the case of 1-output is based on indistinguishability of pseudorandom functions from truly random functions.

Communication Complexity. Note that both our protocols achieve a constant upload rate. Specifically, the rate of our oblivious CDS (Fig. 1) is 1 whereas the rate of our private CDS (Fig. 2) is 5. These are the first protocols to achieve this rate for comparison based on symmetric-key assumptions and Σ-protocols.

Achieving Malicious Security. We further note that our private protocol can achieve stronger security for the clients by relying on the verifiability property of the Σ-protocol for NP statements with a single witness. In particular, before extracting the secret, Claire can check whether the transcripts are generated correctly and that $e_1 \neq e_2$. That would imply that the prover knows *some* secret, but not necessarily that it has used the correct secret. However, from the fact that the parties also send the statements as part of their messages, Claire can easily test whether the same secret was used by both parties and abort otherwise. This implies correctness with respect to the encoding algorithms. To prove malicious security, we would also need to extract the parties' inputs using the randomness extraction property specified in Definition 2.1.

4.2 CDS for Inequality

In this section, we present two CDS constructions for inequality for checking whether two strings are identical. Both constructions, presented in Figs. 3 and 4, achieve private CDS assuming one-way functions. The latter construction uses Σ-protocols as an underlying building block and can be extended to ensure the correctness of the encoding computations of the parties by relying on the verifiability property of the Σ-protocol, similar to the argument for the equality CDS scheme from the prior section.

Our first construction for the inequality predicate uses a 1-degree polynomial as the basis of the construction. The parties first apply a PRF on their inputs, and then utilize it to generate an evaluation of a polynomial that is embedded with the secret as the constant coefficient. Then if the inequality condition holds, algorithm Dec will be given two different evaluations of a 1-degree polynomial and can interpolate the polynomial, learning the secret. If inequality does not

$\mathsf{CDS}_{\text{INEQ}_1}$

- **Inputs.** The CDS protocol for equality is invoked by the interface $\mathsf{CDS}_{\text{INEQ}_1}(x_1, x_2, s, (r, r_{\text{PRF}}))$ where:
  - x_1 and x_2 are the respective inputs of Alice and Bob over a field $\mathbb{F}$.
  - r and r_{PRF} are the shared randomness.
  - $s \in \mathbb{F}$ is a secret value which is shared between Alice and Bob.

  The computed functions if $f(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 \neq x_2 \\ 0 & \text{otherwise} \end{cases}$
- **Output.** Claire outputs s if $f(x_1, x_2) = 1$ and $s' \neq s$ otherwise.
- **Algorithms.**
  - $\mathsf{Enc}_1(x_1, s, (r_1, r_2))$:
    - A PRF key k is generated using r_{PRF}.
    - The input x_1 is encoded as $\mathbf{x}_1 = F_k(x_1)$.
    - A polynomial is constructed as $p(x) = r \cdot x + s$.
    - The output is generated as $\tilde{x}_1 = (\mathbf{x}_1, p(\mathbf{x}_1))$.
  - $\mathsf{Enc}_2(x_2, s, (r_1, r_2))$:
    - A PRF key k is generated using r_{PRF}.
    - The input x_2 is encoded as $\mathbf{x}_2 = F_k(x_2)$.
    - A polynomial is constructed as $p(x) = r \cdot x + s$.
    - The output is generated as $\tilde{x}_2 = (\mathbf{x}_2, p(\mathbf{x}_2))$.
  - $\mathsf{Dec}(\tilde{x}_1, \tilde{x}_2)$:
    - The secret is generated by interpolating the points $\tilde{x}_1$ and $\tilde{x}_2$ to retrieve the polynomial $p'(\cdot)$ and output $s' = p'(0)$. If $\tilde{x}_1 = \tilde{x}_2$, then output $s' = \perp$

Fig. 3. First private CDS for inequality.

hold then there will be only one evaluation of the polynomial, resulting in Dec not being able to restore the polynomial and the secret.

