Traceable mixnets

Prashant Agrawal
Department of Computer Science and Engineering, IIT Delhi
Centre for Digitalisation, AI and Society, Ashoka University
prashant@cse.iitd.ac.in

Abhinav Nakarmi*
Department of Computer Science and Engineering, University of Michigan
nakarmi@umich.edu

Mahabir Prasad Jhanwar
Department of Computer Science and Centre for Digitalisation, AI and Society, Ashoka University
mahavir.jhawar@ashoka.edu.in

Subodh Vishnu Sharma
Department of Computer Science and Engineering, IIT Delhi
svs@cse.iitd.ac.in

Subhashis Banerjee
Department of Computer Science and Engineering, IIT Delhi
Department of Computer Science and Centre for Digitalisation, AI and Society, Ashoka University
suban@ashoka.edu.in

ABSTRACT

We introduce the notion of *traceable mixnets*. In a traditional mixnet, multiple mix-servers jointly permute and decrypt a list of ciphertexts to produce a list of plaintexts, along with a proof of correctness, such that the association between individual ciphertexts and plaintexts remains completely hidden. However, in many applications, the privacy-utility tradeoff requires answering some specific queries about this association, without revealing any information beyond the query result. We consider queries of the following types: a) given a ciphertext in the mixnet input list, whether it encrypts one of a given subset of plaintexts in the output list, and b) given a plaintext in the mixnet output list, whether it is a decryption of one of a given subset of ciphertexts in the input list. Traceable mixnets allow the mix-servers to jointly prove answers to the above queries to a querier such that neither the querier nor a threshold number of mix-servers learn any information beyond the query result. Further, if the querier is not corrupted, the corrupted mix-servers do not even learn the query result. We first comprehensively formalise these security properties of traceable mixnets and then propose a construction of traceable mixnets using novel distributed zero-knowledge proofs (ZKPs) of set membership and of a statement we call reverse set membership. Although set membership has been studied in the single-prover setting, the main challenge in our distributed setting lies in making sure that none of the mix-servers learn the association between ciphertexts and plaintexts during the proof. We implement our distributed ZKPs and show that they are faster than state-of-the-art by at least one order of magnitude.

KEYWORDS

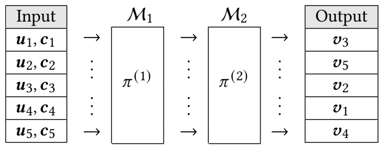
verifiable mixnets; traceability; distributed zero-knowledge proofs; set membership; reverse set membership

1 INTRODUCTION

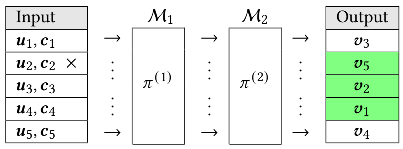
A mixnet is a cryptographic primitive used for anonymous messaging, specifically for unlinking the identity information of data providers with the sensitive data they provide. Let $\boldsymbol{u}_i$ denote the identity information of the i^{th} individual and $\boldsymbol{v}_i$ denote the sensitive data they contribute. For example, in a secure electronic voting context where mixnets are commonly used, $\boldsymbol{u}_i$ may contain a voter id and $\boldsymbol{v}_i$ may contain the vote. The link between $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ is hidden by encrypting $\boldsymbol{v}_i$ to obtain ciphertexts (encrypted votes) $\boldsymbol{c}_i$, uploading $\boldsymbol{u}_i$ along with $\boldsymbol{c}_i$ to an input list, and feeding the list of ciphertexts $\boldsymbol{c}$ as the mixnet input. The mixnet, which consists of a series of *mix-servers*, processes these ciphertexts and outputs a list $\boldsymbol{v}'$ of decryptions of the ciphertexts in a randomly permuted order [23]. The secret permutation that links the ciphertexts with their corresponding plaintexts is shared among the mix-servers and remains hidden unless a threshold number of mix-servers are corrupted, thus completely hiding which ciphertext or voter id corresponds to which vote (see Figure 1a). Using *verifiable mixnets* [53], the mix-servers can also *prove* to a verifier that the output list is obtained correctly by permuting and decrypting each element of the input list, while keeping the linkages between the ciphertexts and plaintexts completely hidden.

However, sometimes, it is necessary to reveal specific partial information about the association between $\boldsymbol{c}_i$ and $\boldsymbol{v}_i$, while still preventing any additional information leakage. For example, consider a large-scale public election running across multiple polling booths. Further, assume a dual voting setup [8, 65] where voters cast their vote both electronically and on paper: the electronic system produces an encrypted vote as the voter receipt and processes these encrypted votes via a mixnet, whereas the paper votes are collected to form a physical audit trail. Such dual voting systems aim to improve the overall robustness and transparency of electoral processes. In such systems, revealing partial information about the encrypted and decrypted votes allows graceful recovery from disputes without re-running the entire election [2]. For example, if there is a mismatch between a plaintext vote in the mixnet output list and its corresponding paper record, the ability to pinpoint which specific booth the disputed vote came from enables a *localised recovery* of the election by selectively rerunning the election only at the corrupted booths. Yet, it is important to not reveal additional information such as vote counts of all the polling booths or, worse, votes of individual voters. Further, any such partial information revealed should be *provable*, to ensure that recovery steps lead to

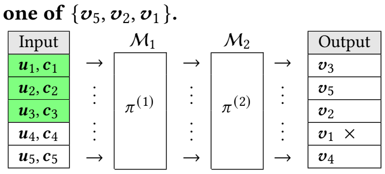
*Work done while at Ashoka University.



(a) A traditional verifiable mixnet.



(b) A TraceIn$(c_2, \{v_5, v_2, v_1\})$ query in a traceable mixnet: whether c_2 encrypted one of $\{v_5, v_2, v_1\}$.



(c) A TraceOut$(\{c_1, c_2, c_3\}, v_1)$ query in a traceable mixnet: whether v_1 was encrypted in one of $\{c_1, c_2, c_3\}$.

Figure 1: Traditional and traceable mixnets. u_i denotes the i^{th} individual's identity information and v_i denotes their sensitive data; c_is encrypt v_is and are passed as input to the mixnet; the mixnet consists of mix-servers $\mathcal{M}_1$ and $\mathcal{M}_2$ that jointly decrypt and permute input list c to output a plaintext list $v' := (v_{\pi(i)})_{i=1}^{5}$, where π is composed of secret permutations $\pi^{(1)}$ and $\pi^{(2)}$ of $\mathcal{M}_1$ and $\mathcal{M}_2$.

the correct election outcome. A traditional mixnet does not allow for the release of such controlled partial information because votes cast at all the booths are anonymised together before decryption.

Conversely, consider a voter claiming that their published encrypted vote does not match the one in their (possibly fake) receipt. If it can be shown that their published encrypted vote decrypted to a plaintext vote that mismatches with its corresponding paper record, it supports the voter's claim because an incorrectly uploaded encrypted vote must also mismatch with the paper vote on decryption. As above, this revealed information should be provable and should not leak any additional information, e.g., the voter's specific vote. Traditional mixnets do not allow this either.

To address this gap, we introduce the notion of traceable mixnets. Let $c_I := \{c_i \mid i \in I\}$ for some index set I denote a subset of ciphertexts in the input list and $v'_J := \{v'_j \mid j \in J\}$ for some index set J denote a subset of plaintexts in the output list. A traceable mixnet allows its mix-servers to jointly and provably answer the following queries to an interested and authorised querier:

- **TraceIn**(c_i, v'_J): Does the ciphertext c_i in the input list encrypt a plaintext in set v'_J (Figure 1b)? Given an encrypted vote c_i mismatching a voter receipt and a set v'_J of plaintext votes mismatching with their paper records, this query answers if c_i decrypted to a mismatching plaintext vote and thus was incorrect.
- **TraceOut**(c_I, v'_j): Is the plaintext v'_j in the output list encrypted in a ciphertext in set c_I (Figure 1c)? Given an output plaintext vote v'_j mismatching with its paper vote and a set c_I of encrypted votes cast at a given polling booth B, this query answers if the disputed vote v'_j came from booth B.

The queries are answered such that no *additional information* beyond the query output is leaked to an adversary controlling the querier and less than a threshold number of mix-servers. Also, to prevent mix-servers from accumulating query responses issued to different queriers over time, the mix-servers are not allowed to even learn the output of a query if they do not control the querier. Such controlled querying mechanism is useful in voting as well as several other applications where privacy-preserving data sharing with guaranteed correctness is required.

Note, though, that the query outputs may themselves leak sensitive information, especially when multiple queries are combined. Thus, arbitrary queries cannot be allowed in any application. Deciding what queries to allow requires a privacy risk analysis of the overall information leaked by the queries, followed by an analysis of whether the leakage is acceptable for the application's privacy requirements. For example, in the voting application, the allowed queries are decided so that the recovery process does not leak the vote counts of any booth except the corrupted booths [2]. Once a policy detailing the allowed queries is decided, possibly allowing different queries to different queriers, the application layer must also ensure that the mix-servers comply with the policy. The traceable mixnet guarantees that if an honest mix-server follows the policy, the adversary learns nothing about the output of the queries disallowed by the policy. Without such analysis and policy compliance though, a traceable mixnet solution should not be deployed.

We emphasise that a traceable mixnet's secrecy requirements are more stringent than what existing proof-of-shuffle and verifiable decryption techniques in the area of verifiable mixnets [53] can provide. For example, given a mixnet input list $(c_1, \dots, c_5)$ and output list $(v'_1, \dots, v'_5)$, a TraceIn$(c_4, \{v'_2, v'_3, v'_4\})$ query requires proving in ZK that c_4 decrypts to one of v'_2, v'_3 and v'_4. Using these techniques, the mix-servers can prove this by, e.g., verifiably revealing the subset $C := \{c_1, c_2, c_4\}$ of the input ciphertexts that decrypt to the set of plaintexts $\{v'_2, v'_3, v'_4\}$ and letting the querier verify that $c_4 \in C$ (see Figure 2b). However, this is not ZK: it reveals, e.g., that ciphertexts c_1, c_2 decrypt to one of v'_2, v'_3 and v'_4 (and not v'_1 or v'_5). Figure 2 shows other similar leakage scenarios. A traceable mixnet does not reveal any such intermediate information.

Finally, our focus is on offline batch processing, i.e., efficiently answering multiple TraceIn/TraceOut queries against the same set. Thus, we define the following batched queries: *a*) **BTraceIn**(c_I, v'_J): which ciphertexts in set c_I encrypt a plaintext in set v'_J; and *b*) **BTraceOut**(c_I, v'_J): which plaintexts in set v'_J are encrypted in a ciphertext in set c_I? We require the amortised time for batched queries to be linear in the input list size to enable practical applications like dispute resolution in elections with millions of votes. Note that offline batch processing in verifiable mixnets is distinct from mixnets in real-time anonymous communication networks [36]. Thus, we do not aim to answer the queries in real time.

1.1 Our contributions

Our main contributions are the following. First, we introduce the notion of *traceable mixnets* and formalise their completeness, soundness and secrecy requirements (Section 2).

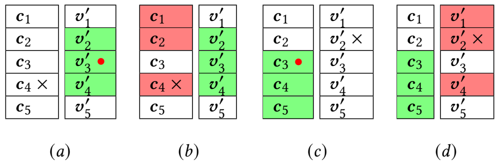


Figure 2: Subfigures (a) and (b) are for a TraceIn(c_4, V) query, where $V = \{v'_2, v'_3, v'_4\}$. With proofs-of-shuffle and verifiable decryption, the mix-servers can either a) verifiably reveal the plaintext encrypted by c_4, say v'_3, and let the querier check if $v'_3 \in V$, or b) verifiably reveal the set of ciphertexts encrypting set V, say $C := \{c_1, c_2, c_4\}$, and let the querier check if $c_4 \in C$. Subfigures (c) and (d) are for a TraceOut(C, v'_2) query, where $C = \{c_3, c_4, c_5\}$. The mix-servers can either c) verifiably reveal the ciphertext encrypting v'_2, say c_3, and let the querier check if $c_3 \in C$, or d) verifiably reveal the set of plaintexts that set C decrypts to, say $V := \{v'_1, v'_2, v'_4\}$, and let the querier check that $v'_2 \in V$. With traceable mixnets, they do not need to reveal any such intermediate information.

Second, we propose a construction of traceable mixnets (Section 4) using novel distributed ZKPs of *set membership* [19] and a novel primitive called *reverse set membership*. Given a commitment scheme [74] comm with commitment space Γ, message space V and randomness space R, a ZKP of set membership for a commitment $\gamma \in \Gamma$ and a set ϕ of values from V proves that γ commits a member of V. Formally, we denote this as $\rho_{\mathsf{SM}}(\gamma, \phi) := \mathsf{PK}\{(v, r) : \gamma = \mathsf{comm}(v; r) \wedge v \in \phi\}$, a proof of knowledge of a member v of set ϕ and a randomness r such that γ commits v with randomness r. A ZKP of reverse set membership for a value $v \in V$ and a set Φ of commitments from Γ proves that v *is committed by* a member of Φ. Formally, we denote this as $\rho_{\mathsf{RSM}}(\Phi, v) := \mathsf{PK}\{(r) : \gamma = \mathsf{comm}(v; r) \wedge \gamma \in \Phi\}$, a proof of knowledge of a randomness r such that some member γ of Φ commits v with randomness r. Prior work has mainly focused on the ZKP of set membership and the single-prover case, whereas our ZKPs (both for set membership and the novel reverse set membership) work in a distributed setting where the mixnet's mix-servers jointly act as the provers. The mix-servers need to carry out the proof without themselves learning any information about either the commitment openings or the association between commitments and plaintext values. Our ZKPs are interactive, which is acceptable for our inherently interactive use-case.

Third, we provide detailed security analysis of our construction and formal rules for privacy risk analysis of a given set of allowed TraceIn/TraceOut queries (Section 5 and Appendices C-D).

Fourth, we provide a comprehensive implementation of our proposal (Section 6). Our construction has linear time complexity in the size of the mixnet input list for batched queries and greatly outperforms even single-prover existing techniques. Specifically, our distributed ZKPs of set membership and reverse set membership (which enable BTraceIn and BTraceOut respectively) have per-prover proving times respectively 43x and 9x faster than single-prover zkSNARK-plus-Merkle tree based proofs. By conservative estimates, this makes them at least 86x and 18x faster than the state-of-the-art collaborative zkSNARKs [72] in the distributed setting. Our implementation is open source and available at [1].

1.2 Related work

1.2.1 Controlled information release in statistical databases. A common approach for controlled information release for statistical analytics is by *anonymising* (releasing a noisy version of) the dataset. Another approach is *differential privacy* (DP) [37]: interactively providing noisy answers to analytics queries such that the answer distribution is insensitive to any given individual's data. However, anonymisation is a known poor safeguard against re-identification attacks [31, 69]. Further, because of correlations in different individuals' data items, differentially private mechanisms can still reveal arbitrary partial information about the link between users' identity information and sensitive data to an adversary running arbitrary queries [84]. In comparison, traceable mixnets reveal answers to only pre-approved queries and otherwise keep users' identity information and sensitive data unlinkable. Also, their distributed setting naturally fits into privacy-sensitive applications, whereas most anonymisation/DP solutions assume a trusted data curator.

1.2.2 Existing traceability notions. Many works [5, 25, 54, 78–80] aim to balance anonymity with accountability by letting users communicate anonymously by default but allowing a trusted third party to revoke the anonymity of users that misbehave by sending illegal, misinformative or offensive messages. This is done by tracing the exact senders of the offending messages, in contrast to our generalised set-based and bidirectional notion of traceability.

Group/ring signatures [22, 75] let a verifier verify that a message was sent by a member of a group, without learning which member. This resembles our TraceOut query if we map the group of senders with the subset of input ciphertexts. However, these signatures require active involvement of the senders and are not logistically suitable for backend analytics applications. Likewise, anonymous credentials [20] let individuals prove in ZK that they satisfy some eligibility criteria, resembling our TraceIn queries, but they also require individuals' active involvement in securing their anonymity, whereas this is done by distributed mix-servers in traceable mixnets.

1.2.3 Existing verifiable mixnets. Haines and Müller [53] review and identify the following existing techniques for building verifiable mixnets: *message tracing* [61, 82], *verification codes* [61, 76], *trip wires* [17, 58], *message replication* [58], *randomised partial checking (RPC)* [55, 59, 62, 63] and *proofs of shuffle* [70, 77, 83]. These techniques only verify that the mixnet output list was a decryption and permutation of its input ciphertext list and do not support the fine-grained TraceIn/TraceOut queries. *Message tracing* and *verification codes* provide limited traceability by letting senders verify the processing of their *own* ciphertexts, but one who does not hold the ciphertext secrets cannot perform this verification.

Proofs of shuffle are the state-of-the-art verifiable mixnet techniques. A proof-of-shuffle proves in ZK that two ciphertext lists are permutations and re-encryptions of each other. This, combined with verifiable decryption techniques [44], provides a ZKP that a list of plaintexts is a decryption and permutation of a list of ciphertexts. These approaches can also prove that a *sublist* of the plaintext list is a decryption and permutation of its corresponding sublist in the input ciphertext list. However, as shown in Figure 2, they leak extra information beyond TraceIn/TraceOut query outputs.

Note that unlike non-interactive verifiable mixnets, an interactive verifiable/traceable mixnet necessarily requires the mix-servers to store their permutations. Thus, forward secrecy of query results is not maintained under future compromise of stored permutations.

1.2.4 Set membership proofs. Below we review set membership and reverse set membership ZKP techniques, first for a single prover.

Techniques with quadratic complexity. Cramer et al. [28] propose a generic technique to create a zero-knowledge Σ-protocol for the OR composition of two statements, given Σ-protocols for each individual statement. Both ZKPs of set membership and reverse set membership can be constructed using this technique: $\rho_{\mathsf{SM}}(\gamma, \phi) := \mathsf{PK}\{(r) : \bigvee_{v \in \phi} \gamma = \mathsf{comm}(v; r)\}$ and $\rho_{\mathsf{RSM}}(\Phi, v) := \mathsf{PK}\{(r) : \bigvee_{\gamma \in \Phi} \gamma = \mathsf{comm}(v; r)\}$. However, for proving $\rho_{\mathsf{SM}}(\gamma, \phi)$ for multiple γs against the same set ϕ or $\rho_{\mathsf{RSM}}(\Phi, v)$ for multiple vs against the same set Φ (the "batched" queries), it results in an overall $O(n^2)$ complexity, where $n \approx |\Phi|, |\phi|$. Groth and Kohlweiss [51] propose a ZKP of knowledge of the form $\rho_{\mathsf{RSM\text{-}0}} := \mathsf{PK}\{(r) : \bigvee_{\gamma \in \Phi} \gamma = \mathsf{comm}(0; r)\}$, i.e., a proof that one of the commitments in a set commits to 0, with $O(\log(n))$ communication complexity. Interestingly, $\rho_{\mathsf{RSM\text{-}0}}$ can be used to prove both ρ_{SM} and ρ_{RSM} [51], resulting in an $O(n \log(n))$ communication complexity for the batched queries. However, the *computational* complexity (for both the prover and the verifier) remains $O(n^2)$.