Theorem 4.3. *Assume the existence of pseudorandom functions then protocol* $\mathsf{CDS}_{\text{INEQ}_1}$ *from Fig. 3 is a private CDS (cf. Definition 3) for the inequality predicate.*

Our second private CDS construction from Fig. 4 for the inequality predicate uses Σ-protocols as a platform to compare between two strings in a private manner. Namely, we leverage the (PRF evaluations of the) parties' inputs as the public randomness of the verifier's algorithm to produce different challenges, while using the same randomness for the prover's algorithm to ensure that its first message is identical for both parties. The secret is fixed as the witness as before. Then, only if inequality holds Claire can extract the secret due to the special soundness of the protocol.

As before, note that this scheme achieves private CDS due to the privacy of the PRF.

$\mathsf{CDS}_{\text{INEQ}_2}$

– **Input.** The CDS protocol for equality is invoked by the interface $\mathsf{CDS}_{\text{INEQ}_2}(x_1, x_2, s, (r, r_{\text{PRF}}))$ where:
  - x_1 and x_2 are the respective inputs of Alice and Bob from $\{0,1\}^\kappa$.
  - r and r_{PRF} are the shared randomness.
  - $s \in \{0,1\}^\kappa$ is a secret value which is shared between Alice and Bob.

  The computed function is $f(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 \neq x_2 \\ 0 & \text{otherwise} \end{cases}$

– **Notations.** We require that the fractions of r_1 and r_2 from the joint randomness should not be equal (namely, $r_1 \neq r_2$). This is necessary for the extractability of secret s. The protocol is parameterized by an NP relation $(x, \omega) \in \mathcal{R}$ and a Σ-protocol with an extractor $\mathcal{E}$, such that s serves as the witness and x is the corresponding public statement (we assume one can compute x from ω). The Σ-protocol transcript is generated by emulating $\mathcal{P}$ and $\mathcal{V}$ yielding $(a, e, z) = \langle \mathcal{P}(x, \omega; r), \mathcal{V}(x) \rangle$ where r is the prover's randomness.

– **Output.** Claire outputs s if $f(x_1, x_2) = 1$ and $\perp$ otherwise.

– **Algorithms.**
  - $\mathsf{Enc}_1(x_1, s, (r, r_{\text{PRF}}))$:
    - A public statement x is generated as explained above using $\omega = s$ for the Σ-protocol.
    - A PRF key k is generated using r_{PRF}.
    - The input x_1 is encoded as $\mathbf{x}_1 = F_k(x_1)$.
    - The output is generated as

$$\tilde{x}_1 = (x, (a_1, e_1, z_1)) = (x, \langle \mathcal{P}(x, \omega; r), \mathcal{V}(x; \mathbf{x}_1) \rangle).$$

  - $\mathsf{Enc}_2(x_2, s, (r_1, r_2, r_{\text{PRF}}))$:
    - A public statement x is generated as explained above using $\omega = s$ for the Σ-protocol.
    - A PRF key k is generated using r_{PRF}.
    - The input x_2 is encoded as $\mathbf{x}_2 = F_k(x_2)$.
    - The output is generated as

$$\tilde{x}_2 = (x, (a_2, e_2, z_2)) = (x, \langle \mathcal{P}(x, \omega; r), \mathcal{V}(x; \mathbf{x}_2) \rangle).$$

  - $\mathsf{Dec}(\tilde{x}_1, \tilde{x}_2)$:
    - The message $\tilde{x}_1$ is broken down into $(x, (a_1, e_1, z_1))$ and the message $\tilde{x}_2$ is broken down into $(x, (a_2, e_2, z_2))$.
    - $s' = \begin{cases} \mathcal{E}(\tilde{x}_1, \tilde{x}_2) & \text{if } e_1 \neq e_2 \\ \perp & \text{otherwise} \end{cases}$

Fig. 4. Second private CDS for inequality.

Theorem 4.4. *Assume the existence of pseudorandom functions and a Σ-protocol for some predefined NP relation $\mathcal{R}$, then protocol $\mathsf{CDS}_{\mathrm{INEQ}_2}$ from Fig. 4 is a private CDS (cf. Definition 3) for the inequality predicate.*

The proof idea of Protocol $\mathsf{CDS}_{\mathrm{INEQ}_2}$ is almost similar to that of Protocol $\mathsf{CDS}_{\mathrm{EQ}_2}$ with small modifications. In the equality protocol ($\mathsf{CDS}_{\mathrm{EQ}_2}$), the input is encoded and utilized as the randomness of $\mathcal{P}$ algorithm in Σ-protocol. The latter protocol ($\mathsf{CDS}_{\mathrm{INEQ}_2}$), the input is encoded and utilized as the randomness of $\mathcal{V}$ algorithm in Σ-protocol.