Accumulator-based techniques. Cryptographic accumulators [9, 68, 71] enable efficient ZKPs of set membership. An accumulator scheme $(\mathsf{Acc}, \mathsf{GenWitness}, \mathsf{AccVer})$ allows computing a short digest A_ϕ to a large set ϕ as $A_\phi \leftarrow \mathsf{Acc}(\phi)$ and a short membership witness w_v for a member $v \in \phi$ as $w_v \leftarrow \mathsf{GenWitness}(A_\phi, v)$ such that $\mathsf{AccVer}(A_\phi, v, w_v) = 1$ is a proof that $v \in \phi$. A ZKP of set membership can thus be constructed by requiring both prover and verifier to compute A_ϕ, the prover to compute w_v and both to then engage in $\rho_{\mathsf{SM\text{-}Acc}}(\gamma, A_\phi) := \mathsf{PK}\{(v, r, w_v) : \gamma = \mathsf{comm}(v; r) \wedge \mathsf{AccVer}(A_\phi, v, w_v) = 1\}$. If the accumulator scheme allows commitments to be set members, a ZKP of reverse set membership can be similarly constructed using $\rho_{\mathsf{RSM\text{-}Acc}}(v, A_\Phi) := \mathsf{PK}\{(r, w) : \mathsf{AccVer}(A_\Phi, \mathsf{comm}(v; r), w) = 1\}$, where w denotes the membership witness of the commitment $\mathsf{comm}(v; r) \in \Phi$.

A popular approach for accumulator-based ZKPs of set membership involves Merkle accumulators [68] as the accumulator scheme and zkSNARKs [24, 45, 50] as the ZK proof system. With Merkle accumulators, both GenWitness and AccVer take $O(\log n)$ time, where $n = |\phi|$, which allows n set membership and reverse set membership proofs in $O(n \log(n))$ time. This approach can also generically support $\rho_{\mathsf{RSM\text{-}Acc}}$ by computing Merkle accumulators for the set of commitments. However, it involves expensive hash computations inside the zkSNARK circuit. Benarroch et al. [12] present a non-generic ZKP of set membership using RSA accumulators, avoiding these expensive hash computations. However, the technique does not support reverse set membership. Also, computing witness w_v for RSA accumulators takes $O(n)$ time, which makes it $O(n^2)$ for batched queries. Techniques to efficiently batch multiple membership proofs together also exist [15, 21].

Extending to the distributed setting. All the above techniques work on the single-prover case. Extending them to our distributed setting where none of the provers (the mix-servers) know either the commitment openings or the permutation between the commitments and the plaintexts is non-trivial. The batching techniques [15, 21] also fail in the distributed setting since they require the prover to know upfront which entries pass the membership proof.

Collaborative zkSNARKs [72] allow a distributed set of provers holding secret shares of a SNARK witness to prove joint knowledge of the same. They add roughly 2x overhead in per-prover proving time over the standard zkSNARKs [72]. DPZKs [32] provide similar guarantees. However, even to securely obtain shares of the SNARK witness, an extra MPC protocol is likely required.

Signature-based set membership. Camenisch et al. [19] initiated a different approach to ZKPs of set membership: the verifier provides to the prover signatures on all members of the set under a fresh signing key generated by the verifier, and the prover proves knowledge of a signature on the committed value in ZK. This proves set membership because if the commitment commits a value outside the set, the prover does not obtain a signature on it and must forge it. This approach gives $O(n)$ batched query complexity because verifier signatures can be reused. We extend these ZKPs by a signature-based ZKP of *reverse set membership* and by distributed signature-based ZKPs for both set membership and reverse set membership.

2 FORMAL DEFINITIONS

We now formalise traceable mixnets. We directly present the batched BTraceIn/BTraceOut protocols as that is our main focus (the TraceIn/TraceOut protocols are trivial special cases). Also, for simplicity, we present the special case when all $t = m$ mix-servers are required to decrypt the ciphertexts but any set of less than t mix-servers cannot break secrecy. Extension to a general t is possible with standard threshold cryptography techniques [34]. Finally, we assume an authenticated broadcast channel that ensures authenticity, availability and non-repudiability of all published messages.

Notation. Given a positive integer n, we denote the set $\{1, \ldots, n\}$ by $[n]$. We let (boldface) $\boldsymbol{x} \in X^{\boldsymbol{n}}$ denote an n-length vector of values drawn from a set X, $\boldsymbol{x}_i$ denote the i^{th} component of $\boldsymbol{x}$, and, given an index set $I \subseteq [n]$, $\boldsymbol{x}_I$ denote the set $\{\boldsymbol{x}_i \mid i \in I\}$. For any scalar binary operation $\odot : X \times Y \to Z$ and vectors $\boldsymbol{x} \in X^{\boldsymbol{n}}$, $\boldsymbol{y} \in Y^{\boldsymbol{n}}$, we let $\boldsymbol{x} \odot \boldsymbol{y}$ denote the vector $(\boldsymbol{x}_i \odot \boldsymbol{y}_i)_{i \in [n]}$, i.e., the vector obtained by component-wise application of $\odot$. We let $\boldsymbol{u}^{\boldsymbol{v}}$ denote $(\boldsymbol{u}_i^{\boldsymbol{v}_i})_{i \in [n]}$, $\boldsymbol{f}(\boldsymbol{v})$ denote $(f(\boldsymbol{v}_i))_{i \in [n]}$, $\boldsymbol{f}(x, \boldsymbol{v})$ denote $(f(x, \boldsymbol{v}_i))_{i \in [n]}$, etc.

We denote a multiparty computation protocol P between parties $\mathcal{P}_1, \ldots, \mathcal{P}_m$ where the common input of each party is ci, $\mathcal{P}_k$'s secret input is si_k, the common output is co and $\mathcal{P}_k$'s secret output is so_k as follows: $co, (\mathcal{P}_k[[so_k]])_{k \in [m]} \leftarrow P(ci, (\mathcal{P}_k[[si_k]])_{k \in [m]})$. When a party does not have a secret output, we drop it from the left-hand side. In security experiments where the experimenter plays the role of honest parties $(\mathcal{P}_k)_{k \in H}$ for some $H \subset [m]$ and an adversary $\mathcal{A}$ plays the role of $(\mathcal{P}_k)_{k \in [m] \setminus H}$, we indicate it as $co, (\mathcal{P}_k[[so_k]])_{k \in H} \leftarrow P(ci, (\mathcal{P}_k[[si_k]])_{k \in H}, (\mathcal{P}_k{}^{\mathcal{A}})_{k \in [m] \setminus H})$. We let $x^{(k)}$ denote a component (typically, secret-share) of x designated for $\mathcal{P}_k$. We call a function f *negligible* if for any polynomial p, there exists an $N \in \mathbb{N}$ such that $f(x) < 1/p(x)$ for all $x > N$.

Definition 1 (Traceable mixnets). *A* traceable mixnet *is a tuple of protocols/algorithms* (Keygen, Enc, Mix, BTraceIn, BTraceOut) *between n senders* $(S_i)_{i\in[n]}$, *m mix-servers* $(\mathcal{M}_k)_{k\in[m]}$ *and a querier or verifier* Q *such that:*

- $\mathsf{mpk}, (\mathcal{M}_k[[\mathsf{msk}^{(k)}]])_{k\in[m]} \leftarrow \mathsf{Keygen}(1^\lambda, (\mathcal{M}_k[[]])_{k\in[m]})$ *is a key generation protocol between* $(\mathcal{M}_k)_{k\in[m]}$, *where* λ *is a security parameter (given in unary), individual mix-servers do not have any secret input, the common output is a mixnet public key* mpk *and each* $\mathcal{M}_k$*'s secret output is a secret key* $\mathsf{msk}^{(k)}$.
- $\boldsymbol{c}_i \leftarrow \mathsf{Enc}(\mathsf{mpk}, \boldsymbol{v}_i)$ *is an algorithm run by sender* S_i, *where* mpk *is the mixnet public key and* $\boldsymbol{v}_i$ *is* S_i*'s sensitive input drawn from some plaintext space* $\mathbb{V}$. *Output* $\boldsymbol{c}_i$ *is a ciphertext "encrypting"* $\boldsymbol{v}_i$.
- $\boldsymbol{v}', (\mathcal{M}_k[[\omega^{(k)}]])_{k\in[m]} \leftarrow \mathsf{Mix}(\mathsf{mpk}, \boldsymbol{c}, (\mathcal{M}_k[[\mathsf{msk}^{(k)}]])_{k\in[m]})$ *is a mixing protocol between* $(\mathcal{M}_k)_{k\in[m]}$, *where* mpk *is the mixnet public key and* $\boldsymbol{c} \leftarrow (\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{v}_i))_{i\in[n]}$ *is a vector of ciphertexts encrypting the senders' plaintexts* $(\boldsymbol{v}_i)_{i\in[n]}$. *Each* $\mathcal{M}_k$*'s secret input is its secret key* $\mathsf{msk}^{(k)}$. *The common output is a vector* $\boldsymbol{v}'$ *of plaintext values obtained after permuting and decrypting* $\boldsymbol{c}$ *(thus* $\boldsymbol{v}' = (\boldsymbol{v}_{\pi(i)})_{i\in[n]}$ *for some permutation* π*). Each* $\mathcal{M}_k$*'s secret output is a witness* $\omega^{(k)}$ *to be used in proving correctness of the* BTraceIn/BTraceOut *outputs (see below).*
- $Q[[\boldsymbol{c}_{I^*}]] \leftarrow \mathsf{BTraceIn}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{msk}^{(k)}, \omega^{(k)}]])_{k\in[m]}, Q[[]])$ *is a protocol between* $(\mathcal{M}_k)_{k\in[m]}$ *and querier* Q, *where* $\boldsymbol{v}', \mathcal{M}_k[[\omega^{(k)}]] \leftarrow \mathsf{Mix}(\mathsf{mpk}, \boldsymbol{c}, \mathcal{M}_k[[\mathsf{msk}^{(k)}]])$, $\boldsymbol{c} \leftarrow (\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{v}_i))_{i\in[n]}$ *and* $I, J \subseteq [n]$. Q*'s secret output is a set of ciphertexts* $\boldsymbol{c}_{I^*} = \{\boldsymbol{c}_i \in \boldsymbol{c}_I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_J\}$; Q *may abort if it is not convinced about the correctness of* $\boldsymbol{c}_{I^*}$.
- $Q[[\boldsymbol{v}'_{J^*}]] \leftarrow \mathsf{BTraceOut}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{msk}^{(k)}, \omega^{(k)}]])_{k\in[m]}, Q[[]])$ *is a protocol between* $(\mathcal{M}_k)_{k\in[m]}$ *and* Q, *where all inputs are exactly the same as* BTraceIn *and* Q*'s secret output is a set of plaintexts* $\boldsymbol{v}'_{J^*} = \{\boldsymbol{v}'_j \in \boldsymbol{v}'_J \mid \boldsymbol{v}'_j \in \boldsymbol{v}_I\}$; Q *may abort if it is not convinced about the correctness of* $\boldsymbol{v}'_{J^*}$.

$\underline{\mathsf{Exp}_{\mathsf{completeness}}(1^\lambda, n, \boldsymbol{v}, I, J):}$
1 $\mathsf{mpk}, (\mathcal{M}_k[[\mathsf{msk}^{(k)}]])_{k\in[m]} \leftarrow \mathsf{Keygen}(1^\lambda, (\mathcal{M}_k[[]])_{k\in[m]})$
2 $\boldsymbol{c} := (\boldsymbol{c}_i)_{i\in[n]} \leftarrow (\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{v}_i))_{i\in[n]}$ // $\boldsymbol{c}_i$ encrypts $\boldsymbol{v}_i$
3 $\boldsymbol{v}', (\mathcal{M}_k[[\omega^{(k)}]])_{k\in[m]} \leftarrow \mathsf{Mix}(\mathsf{mpk}, \boldsymbol{c}, (\mathcal{M}_k[[\mathsf{msk}^{(k)}]])_{k\in[m]})$
4 $Q[[\boldsymbol{c}_{I^*}]] \leftarrow \mathsf{BTraceIn}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{msk}^{(k)}, \omega^{(k)}]])_{k\in[m]}, Q[[]])$
5 $Q[[\boldsymbol{v}'_{J^*}]] \leftarrow \mathsf{BTraceOut}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{msk}^{(k)}, \omega^{(k)}]])_{k\in[m]}, Q[[]])$
6 **return** 1 if all $\boldsymbol{c}_i$s and all $\boldsymbol{v}'_j$s are distinct, and
7 1) $\boldsymbol{c}_{I^*} = \{\boldsymbol{c}_i \in \boldsymbol{c}_I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_J\}$; and
8 2) $\boldsymbol{v}'_{J^*} = \{\boldsymbol{v}'_j \in \boldsymbol{v}'_J \mid \boldsymbol{v}'_j \in \boldsymbol{v}_I\}$ // $\boldsymbol{c}_I = \{\mathsf{Enc}(pk, v) \mid v \in \boldsymbol{v}_I\}$

Figure 3: Completeness experiment

$\underline{\mathsf{Exp}^{\mathcal{A}}_{\mathsf{soundness}}(1^\lambda):}$
1 $\mathsf{mpk} \leftarrow \mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k\in[m]})$
2 $\boldsymbol{v} := (\boldsymbol{v}_i)_{i\in[n]} \leftarrow \mathcal{A}()$
3 $\boldsymbol{c} := (\boldsymbol{c}_i)_{i\in[n]} \leftarrow (\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{v}_i))_{i\in[n]}$
4 $\boldsymbol{v}' \leftarrow \mathsf{Mix}(\mathsf{mpk}, \boldsymbol{c}, (\mathcal{M}_k^{\mathcal{A}})_{k\in[m]})$
5 $I, J \leftarrow \mathcal{A}(\boldsymbol{v}')$
6 $Q[[\boldsymbol{c}_{I^*}]] \leftarrow \mathsf{BTraceIn}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k^{\mathcal{A}})_{k\in[m]}, Q[[]])$
7 $Q[[\boldsymbol{v}'_{J^*}]] \leftarrow \mathsf{BTraceOut}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, (\mathcal{M}_k^{\mathcal{A}})_{k\in[m]}, Q[[]])$
8 **return** 1 if all $\boldsymbol{v}_i$s and all $\boldsymbol{v}'_j$s are distinct, and either
9 1) $\boldsymbol{c}_{I^*} \neq \{\boldsymbol{c}_i \in \boldsymbol{c}_I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_J\}$; or
10 2) $\boldsymbol{v}'_{J^*} \neq \{\boldsymbol{v}'_j \in \boldsymbol{v}'_J \mid \boldsymbol{v}'_j \in \boldsymbol{v}_I\}$ // $\boldsymbol{c}_I = \{\mathsf{Enc}(pk, v) \mid v \in \boldsymbol{v}_I\}$

Figure 4: Soundness experiment

2.1 Completeness

The completeness definition for traceable mixnets (Definition 2 and Figure 3) models that when all the parties are honest then *a)* Q's output $\boldsymbol{c}_{I^*}$ on a BTraceIn query on $(\boldsymbol{c}, \boldsymbol{v}', I, J)$ is exactly the set of ciphertexts in $\boldsymbol{c}_I$ that encrypted some plaintext in $\boldsymbol{v}'_J$ (line 7), and *b)* its output $\boldsymbol{v}'_{J^*}$ on a BTraceOut query on $(\boldsymbol{c}, \boldsymbol{v}', I, J)$ is exactly the set of plaintexts in $\boldsymbol{v}'_J$ that were encrypted by some ciphertext in $\boldsymbol{c}_I$ (line 8). Note that we only consider the case of distinct values. The case of repeated values is trivially reducible to this case if S_i prefixes $\boldsymbol{v}_i$ with a nonce drawn uniformly from a large set.

Definition 2 (Completeness). *A traceable mixnet is* complete *if for each security parameter* $\lambda \in \mathbb{N}$, *number of ciphertexts* $n \in \mathbb{N}$, *vector* $\boldsymbol{v} \in \mathbb{V}^n$ *of distinct plaintext values and index sets* $I, J \subseteq [n]$, *there exists a negligible function* negl *such that*

$$\Pr[\mathsf{Exp}_{\mathsf{completeness}}(1^\lambda, n, \boldsymbol{v}, I, J) = 1] \geq 1 - \mathsf{negl}(\lambda), \qquad (1)$$

where $\mathsf{Exp}_{\mathsf{completeness}}$ *is as defined in Figure 3.*

2.2 Soundness

The soundness definition (Definition 3) models that as long as input ciphertexts are well-formed, even if all the mix-servers are dishonest, sets $\boldsymbol{c}_{I^*}$ and $\boldsymbol{v}'_{J^*}$ output by Q in BTraceIn and BTraceOut respectively must be "correct," where the correctness of $\boldsymbol{c}_{I^*}$ and $\boldsymbol{v}'_{J^*}$ is exactly as defined in $\mathsf{Exp}_{\mathsf{completeness}}$. Thus, we do not allow the cheating mix-servers to force Q to produce incorrect output (Q may abort, though). We only consider well-formed input ciphertexts because "correct" processing of ill-formed inputs is undefined. Nevertheless, proofs of well-formedness of inputs are generally required for application-level correctness and these can be constructed by the senders at the time of uploading their ciphertexts.

The experiment (Figure 4) begins with the key generation protocol where the adversary $\mathcal{A}$ controlling all the mix-servers $(\mathcal{M}_k)_{k\in[m]}$ provides the mixnet public key mpk (line 1). We let $\mathcal{A}$ supply the plaintexts but create ciphertexts from them honestly (lines 2-3), to model that $\mathcal{A}$ can supply plaintexts of its choice to cheat but the ciphertexts must be well-formed. We then allow $\mathcal{A}$ to run the Mix protocol and produce the output list $\boldsymbol{v}'$ (line 4). $\mathcal{A}$ then outputs index sets I and J on which it wants to break the subsequent BTraceIn/BTraceOut queries (line 3). During these queries, Q outputs $\boldsymbol{c}_{I^*}$ and $\boldsymbol{v}'_{J^*}$ respectively (lines 6-7). $\mathcal{A}$ wins if it supplies distinct entries and Q produces valid outputs $\boldsymbol{c}_{I^*}, \boldsymbol{v}'_{J^*}$ (does not abort) but at least one of them is incorrect (lines 8-10).