Note that the communication rate of our protocols is between 2 and 5 and that the security for Alice and Bob can be enhanced based on the verification procedure of the Σ-protocol (as discussed in the previous section) as well as extract the input of corrupt parties in malicious case by extracting the verifier's randomness (using Definition 2.1).

4.3 CDS for Bounds on PSI Cardinality

In this section, we present two CDS constructions for verifying the bound of PSI cardinality (namely, the intersection size). Figure 5 presents a CDS construction to verify a lower bound on the PSI cardinality while Fig. 6 presents a CDS construction to verify an upper bound on this cardinality. Our protocols leak the PSI cardinality to Claire. Technically, both constructions rely on polynomial evaluations as the base of the construction and utilize PRF for the privacy of parties' input. We remark that our protocols follow due simple modifications of one of the PSI protocols from [16]. Our main observation here shows that the techniques used in [16] induce CDS constructions on PSI (lower and upper bounds) cardinality.

Our first construction is a CDS scheme that verifies whether the PSI cardinality is lower bounded by some value t, that is hardcoded within the scheme. Namely, it utilizes the property that if t values are in common for both sets, then Claire should be able to extract the secret by recovering the polynomial $p(\cdot)$ of a degree $t-1$. In essence, the parties utilize an additive secret sharing on their polynomial evaluations to enable that. Claire can see which encoded elements are in common and retrieve their polynomial evaluations to extract the secret s. We next prove the following theorem.

Theorem 4.5. *Assume the existence of pseudorandom functions, then protocol $\mathsf{CDS}_{\geq t}$ from Fig. 5 is a private CDS (cf. Definition 3) for the greater than predicate.*

The second construction in Fig. 6 is a CDS scheme to verify if the PSI cardinality is of size at most t. It utilizes the property that if t values are in common for both sets, then the union $X_1 \cup X_2$ will include $n+m-t$ items. Therefore, Alice and Bob encode their inputs using a polynomial of degree $n+m-t-1$, embedding the secret s as the constant coefficient. As a result, given enough points, Claire should be able to extract the secret by recovering the polynomial $p(\cdot)$ of a degree $n+m-t-1$.

$\mathsf{CDS}_{\geq t}$

- **Inputs.** The CDS protocol for a lower bound of PSI cardinality is parameterized by t and invoked by the interface $\mathsf{CDS}_{\geq t}(X_1, X_2, s, (r_{\text{poly}}, r_{\text{PRF}_1}, r_{\text{PRF}_2}))$ where:
  - X_1 and X_2 are the inputs of Alice and Bob over $\mathbb{F}$ of respective sizes n and m.
  - r_{poly},r_{PRF_1} and r_{PRF_2} are the shared randomness.
  - $s \in \mathbb{F}$ is a secret value which is shared between Alice and Bob.