Definition 3 (Soundness). *A traceable mixnet is* sound *if for each PPT adversary* $\mathcal{A}$ *and security parameter* $\lambda \in \mathbb{N}$, *there exists a negligible function* negl *such that*

$$\Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{soundness}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda), \qquad (2)$$

where $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{soundness}}$ *is as defined in Figure 4.*

2.3 Secrecy

The secrecy definition (Definition 4) extends a standard anonymity property [10, 11, 13, 33] to the case when the BTraceIn/BTraceOut queries are also allowed. The standard anonymity property can be stated for our setting as follows: an adversary controlling all-but-two senders, the querier and any set of less than m mix-servers should not be able to distinguish between a world where ciphertexts (c_0, c_1) sent by the two honest senders encrypt values (v_0, v_1) (world 0) and the world where they encrypt (v_1, v_0) (world 1). When BTraceIn/BTraceOut queries are allowed, distinguishing between the two worlds is trivial because of the query outputs (e.g., if I, J given to a BTraceIn query are such that $c_0, c_1 \in \boldsymbol{c}_I$ and $v_0 \in \boldsymbol{v}'_J$ but $v_1 \notin \boldsymbol{v}'_J$ then $\mathcal{A}$ immediately knows it is in world 0 if output $\boldsymbol{c}_{I^*}$ includes c_0). Thus, we require that *a)* in all the BTraceIn queries either both $v_0, v_1 \in \boldsymbol{v}'_J$ or both $v_0, v_1 \notin \boldsymbol{v}'_J$ and *b)* in all the BTraceOut queries either both $c_0, c_1 \in \boldsymbol{c}_I$ or both $c_0, c_1 \notin \boldsymbol{c}_I$.

In more detail, in this experiment (Figure 5), adversary $\mathcal{A}$ engages in the key generation protocol where it controls all the mix-servers except one, i.e., $\mathcal{M}_{k^*}$ (line 1). It then supplies input ciphertexts for all the senders except the two that it does not control, say S_{i_0} and S_{i_1}. For these senders, it supplies the values v_0, v_1 (line 2). In world 0 $(b = 0)$, S_{i_0}'s ciphertext $\boldsymbol{c}_{i_0}$ encrypts v_0 and S_{i_1}'s ciphertext $\boldsymbol{c}_{i_1}$ encrypts v_1; in world 1 $(b = 1)$, this order is reversed (lines 3-4). The ciphertext list thus formed is processed through the Mix protocol, where $\mathcal{A}$ controls all mix-servers except $\mathcal{M}_{k^*}$ and produces an output plaintext list $\boldsymbol{v}'$ (line 5). Then, $\mathcal{A}$ obtains access to oracles OTraceIn, OTraceOut that let it choose I, J, control Q and all mix-servers except $\mathcal{M}_{k^*}$ and interact with $\mathcal{M}_{k^*}$ in the BTraceIn, BTraceOut protocols (lines 6-14). $\mathcal{A}$ is required to respect the condition of including either both or none of the honest senders' ciphertexts/plaintexts in its oracle calls (lines 12 and 18). Finally, $\mathcal{A}$ outputs a bit b' as its guess of the bit b (line 5) and wins if its advantage in making the correct guess is non-negligible.

Definition 4 (Secrecy). *A traceable mixnet protects* secrecy *if for each PPT adversary $\mathcal{A}$, security parameter $\lambda \in \mathbb{N}$, $k^* \in [m]$, and $i_0, i_1 \in [n]$, there exists a negligible function* negl *such that*

$$\begin{aligned}|\Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{secrecy}}(1^\lambda, k^*, i_0, i_1, 0) = 1] - \\ \Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{secrecy}}(1^\lambda, k^*, i_0, i_1, 1) = 1]| \leq \mathsf{negl}(\lambda),\end{aligned} \tag{3}$$

where $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{secrecy}}$ *is as defined in Figure 5.*

Definition 5 below models that when $\mathcal{A}$ does not control Q, it should not even learn the query outputs.

Definition 5 (Output secrecy). *A traceable mixnet protects* output secrecy *if it protects secrecy as per Definition 4 except that in experiment* $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{secrecy}}$ *(Figure 5), $\mathcal{A}$ does not control Q during the* BTraceIn/BTraceOut *calls (lines 10 and 14) and the constraints on lines 9 and 13 are removed.*

Note that Definition 4 also models that if the honest mix-server follows a query policy then the adversary gains no information about the output of queries disallowed by the policy, since this case corresponds to an adversary that simply does not call the OTraceIn or OTraceOut oracles for the disallowed queries. The privacy risk associated with the outputs of allowed queries is outside the scope of formal security requirements of traceable mixnets, but we provide a mechanism to analyse this risk in Appendix D.

$\underline{\mathsf{Exp}^{\mathcal{A}}_{\mathsf{secrecy}}(1^\lambda, k^*, i_0, i_1, b)}$:
1 $\mathsf{mpk}, \mathcal{M}_{k^*}[[\mathsf{msk}^{(k^*)}]] \leftarrow \mathsf{Keygen}(1^\lambda, \mathcal{M}_{k^*}[[]], (\mathcal{M}^{\mathcal{A}}_k)_{k \neq k^*})$
2 $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}} \leftarrow \mathcal{A}(\mathsf{mpk})$
3 **if** $b = 0$: $\boldsymbol{c}_{i_0} \leftarrow \mathsf{Enc}(\mathsf{mpk}, v_0)$; $\boldsymbol{c}_{i_1} \leftarrow \mathsf{Enc}(\mathsf{mpk}, v_1)$
4 **else:** $\boldsymbol{c}_{i_0} \leftarrow \mathsf{Enc}(\mathsf{mpk}, v_1)$; $\boldsymbol{c}_{i_1} \leftarrow \mathsf{Enc}(\mathsf{mpk}, v_0)$
5 $\boldsymbol{v}', \mathcal{M}_{k^*}[[\omega^{(k^*)}]] \leftarrow \mathsf{Mix}(\mathsf{mpk}, (\boldsymbol{c}_i)_{i \in [n]}, \mathcal{M}_{k^*}[[\mathsf{msk}^{(k^*)}]], (\mathcal{M}^{\mathcal{A}}_k)_{k \neq k^*})$
6 **return** $b' \leftarrow \mathcal{A}^{\mathsf{OTraceIn}, \mathsf{OTraceOut}}(\boldsymbol{v}')$
7
8 $\underline{\mathsf{OTraceIn}(I, J)}$:
9 **assert** $(v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J)$
10 $\mathsf{BTraceIn}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, \mathcal{M}_{k^*}[[\mathsf{msk}^{(k^*)}, \omega^{(k^*)}]], (\mathcal{M}^{\mathcal{A}}_k)_{k \neq k^*}, Q^{\mathcal{A}})$
11
12 $\underline{\mathsf{OTraceOut}(I, J)}$:
13 **assert** $(\boldsymbol{c}_{i_0} \in \boldsymbol{c}_I \iff \boldsymbol{c}_{i_1} \in \boldsymbol{c}_I)$
14 $\mathsf{BTraceOut}(\mathsf{mpk}, \boldsymbol{c}, \boldsymbol{v}', I, J, \mathcal{M}_{k^*}[[\mathsf{msk}^{(k^*)}, \omega^{(k^*)}]], (\mathcal{M}^{\mathcal{A}}_k)_{k \neq k^*}, Q^{\mathcal{A}})$

Figure 5: Secrecy experiment

3 PRELIMINARIES

Setup. We assume that the output of the following Setup algorithm is implicitly available to all the parties: $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, f_1, g_1, h_1, f_2, g_2, f_T) \leftarrow \mathsf{Setup}(1^\lambda, m, n)$. Setup takes as input a security parameter $\lambda \in \mathbb{N}$, integers m and n (m represents the number of mix-servers and n represents the number of input ciphertexts) and outputs the following setup parameters: a large prime number q $(q \gg m, n)$, cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order q, generators $\{f_1, g_1, h_1\}$, $\{f_2, g_2\}$ and f_T of groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ respectively, and an efficiently computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ [1]. We assume that the n-Strong Diffie Hellman (SDH) assumption [14] holds in groups $(\mathbb{G}_1, \mathbb{G}_2)$ and that the decisional Diffie-Hellman (DDH) and discrete logarithm (DL) problems are hard in $\mathbb{G}_1$. We assume that all generators are randomly generated, e.g., as the output of a hash function modelled as a random oracle.

3.1 Key cryptographic primitives

3.1.1 (Basic) Boneh-Boyen (BB) signatures (Section 3.1; [14]). In this signature scheme, the signer chooses its secret key (SK) as $x \xleftarrow{\$} \mathbb{Z}_q$ and verification key (VK) as $y \leftarrow g_2^x$. To sign a message $m \in \mathbb{Z}_q$, it computes $\sigma \leftarrow g_1^{\frac{1}{m+x}}$. The signature is verified if $e(\sigma, y g_2^m) \stackrel{?}{=} e(g_1, g_2)$. This scheme is unforgeable against weak chosen message attacks under the n-SDH assumption [14].