  The computed function is $f(X_1, X_2) = \begin{cases} 1 & \text{if } |X_1 \cap X_2| \geq t \\ 0 & \text{otherwise} \end{cases}$
- **Output.** Claire outputs s if $f(X_1, X_2) = 1$ and $\perp$ or $s' \neq s$ otherwise.
- **Algorithms.**
  - $\mathsf{Enc}_1(X_1, s, (r_{\text{poly}}, r_{\text{PRF}_1}, r_{\text{PRF}_2}))$:
    - Two PRF keys k_1 and k_2 are generated using r_{PRF_1} and r_{PRF_2}, respectively.
    - A polynomial $p(\cdot)$ of degree $t-1$ is picked at random based on randomness r_{poly} and s as the constant coefficient of $p(\cdot)$ (namely, $p(0) = s$).
    - The input X_1 is encoded as $\mathcal{X}_1 = \{F_{k_1}(x_1) \mid \forall x_1 \in X_1\}$.
    - The output is generated as $\tilde{X}_1 = \{(\mathbf{x}_1, p(\mathbf{x}_1) - F_{k_2}(\mathbf{x}_1)) \mid \mathbf{x}_1 \in \mathcal{X}_1\}$.
  - $\mathsf{Enc}_2(X_2, s, (r_{\text{poly}}, r_{\text{PRF}_1}, r_{\text{PRF}_2}))$:
    - Two PRF keys k_1 and k_2 are generated using r_{PRF_1} and r_{PRF_2}, respectively.
    - A polynomial $p(\cdot)$ of degree $t-1$ is generated with randomness r_{poly} and s as the constant coefficient of s.
    - The input X_2 is encoded as $\mathcal{X}_2 = \{F_{k_1}(x_2) \mid \forall x_2 \in X_2\}$.
    - The output is generated as $\tilde{X}_2 = \{(\mathbf{x}_2, F_{k_2}(\mathbf{x}_2)) \mid \mathbf{x}_2 \in \mathcal{X}_2\}$.
  - $\mathsf{Dec}(\tilde{X}_1, \tilde{X}_2)$:
    - A set $\tilde{X} = \{(a, b+c) \mid (a, b) \in \tilde{X}_1 \text{ AND } (a, c) \in \tilde{X}_2\}$ is defined.
    - If $\left|\tilde{X}\right| \geq t$ then polynomial $p'(\cdot)$ is generated by interpolating the points in $\tilde{X}$.
    - $s' = \begin{cases} p'(0) & \text{if } \left|\tilde{X}\right| \geq t \\ \perp & \text{otherwise} \end{cases}$

Fig. 5. Private CDS for a lower bound on PSI cardinality.

Theorem 4.6. *Assume the existence of pseudorandom functions, then protocol* $\mathsf{CDS}_{\leq t}$ *from Fig. 6 is a private CDS (cf. Definition 3) for the less than predicate.*

5 Multi-clients Verifiable Computation via CDS

We next discuss how to use our CDS constructions from the previous section in order to construct verifiable computation schemes in the presence of two clients (Alice and Bob) and a server (Claire). We construct such schemes for the equality predicate (Sect. 5.1), for PSI cardinality (Sect. 5.2) and for PSI (Sect. 5.3). Our constructions follow the paradigm where we construct verifiable computation based on CDS schemes for predicate f and $\bar{f}$. Namely, security is proven in

$\mathsf{CDS}_{\leq t}$

- **Inputs.** The CDS protocol for an upper bound of PSI cardinality is parameterized by t and invoked by the interface $\mathsf{CDS}_{\leq t}(X_1, X_2, s, (r_{\text{poly}}, r_{\text{PRF}}))$ where:
  - X_1 and X_2 are the inputs of Alice and Bob over $\mathbb{F}$ of the respective sizes n and m.
  - r_{poly} and r_{PRF} are the shared randomness.
  - $s \in \mathbb{F}$ is a secret value which is shared between Alice and Bob.

  The computed function is $f(X_1, X_2) = \begin{cases} 1 & \text{if } |X_1 \cap X_2| \leq t \\ 0 & \text{otherwise} \end{cases}$
- **Output.** Claire outputs s if $f(X_1, X_2) = 1$ and $\perp$ or $s' \neq s$ otherwise.
- **Algorithms.**
  - $\mathsf{Enc}_1(X_1, s, (r_{\text{poly}}, r_{\text{PRF}}))$:
    - A PRF key k is generated using r_{PRF}.
    - A polynomial $p(\cdot)$ of degree $n + m - t - 1$ is picked at random based on randomness r_{poly} and s as the constant coefficient of $p(\cdot)$ (namely, $p(0) = s$).
    - The input X_1 is encoded as $\mathcal{X}_1 = \{F_k(x_1) \mid \forall x_1 \in X_1\}$.
    - The output is generated as $\tilde{X}_1 = \{(\mathbf{x}_1, p(\mathbf{x}_1)) \mid \mathbf{x}_1 \in \mathcal{X}_1\}$.
  - $\mathsf{Enc}_2(X_2, s, (r_{\text{poly}}, r_{\text{PRF}}))$:
    - A PRF key k is generated using r_{PRF}.
    - A polynomial $p(\cdot)$ of degree $n + m - t - 1$ is generated with randomness r_{poly} and s as the constant coefficient of s ($p(0) = s$).
    - The input X_2 is encoded as $\mathcal{X}_2 = \{F_k(x_2) \mid \forall x_2 \in X_2\}$.
    - The output is generated as $\tilde{X}_2 = \{(\mathbf{x}_2, p(\mathbf{x}_2)) \mid \mathbf{x}_2 \in \mathcal{X}_2\}$.
  - $\mathsf{Dec}(\tilde{X}_1, \tilde{X}_2)$:
    - A set is generated $\tilde{X} = \{(a, b) \mid (a, b) \in \tilde{X}_1 \text{ OR } (a, b) \in \tilde{X}_2\}$.
    - If $\left|\tilde{X}\right| \geq n + m - t$ then $p'(\cdot)$ is generated by interpolating the points in $\tilde{X}$.
    - $s' = \begin{cases} p'(0) & \text{if } \left|\tilde{X}\right| \geq n + m - t \\ \perp & \text{otherwise} \end{cases}$

Fig. 6. Private CDS construction for an upper bound on PSI cardinality.

the presence of a malicious server, and either semi-honest or malicious servers. We assume that the clients have access to a coin-tossing functionality $\mathcal{F}_{\text{COIN}}$ that produces a sufficiently long string for the underlying CDS schemes, chosen uniformly at random.

5.1 MVC for Equality

We begin with a protocol for equality. Namely, Alice and Bob hold two strings they want to learn whether they are equal or not. Our protocol relies on the techniques used for the CDS constructions from Figs. 1 and 3, implying the following theorem.

Theorem 5.1. *Assume the existence of pseudorandom functions, then protocol* VC_{EQ} *from Fig. 7 is a two-clients verifiable computation for the equality predicate in the presence of semi-honest clients and malicious server.*

We prove that protocol VC_{EQ} is a verifiable computation in the presence of semi-honest Alice and Bob and malicious Claire. We actually prove that parties' views can be simulated. The only interesting case is when Claire is corrupted, since when either Alice or Bob are corrupted, the simulator can send then the corresponding secret (which is their only incoming message). When Claire is corrupted, the security follows from the privacy property of CDS for equality and inequality and so is straight forward.

VC_{EQ}

- **Inputs**. The VC protocol for equality is invoked by the interface $\mathsf{VC}_{\text{EQ}_1}(x_1, x_2)$ where x_1 and x_2 are respective inputs of Alice and Bob over a field $\mathbb{F}$.
- **Setup.** $(s_0, s_1, r_1, r_2, r_3, r_4, r_{\text{PRF}}) \leftarrow \text{G}$

  An ideal functionality $\mathcal{F}_{\text{COIN}}$ is invoked to generate the shared randomness $(s_0, s_1, r_1, r_2, r_3, r_4, r_{\text{PRF}})$.
- **Protocol.**
  - Alice, Bob and Claire run $\mathsf{CDS}_{\text{INEQ}_1}(x_1, x_2, s_0, (r_1, r_2, r_{\text{PRF}}))$ (cf. Figure 3). Claire sends s'_0 to Alice and Bob as the output of the CDS protocol.
  - Alice and Bob run $\mathsf{CDS}_{\text{EQ}_1}(x_1, x_2, s_1, (r_3, r_4))$ (cf. Figure 1). Claire sends s'_1 to Alice and Bob as the output of the CDS protocol. In case $s'_0 \neq \perp$, Claire sets $s'_1 = \perp$ and sends it as the output of $\mathsf{CDS}_{\text{EQ}_1}$. This step can run in parallel to the step above where Claire sends (s'_0, s'_1) together.
  - Upon receiving (s'_0, s'_1) from Claire, Alice/Bob outputs '0' if $s'_0 = s_0$, '1' if $s'_1 = s_1$ and outputs "ABORT" otherwise.

Fig. 7. 2-client VC for equality.