3.1.2 BBS+ signatures [4]. In this signature scheme, the signer chooses its SK as $x \xleftarrow{\$} \mathbb{Z}_q^*$ and VK as $y \leftarrow f_2^x$. To sign a message $m \in \mathbb{Z}_q$, it computes $c, r \xleftarrow{\$} \mathbb{Z}_q$ and $S \leftarrow (f_1 g_1^m h_1^r)^{\frac{1}{c+x}}$ and outputs $\sigma := (S, c, r)$. The signature is verified if $e(S, y f_2^c) \stackrel{?}{=} e(f_1 g_1^m h_1^r, f_2)$. The scheme is unforgeable against adaptively chosen message attacks under the n-SDH assumption [4].

3.1.3 Signatures on committed values. BBS+ signatures also let one reveal only a Pedersen commitment [74] $\gamma = g_1^v h_1^r$ to a signer (and a PoK of v, r) and obtain a signature on the *committed value* v:

- The requester sends γ and $\rho_\gamma \leftarrow \mathsf{PK}\{(v, r) : \gamma = g_1^v h_1^r\}$ to the signer. The signer verifies ρ_γ.

[1]For all $a, b \in \mathbb{Z}_q$ and generators g_1, g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ and $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ denotes the identity element of $\mathbb{G}_T$.

- The signer (with SK x and VK $y = f_2^x$) computes a *quasi BBS+ signature* by choosing $c, \hat{r} \xleftarrow{\$} \mathbb{Z}_q$ and computing $S \leftarrow (f_1 h_1^{\hat{r}} \gamma)^{\frac{1}{c+x}}$. It sends $\hat{\sigma} := (S, c, \hat{r})$ to the requester.
- The requester computes $\sigma \leftarrow (S, c, \hat{r} + r)$. ($\sigma = ((f_1 h_1^{\hat{r}+r} g_1^v)^{\frac{1}{c+x}}, c, \hat{r} + r)$ is a valid BBS+ signature on message v under VK y.)

$\hat{\sigma} := (S, c, \hat{r})$ can be verified by checking if $e(S, y f_2^c) \stackrel{?}{=} e(f_1 h_1^{\hat{r}} \gamma, f_2)$.

3.1.4 (m, m)-threshold secret sharing. We consider the following standard (m, m)-threshold secret sharing scheme where Share allows sharing a secret x among m parties and Recons allows its reconstruction by all of them (fewer parties do not learn x):

- $(x^{(k)})_{k\in[m]} \leftarrow \mathsf{Share}_{m,m}(x \in \mathbb{Z}_q) : (x^{(k)})_{k\in[m-1]} \xleftarrow{\$} \mathbb{Z}_q^{m-1}$; $x^{(m)} \leftarrow (x - \sum_{k\in[m-1]} x^{(k)}) \pmod q$
- $x \leftarrow \mathsf{Recons}((x^{(k)})_{k\in[m]}) : x \leftarrow \sum_{k\in[m]} x^{(k)} \pmod q$

A secret sharing scheme is called *additive* (resp. *multiplicative*) if $\mathcal{P}_k$ on input its shares $x^{(k)}, y^{(k)}$ of secrets x and y respectively can obtain its share of $x+y$ (resp. xy) without any additional interaction with other parties. The above scheme is clearly additive. It can also be made multiplicative using Beaver's trick [27], which employs an input-independent precomputation step and an algorithm Mult such that $\mathcal{P}_k$ can obtain its share of xy as $\mathsf{Mult}(x^{(k)}, y^{(k)})$.

3.1.5 (m, m)-threshold proofs of knowledge. An (m, m)-threshold proof of knowledge (also called a *distributed proof of knowledge* or a DPK) is a protocol between provers $(\mathcal{P}_k)_{k\in[m]}$ and a verifier $\mathcal{V}$ that convinces $\mathcal{V}$ that for a given common input x, the provers know secret shares $\omega^{(k)}$ of a secret ω such that a predicate $p(x, \omega)$ is true. We denote these DPKs as $\mathcal{V}[[res]] \leftarrow \mathsf{DPK}(x, p, (\mathcal{P}_k[[\omega^{(k)}]])_{k\in[m]}, \mathcal{V}[[]])$, where $res = 1$ means that $\mathcal{V}$ accepted the proof. The secrecy guarantee is that an adversary $\mathcal{A}$ controlling $\mathcal{V}$ and all $(\mathcal{P}_k)_{k\neq k^*}$ for some k^* cannot learn anything about $\omega^{(k^*)}$ (and thus ω) [57].

We use DPKs where the predicate p is of the form $\bigwedge_{i\in[\ell]} y_i = \prod_{j\in[\ell']} g_{ij}^{\omega_j}$ for $\ell, \ell' \in \mathbb{N}$, public values $y_i, g_{ij} \in \mathbb{G}_1$, $\mathbb{G}_2$ or $\mathbb{G}_T$ and $\omega_j \in \mathbb{Z}_q$. These DPKs can be constructed using standard Σ-protocol techniques [18, 35, 57] if each prover $\mathcal{P}_k$ knows share $\omega_j^{(k)}$ of each ω_j. We use the following NIZK variant obtained using the Fiat-Shamir heuristic [42]:

- $(\mathcal{P}_k)_{k\in[m]}$: Publish $a_i^{(k)} \leftarrow \prod_{j\in[\ell']} g_{ij}^{r_j^{(k)}}$, where $r_j^{(k)} \xleftarrow{\$} \mathbb{Z}_q$.
- $(\mathcal{P}_k)_{k\in[m]}$: Compute $a_i \leftarrow \prod_{k\in[m]} a_i^{(k)}$; $c \leftarrow H(p \| (a_i)_{i\in[\ell]})$, where H is a cryptographic hash function modelled as a random oracle; $z_j^{(k)} \leftarrow r_j^{(k)} - c\omega_j^{(k)} \pmod q$. Send $z_j^{(k)}$ to $\mathcal{V}$.
- $\mathcal{V}$: Obtain $((a_i^{(k)})_{i\in[\ell],k\in[m]}, c, (z_j^{(k)})_{j\in[\ell'],k\in[m]})$ from $(\mathcal{P}_k)_{k\in[m]}$. Compute $a_i \leftarrow \prod_{k\in[m]} a_i^{(k)}$; $z_j \leftarrow \sum_{k\in[m]} z_j^{(k)} \pmod q$. Check $c \stackrel{?}{=} H(p \| (a_i)_{i\in[\ell]})$ and $\bigwedge_{i\in[\ell]} a_i \stackrel{?}{=} y_i^c \prod_{j\in[\ell']} g_{ij}^{z_j}$.

In this variant, if $\mathcal{A}$ controls $(\mathcal{P}_k)_{k\neq k^*}$ but not $\mathcal{V}$, it does not even learn whether the statement was proved successfully or not, because it only sees $a_i^{(k^*)} \leftarrow \prod_{j\in[\ell']} g_{ij}^{r_j^{(k^*)}}$ and not $z_j^{(k^*)}$.

3.1.6 (m, m)-threshold homomorphic encryption. An (m, m)-threshold encryption scheme E^{th} between parties $(\mathcal{P}_k)_{k\in[m]}$ with plaintext space $\mathbb{M}(\mathsf{E}^{\mathsf{th}})$ and ciphertext space $\mathbb{C}(\mathsf{E}^{\mathsf{th}})$ is a tuple (Keygen, Enc, TDec), where Keygen is a key generation protocol, Enc is an encryption algorithm and TDec is a threshold decryption protocol, such that for all $x \in \mathbb{M}(\mathsf{E}^{\mathsf{th}})$, security parameters $\lambda \in \mathbb{N}$, $(\mathsf{pk}, (\mathcal{P}_k[[\mathsf{sk}^{(k)}]])_{k\in[m]}) \leftarrow \mathsf{Keygen}(1^\lambda, \mathcal{P}_k[[]])$, $\mathsf{TDec}(\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}, x), (\mathcal{P}_k[[\mathsf{sk}^{(k)}]])_{k\in[m]}) = x$. IND-CPA security of these schemes is analogous to the IND-CPA security of vanilla public-key encryption schemes [48], where the adversary controls less than m parties.

We use the following threshold encryption schemes: *a*) $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$: the threshold El Gamal encryption scheme [34] where $\mathbb{M}(\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}) = \mathbb{G}_1$ and $\mathbb{C}(\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}) = \mathbb{G}_1 \times \mathbb{G}_1$, and *b*) $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$: an optimised threshold Paillier encryption scheme from Damgård et al. [29] where $\mathbb{M}(\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}) = \mathbb{Z}_N$ for an RSA modulus N and $\mathbb{C}(\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}) = \mathbb{Z}^*_{N^2}$. $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ is multiplicatively homomorphic in $\mathbb{G}_1$: for any two ciphertexts $c_1, c_2 \in \mathbb{G}_1 \times \mathbb{G}_1$ encrypting messages $m_1, m_2 \in \mathbb{G}_1$, $c_1 c_2$ (their component-wise multiplication in $\mathbb{G}_1$) decrypts to the message $m_1 m_2$ (group multiplication in $\mathbb{G}_1$). $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ is additively homomorphic in $\mathbb{Z}_N$: for any two ciphertexts $c_1, c_2 \in \mathbb{Z}^*_{N^2}$ encrypting messages $m_1, m_2 \in \mathbb{Z}_N$, $c_1 c_2 \bmod N^2$ decrypts to the message $m_1 + m_2 \mod N$. However, we require additive homomorphism in $\mathbb{Z}_q$ (a prime order group). Thus, we let $N > q$, intepret messages in $\mathbb{Z}_q$ as messages in $\mathbb{Z}_N$ and carefully use $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$'s homomorphic addition modulo N to obtain homomorphic addition modulo q (see Section 4.3).