Recalling that we have two instantiations of both the equality and inequality CDS schemes. Then another construction can be defined based on $\mathsf{CDS}_{\text{INEQ}_2}$ and $\mathsf{CDS}_{\text{EQ}_2}$. The difference between the two protocols is that in the former both underlying CDS schemes are based on Σ-protocols that we exploit to achieve malicious security. Specifically, Claire checks whether the transcripts are generated correctly and tries to extract one of the secrets.

5.2 MVC for PSI Cardinality

We continue with our construction for PSI cardinality. As a warmup, we describe a slight variant of [16], where the parties first learn from Claire (a possibly incorrect) size t of the intersection, and then invoke our CDS schemes $\mathsf{CDS}_{\geq t}$ and

$\mathsf{PSI-CA}$

- **Inputs.** The VC for PSI cardinality protocol is invoked by the interface $\mathsf{PSI-CA}_1(X_1, X_2)$ where X_1 and X_2 are the inputs of Alice and Bob of the respective sizes n and m. Every element in the set is over a field $\mathbb{F}$.
- **Setup.** $(s_0, s_1, r_{\text{poly}_1}, r_{\text{poly}_2}, r_{\text{PRP}}, r_{\text{PRP}_2}) \leftarrow \text{G}$

  An ideal functionality $\mathcal{F}_{\text{COIN}}$ is invoked to generate the shared randomness $(s_0, s_1, r_{\text{poly}_1}, r_{\text{poly}_2}, r_{\text{PRP}}, r_{\text{PRP}_2})$.
- **Protocol.**
  - A PRP key k is generated using r_{PRP}.
  - Alice and Bob encode their inputs as $\mathcal{X}_1 = \{F_k(x_1) \mid x_1 \in X_1\}$ and $\mathcal{X}_2 = \{F_k(x_2) \mid x_2 \in X_2\}$ and send $\mathcal{X}_1$ and $\mathcal{X}_2$ to Claire respectively.
  - Claire sends $t = |\mathcal{X}_1 \cap \mathcal{X}_2|$ to Alice and Bob.
  - Alice, Bob and Claire run $\mathsf{CDS}_{\geq t}(X_1, X_2, s_0, (r_{\text{poly}_1}, r_{\text{PRP}}, r_{\text{PRP}_2}))$. Claire sends s_0' to Alice and Bob as the output of the CDS protocol.
  - Alice, Bob and Claire run $\mathsf{CDS}_{\leq t}(X_1, X_2, s_1, (r_{\text{poly}_2}, r_{\text{PRP}@Its}))$. Claire sends s_1' to Alice and Bob as the output of the CDS protocol. This step can run in parallel with the step above where Claire can send (s_0', s_1').
  - Upon receiving (s_0', s_1') from Claire, Alice/Bob outputs t if $(s_0', s_1') = (s_0, s_1)$ and outputs "ABORT" otherwise.

Fig. 8. 2-round PSI cardinality.

$\mathsf{CDS}_{\leq t}$ for checking lower and upper bounds. Although we abstract the protocol differently, the end result is very similar to the protocol from [16], which can also be proven secure in the presence of malicious corruptions by replacing the PRF we use for the CDS with a pseudorandom permutation (PRP).

Theorem 5.2. *Assume the existence of pseudorandom permutations, then protocol* $\mathsf{PSI-CA}$ *from Fig. 8 is a 2-round two-clients verifiable computation for PSI cardinality in the presence of malicious parties.*

5.3 MVC for PSI

We conclude this section with a verifiable construction for PSI (given in Fig. 9) which extends the PSI cardinality from the previous section. In more detail, our scheme is built on the equality CDS scheme $\mathsf{CDS}_{\text{EQ}_2}$ from Fig. 2, taking a different approach than the PSI from [16] (which we build our PSI cardinality on). In the first phase, Claire provides the list of elements in the intersection together with an equality proof. This is required to ensure that the only attack Claire can carry out is excluding elements from the set. In particular, every