Both $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ are IND-CPA secure, respectively under the DDH assumption in $\mathbb{G}_1$ [40] and the decisional composite residuosity (DCR) assumption [73]. Also, both support distributed key generation protocols. For $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$, each party already generates its key shares independently; for $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$, a secure key generation protocol can be designed [30]. Further, the TDec protocols of both schemes provide *simulation security*, i.e, the adversary's view in the TDec protocol can be simulated given access to a decryption oracle.

We also use a standard (non-threshold) IND-CPA secure public-key encryption scheme E on message space $\mathbb{Z}_q$.

3.1.7 Shuffles. Let E^{th} be an (m, m) threshold homomorphic encryption scheme, pk be a public key under E^{th}, $\boldsymbol{\epsilon}$ be an n-length vector of ciphertexts against pk, $\pi^{(1)}, \dots, \pi^{(m)} \in \mathsf{Perm}(n)$ be secret permutations of parties $\mathcal{P}_1, \dots, \mathcal{P}_m$, where $\mathsf{Perm}(n)$ denotes the space of permutation functions, and $\mathsf{E}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}, c)$ be re-encryption under E^{th} of ciphertext c with fresh randomness. We let $\boldsymbol{\epsilon}' \leftarrow \mathsf{Shuffle}(\mathsf{E}^{\mathsf{th}}, \mathsf{pk}, \boldsymbol{\epsilon}, \mathcal{P}_1[[\pi^{(1)}]], \dots, \mathcal{P}_m[[\pi^{(m)}]])$ be a shorthand for repeated re-encryption and permutation of $\boldsymbol{\epsilon}$ by each of $(\mathcal{P}_k)_{k\in[m]}$ in sequence, such that for all $j \in [n]$, $\boldsymbol{\epsilon}'_j = \mathsf{E}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}, \boldsymbol{\epsilon}_{\pi(j)})$, where $\pi = \pi^{(m)} \circ \cdots \circ \pi^{(1)}$. The order of parties in the Shuffle protocol denotes that first $\mathcal{P}_1$ runs, then $\mathcal{P}_2$, etc.

4 OUR CONSTRUCTION

Our traceable mixnet construction extends Camenisch et al.'s single-prover ZKPs of set membership [19] and our novel ZKP of reverse set membership (Section 4.1.2). We explain these protocols first.

4.1 Single prover case

4.1.1 ZKP of set membership [19]. In this ZKP, given a Pedersen commitment [74] γ and a set of values ϕ, a single prover proves knowledge of v, r such that $\gamma = g_1^v h_1^r$ and $v \in \phi$. The main idea is that the verifier generates a *fresh* BB signature key pair $x \xleftarrow{\$} \mathbb{Z}_q, y \leftarrow g_2^x$ and sends to the prover the verification key y and signatures $\sigma_{v'} \leftarrow g_1^{\frac{1}{x+v'}}$ on each $v' \in \phi$. The prover chooses a blinding factor $b \xleftarrow{\$} \mathbb{Z}_q$ and sends to the verifier a blinded version $\tilde{\sigma}_v$ of the signature on the value v committed by γ, as $\tilde{\sigma}_v \leftarrow \sigma_v^b = g_1^{\frac{b}{x+v}}$. Both then engage in a ZKP of knowledge $\mathsf{PK}\{(v, r, b) : \gamma = g_1^v h_1^r \wedge e(\tilde{\sigma}_v, y) = e(g_1, g_2)^b e(\tilde{\sigma}_v, g_2)^{-v}\}$, which proves knowledge of a valid signature $(\tilde{\sigma}_v)^{1/b}$ on the value committed by γ (see Section 3.1.1). This is a proof of set membership because if γ does not commit a member of ϕ then the proof fails since the prover does not obtain signatures on non-members of the set and cannot forge them. The scheme is an honest-verifier ZKP of set membership if $|\phi|$-Strong Diffie Hellman assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ [19].

A nice property of the scheme is that multiple proofs for a set Φ of commitments against the same set ϕ of values can be given efficiently by reusing verifier signatures. After obtaining signatures $(\sigma_v)_{v \in \phi}$, the prover can precompute $(\tilde{\sigma}_v)_{v \in \phi}$ in a stage 1. In stage 2, for each commitment $\gamma \in \Phi$ committing a value $v \in \phi$, the corresponding $\tilde{\sigma}_v$ can be looked up and the ZKP of knowledge can be constructed in $O(1)$ time. This results in an $O(|\phi| + |\Phi|)$ amortised complexity for proving set membership for $|\Phi|$ commitments.

4.1.2 ZKP of reverse set membership. Now we show how to extend this idea to prove reverse set membership. Note that the BB signatures used above require messages to be in group $\mathbb{Z}_q$. Since commitments are members of $\mathbb{G}_1$, one cannot use BB signatures to sign members of the set Φ of commitments for the reverse set membership proof. Recall, however, that the BBS+ signature scheme [4] lets one present a commitment $\gamma = g_1^v h_1^r$ along with $\rho_\gamma := \mathsf{NIZKPK}\{(v, r) : \gamma = g_1^v h_1^r\}$ to the signer and obtain a BBS+ signature on the value v, without leaking v to the signer. We exploit this property for the reverse set membership proof.

Our reverse set membership verifier generates fresh BBS+ signature key pairs $x \xleftarrow{\$} \mathbb{Z}_q$, $y \leftarrow f_2^x$ and sends quasi-signatures $\hat{\sigma}_\gamma := (S, c, \hat{r}) \leftarrow ((f_1 h_1^{\hat{r}} \gamma)^{\frac{1}{c+x}}, c, \hat{r})$ for each $\gamma = g_1^v h_1^r \in \Phi$, after verifying ρ_γ. If the prover knows commitment randomness r for each $\gamma \in \Phi$, it can use $\hat{\sigma}_\gamma$ to derive a valid BBS+ signature $\sigma_v \leftarrow (S, c, \mathrm{r} := \hat{r} + r) = ((f_1 g_1^v h_1^{\hat{r}+r})^{\frac{1}{c+x}}, c, \hat{r} + r)$ on the committed value v and store σ_v indexed by v. To prove that a given v is committed by some commitment in Φ, the prover looks up σ_v in O(1) time, blinds each component of σ_v to obtain a blinded signature $\tilde{\sigma}_v$ and proves knowledge of a BBS+ signature on v by revealing only $\tilde{\sigma}_v$ to the verifier. This is a proof of reverse set membership because the prover can obtain valid BBS+ signatures only on values committed by $\gamma \in \Phi$ and cannot forge it for v if no $\gamma \in \Phi$ committed v. This protocol also enjoys $O(|\phi| + |\Phi|)$ amortised complexity for multiple proofs for each $v \in \phi$ against the same set of commitments Φ. See Appendix A for the detailed protocol.

4.2 Overview of our construction

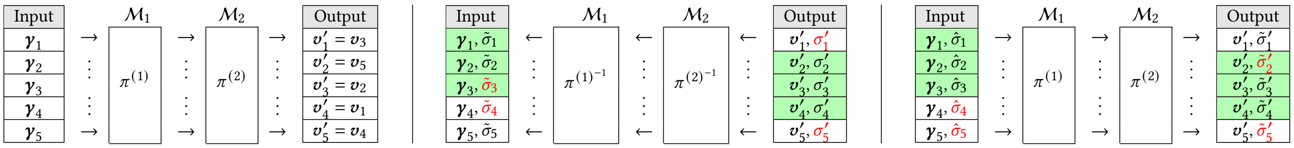
We now give an overview of our traceable mixnet construction (see Section 4.3 for detailed protocol steps). In our construction, senders send threshold encryptions of their sensitive values as input ciphertexts which get shuffled by a series of mix-servers and eventually decrypted just like a standard re-encryption mixnet [53]. However, in addition, the senders also upload Pedersen commitments to the encrypted values along with the input ciphertexts and secret-share the commitment openings among the mix-servers (see Figure 6a). The mix-servers use these shares to distributedly answer BTraceIn and BTraceOut queries via our distributed and batched ZKPs of set membership and reverse set membership (DB-SM and DB-RSM).

A DB-SM protocol for index sets I, J allows the mix-servers to prove for each uploaded commitment at an index $i \in I$ that it commits a value in the set of output plaintexts at indices J; the querier's output is the indices $I^* \subseteq I$ where the statement holds. As in the single-prover ZKP of set membership, the querier is asked to provide BB signatures on the set of plaintexts at indices J. Note, however, that in the single prover case, the prover knows the commitment's committed value v, which allows it to look-up its blinded signature $\tilde{\sigma}_v$ in $O(1)$ time. In the distributed mixnet setting, no prover (mix-server) knows the committed value, randomness or the permutation between the list of commitments and plaintexts. The challenge is to efficiently identify the blinded signature on a given commitment's committed value without letting any set of less than m mix-servers or the querier learn these secrets.

To solve this challenge, the querier signatures are encrypted and shuffled by the mix-servers *in the reverse direction* as the forward mixnet shuffle — following the inverse of the mixnet permutation – and are homomorphically blinded by random blinding factors before decryption (see Figure 6b). This process produces the blinded signatures next to the corresponding input commitments. To prove set membership, the mix-servers use the blinding factors and the commitment opening shares sent by the senders to jointly prove knowledge of a BB signature on the committed value via a DPK.