PSI_1

- **Inputs**. The MVC for PSI protocol is invoked by the interface $\mathsf{PSI}(X_1, X_2)$ where X_1 and X_2 are the respective inputs of Alice and Bob of sizes n and m.
- **Setup.** $(s, s_{\text{poly}}, r_1, r_2, r_{\text{poly}}, r_{\text{PRP}}, r_{\text{PRP}_2}) \leftarrow \mathsf{Setup}(1^\kappa)$.
  An ideal functionality $\mathcal{F}_{\text{COIN}}$ is invoked to generate the shared randomness $(s, s_{\text{poly}}, r_1, r_2, r_{\text{poly}}, r_{\text{PRP}}, r_{\text{PRP}_2})$.
- **Protocol.**
  - A PRP key k and k_2 is generated using respective randomness r_{PRP} and r_{PRP_2}.
  - Alice and Bob encode their inputs as $\mathcal{X}_1 = \{F_k(x_1) \mid x_1 \in X_1\}$ and $\mathcal{X}_2 = \{F_k(x_2) \mid x_2 \in X_2\}$.
  - For each element $\mathbf{x}_1 \in \mathcal{X}_1$, Alice generates:
    * $s^{(\mathbf{x}_1)} = F_{k_2}(s\|\mathbf{x}_1)$.
    * $r_1^{(\mathbf{x}_1)} = F_{k_2}(r_1\|\mathbf{x}_1)$.
    * $r_2^{(\mathbf{x}_1)} = F_{k_2}(r_2\|\mathbf{x}_1)$.
  - For each element $\mathbf{x}_2 \in \mathcal{X}_2$, Bob generates:
    * $s^{(\mathbf{x}_2)} = F_{k_2}(s\|\mathbf{x}_2)$.
    * $r_1^{(\mathbf{x}_2)} = F_{k_2}(r_1\|\mathbf{x}_2)$.
    * $r_2^{(\mathbf{x}_2)} = F_{k_2}(r_2\|\mathbf{x}_2)$.
  - Alice generates

$$\tilde{X}_1 = \{(\mathbf{x}_1, \mathsf{CDS}_{\text{EQ}_2} : \mathsf{Enc}_1(\mathbf{x}_1, s^{(\mathbf{x}_1)}, (r_1^{(\mathbf{x}_1)}, r_2^{(\mathbf{x}_2)}, r_{\text{PRP}}))) \mid \mathbf{x}_1 \in \mathcal{X}_1\}.$$

  - Bob generates

$$\tilde{X}_2 = \{(\mathbf{x}_2, \mathsf{CDS}_{\text{EQ}_2} : \mathsf{Enc}_2(\mathbf{x}_2, s^{(\mathbf{x}_2)}, (r_1^{(\mathbf{x}_2)}, r_2^{(\mathbf{x}_2)}, r_{\text{PRP}}))) \mid \mathbf{x}_2 \in \mathcal{X}_2\}.$$

  - Alice and Bob send their respective messages $\tilde{X}_1$ and $\tilde{X}_2$ to Claire.
  - Claire generates $\tilde{X} = \{(\mathbf{x}, {}_1, {}_2) \mid (\mathbf{x}, {}_1) \in \tilde{\mathtt{X}}_1$ and $(\mathbf{x}, {}_2) \in \tilde{\mathtt{X}}_2\}$.
  - Claire generates a set $S = \{(\mathbf{x}, s) \mid \forall(\mathbf{x}, {}_1, {}_2) \in \tilde{\mathtt{X}}$ where each element $s = \mathsf{CDS}_{\text{EQ}_2} : \mathsf{Dec}({}_1, {}_2) \neq \perp\}$ and sends S to Alice and Bob.
  - Alice and Bob calculate $t = |S|$ and check if all $(\mathbf{x}, s^{(\mathbf{x})}) \in S$ satisfy $F_{k_2}(s\|\mathbf{x}) = s^{(\mathbf{x})}$. If any check fails, they output "ABORT".
  - If Alice/Bob doesn't output "ABORT", PSI $= \{v | (\mathbf{x}, s) \in S$ and $F_k(v) = \mathbf{x}\}$.
  - Alice, Bob and Claire run $\mathsf{CDS}_{\leq t}(X_1, X_2, s_{\text{poly}}, (r_{\text{poly}}, r_{\text{PRP}}))$. Claire sends s'_{poly} to Alice and Bob as the output of the CDS protocol.
  - Upon receiving s'_{poly}, Alice/Bob output PSI if $s'_{\text{poly}} = s_{\text{poly}}$ otherwise they output "ABORT".

Fig. 9. 2-round PSI.

party can check whether the declared intersection is part of its input, and is also convinced that the same elements appear within the other party's input, as ensured by the security of the CDS for quality. In the second phase, the parties continue with a PSI cardinality based on the estimated outcome from the first phase.

Theorem 5.3. *Assume the existence of pseudorandom permutations, a Σ-protocol for some predefined NP relation $\mathcal{R}$, then protocol* PSI *from Fig. 9 is a 2-round two-clients verifiable computation for PSI in the presence of malicious parties.*

References

1. Applebaum, B., Arkis, B.: On the power of amortization in secret sharing: d-Uniform secret sharing and CDS with constant information rate. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 317–344. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_12
2. Applebaum, B., Arkis, B., Raykov, P., Vasudevan, P.N.: Conditional disclosure of secrets: amplification, closure, amortization, lower-bounds, and separations. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 727–757. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_24
3. Applebaum, B., Vasudevan, P.N.: Placing conditional disclosure of secrets in the communication complexity universe. In: ITCS, pp. 4:1–4:14 (2019)
4. Boyle, E., Gilboa, N., Ishai, Y.: Secure computation with preprocessing via function secret sharing. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11891, pp. 341–371. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36030-6_14
5. Canetti, R., Paneth, O., Papadopoulos, D., Triandopoulos, N.: Verifiable set operations over outsourced databases. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 113–130. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_7
6. Choi, S.G., Katz, J., Kumaresan, R., Cid, C.: Multi-client non-interactive verifiable computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 499–518. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_28
7. Couteau, G.: New protocols for secure equality test and comparison. In: Preneel, B., Vercauteren, F. (eds.) ACNS 2018. LNCS, vol. 10892, pp. 303–320. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93387-0_16
8. Dong, C., Chen, L., Camenisch, J., Russello, G.: Fair private set intersection with a semi-trusted arbiter. In: Wang, L., Shafiq, B. (eds.) DBSec 2013. LNCS, vol. 7964, pp. 128–144. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39256-6_9
9. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: outsourcing computation to untrusted workers. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 465–482. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_25
10. Gentry, C., Halevi, S., Jutla, C., Raykova, M.: Private database access with HE-over-ORAM architecture. In: Malkin, T., Kolesnikov, V., Lewko, A.B., Polychronakis, M. (eds.) ACNS 2015. LNCS, vol. 9092, pp. 172–191. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-28166-7_9

11. Gertner, Y., Ishai, Y., Kushilevitz, E., Malkin, T.: Protecting data privacy in private information retrieval schemes. J. Comput. Syst. Sci. **60**(3), 592–629 (2000)
12. Ghosh, S., Simkin, M.: The communication complexity of threshold private set intersection. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 3–29. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_1
13. Gordon, S.D., Katz, J., Liu, F.-H., Shi, E., Zhou, H.-S.: Multi-client verifiable computation with stronger security guarantees. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 144–168. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_6
14. Ishai, Y., Kushilevitz, E., Paskin, A.: Secure multiparty computation with minimal interaction. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 577–594. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_31
15. Kamara, S., Mohassel, P., Raykova, M., Sadeghian, S.: Scaling private set intersection to billion-element sets. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014. LNCS, vol. 8437, pp. 195–215. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45472-5_13
16. Le, P.H., Ranellucci, S., Gordon, S.D.: Two-party private set intersection with an untrusted third party. In: CCS (2019)
17. Liu, T., Vaikuntanathan, V., Wee, H.: Conditional disclosure of secrets via non-linear reconstruction. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 758–790. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_25
18. Mohassel, P., Rindal, P.: ABY3: a mixed protocol framework for machine learning. IACR Cryptology ePrint Archive 2018, 403 (2018)
19. Papamanthou, C., Tamassia, R., Triandopoulos, N.: Optimal verification of operations on dynamic sets. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 91–110. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_6
20. Parno, B., Raykova, M., Vaikuntanathan, V.: How to delegate and verify in public: verifiable computation from attribute-based encryption. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 422–439. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_24
21. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptology **4**(3), 161–174 (1991). https://doi.org/10.1007/BF00196725
22. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: FOCS, pp. 162–167 (1986)