A technical complication in the above outline is that in addition to supplying BB signatures for each plaintext at an index $j \in J$, the querier must also provide invalid signatures for plaintexts at indices $j \notin J$. These invalid ("fake") signatures should also be encrypted, reverse-shuffled and blinded in the same way as the valid signatures. Without these invalid signatures, the blinded signatures would appear against exactly the input list commitments that committed a plaintext in set $\boldsymbol{v}'_J$. This would reveal even for commitments outside set $\boldsymbol{c}_I$ whether they committed a plaintext in set $\boldsymbol{v}'_J$ or not, violating our secrecy definition. With these invalid signatures, the DPKs pass only for commitments in $\boldsymbol{c}_I$ that commit a value in set $\boldsymbol{v}'_J$ but no information is revealed for commitments outside $\boldsymbol{c}_I$.

A DB-RSM protocol for index sets I, J allows the mix-servers to prove for each output plaintext at an index $j \in J$ that it is committed by a member of the set of commitments at indices I; the querier's output is the indices $J^* \subseteq J$ where the statement holds. As in the single-prover ZKP of reverse set membership, the querier is asked to provide BBS+ quasi-signatures on the set of commitments at indices I (and invalid quasi-signatures for indices $[n] \setminus I$). These quasi-signatures are encrypted, homomorphically converted to encrypted BBS+ signatures and then shuffled by the mix-servers *in the forward direction* — following the mixnet permutation — to obtain encrypted BBS+ signatures next to the corresponding plaintexts (see Figure 6c). These encrypted signatures are then homomorphically blinded



(a) The mixing process **(b) Protocol DB-SM (Figure 8)** **(c) Protocol DB-RSM (Figure 9)**

Figure 6: A summary of our construction: a) the mixing process with two mix-servers; b) a DB-SM call for index sets I, J represented by the green entries in the mixnet input and output lists: in stage 1, the querier prepares valid BB signatures σ'_j for $(v'_j)_{j \in J}$ and invalid ones (marked red) for $(v'_j)_{j \notin J}$; blinded versions $(\tilde{\sigma}_i)_{i \in [n]}$ of these signatures appear alongside commitments $(\gamma_i)_{i \in [n]}$; the DPKs of stage 2 pass only for commitments $(\gamma_i)_{i \in I}$ whose corresponding blinded signature was valid (not marked red); c) a DB-RSM call: the querier prepares valid BBS+ quasi-signatures $(\hat{\sigma}_i)_{i \in I}$ for commitments $(\gamma_i)_{i \in I}$ and invalid ones for $(\gamma_i)_{i \notin I}$; blinded versions $(\tilde{\sigma}'_j)_{j \in [n]}$ of signatures on the respective committed values appear alongside the values $(v'_j)_{j \in [n]}$; the DPKs of stage 2 pass only for values $(v'_j)_{j \in J}$ whose blinded signature was valid.

before decryption and the mix-servers use the blinding factors to provide a DPK of a BBS+ signature on the corresponding plaintext.

Even when all the mix-servers are cheating, they cannot make make the proofs pass for an incorrect entry and include, e.g., a commitment that did not commit a value in set v'_J in DB-SM output. However, this does not prevent them from deliberately failing proofs for commitments that actually committed a value in v'_J, producing a smaller-than-correct output set and violating Definition 3. Thus, we run DB-SM against both J and $[n] \setminus J$ in a BTraceIn call and make the querier abort if proofs against both the runs failed for some commitment (similarly for BTraceOut).

4.3 Technical details

Figure 7 shows our traceable mixnet construction in detail (this construction preserves secrecy only in the honest-but-curious (HBC) setting; see Section 4.3.1 for malicious security). We use the threshold $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ scheme to create ciphertexts ϵ_i, the $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ scheme to reverse-shuffle BB signatures in DB-SM and both $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ to forward-shuffle BBS+ signatures in DB-RSM. Further, we use a standard public-key encryption scheme E for securely sending shares of commitment openings to the individual mix-servers. The Keygen step creates public/private keys for all these schemes. Secret keys for $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ are shared among the mix-servers and secret keys for E for each public key is held by individual mix-servers. Via the Enc algorithm, senders upload ciphertexts ϵ_i encrypting their secret values, along with commitments γ_i and encryptions of secret shares of the commitment openings for each mix-server. They also upload proofs of knowledge ρ_{γ_i} of the commitment openings and encryptions ϵ_{r_i} of commitment randomnesses to enable the DB-RSM proofs. During the Mix protocol, the mix-servers shuffle and threshold-decrypt ϵ_i to produce permuted plaintexts $v'_j = (v_{\pi(j)})_{j \in [n]}$, where π is composed of secret permutations $\pi^{(k)}$ of each mix-server $\mathcal{M}_k$. Each $\mathcal{M}_k$ stores $\pi^{(k)}$ and decryptions of commitment opening shares to jointly answer BTraceIn/BTraceOut queries via DB-SM/DB-RSM.

Figure 8 shows our DB-SM protocol. In stage 1, the querier Q publishes valid BB signatures σ'_j on v'_j for each $j \in J$ and invalid signatures (a fixed group element) for $j \notin J$. These "signatures" are encrypted under $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and shuffled by the mix-servers in the reverse direction from $\mathcal{M}_m$ to $\mathcal{M}_1$, with each $\mathcal{M}_k$ using permutation $(\pi^{(k)})^{-1}$, to produce encrypted signatures ϵ_{σ_i} on the value v_i committed by γ_i (the encrypted signature being valid only if $v_i \in v'_J$). The mix-servers then use the multiplicative homomorphism of $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ to jointly obtain encryptions $\tilde{\epsilon}_{\sigma_i} \leftarrow \epsilon_{\sigma_i}^{\sum_{k \in [m]} b_i^{(k)}}$, where each $\mathcal{M}_k$ contributes blinding factors $b^{(k)} \xleftarrow{\$} \mathbb{Z}_q^n$. The plaintext blinded signatures $\tilde{\sigma}_i$ are finally obtained by threshold decryption of $\tilde{\epsilon}_{\sigma_i}$ and published alongside γ_i. In stage 2, $\tilde{\sigma}_i$ for each $(\gamma_i)_{i \in I}$ are looked up in $O(1)$ time. A DPK of a BB signature on the value committed by γ_i is given by proving joint knowledge of the commitment openings and blinding factors to unblind $\tilde{\sigma}_i$ to a valid signature. This DPK has the format of Section 3.1.5 and can be given efficiently since each $\mathcal{M}_k$ knows additive shares $v_i^{(k)}, r_i^{(k)}$ for the commitment openings and $b_i^{(k)}$ for the blinding factors. All indices for which the DPK passed are included in Q's output I^*. If Q is not corrupted, $(\mathcal{M}_k)_{k \in [m]}$ do not learn I^* since they do not learn which DPK passed. The amortised complexity of the entire protocol is $O(n)$.

Figure 9 shows our DB-RSM protocol. Note that to obtain BBS+ signatures on committed values, knowledge of commitment openings must be first shown. For this, Q checks the NIZKs ρ_{γ_i} uploaded by the senders for each $i \in I$. Q then sends valid BBS+ quasi-signatures $\hat{\sigma}_i$ on γ_i for $(\gamma_i)_{i \in I}$ and invalid ones for $(\gamma_i)_{i \notin I}$. It encrypts each component $(S_i, c_i, \hat{r}_i)$ of $\hat{\sigma}_i$ independently, using $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ for S_i and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ for c_i and $\hat{r}_i$, to create encrypted quasi-signatures $\epsilon_{\hat{\sigma}_i} := (\epsilon_{S_i}, \epsilon_{c_i}, \epsilon_{\hat{r}_i})$. Using $\epsilon_{\hat{r}_i}$ and the sender-uploaded encryptions ϵ_{r_i}, encrypted BBS+ signatures on the committed values are derived by homomorphically adding the commitment randomness r_i to the signature's $\hat{r}_i$ component. Thus, encrypted (valid and invalid) BBS+ signatures $\epsilon_{\sigma_i} := (\epsilon_{S_i}, \epsilon_{c_i}, \epsilon_{\mathrm{r}_i} := \epsilon_{\hat{r}_i}\epsilon_{r_i})_{i \in [n]}$ are obtained next to each commitment γ_i in the input list.

To obtain blinded BBS+ signatures on plaintext values in the permuted list v', each component $(\epsilon_S, \epsilon_c, \epsilon_{\mathrm{r}})$ of the BBS+ signature is re-encrypted individually using encryption schemes $(\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}, \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}, \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}})$ respectively and shuffled in the forward direction. The permuted and re-encrypted signatures $(\epsilon_S{}', \epsilon_c{}', \epsilon_{\mathrm{r}}{}')$ are individually blinded to obtain $(\tilde{\epsilon_S}{}', \tilde{\epsilon_c}{}', \tilde{\epsilon_{\mathrm{r}}}{}')$, which are individually threshold-decrypted to obtain blinded BBS+ signatures $\tilde{\sigma}'_j := (\tilde{S}'_j, \tilde{c}'_j, \tilde{r}'_j)$ alongside v'_j.

Note that $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ used for encrypting $\epsilon_c{}', \epsilon_{\mathrm{r}}{}'$ is homomorphic in group $\mathbb{Z}_N$, which induces addition modulo N in the plaintext space, not modulo q. Thus, blinding by blinding factors drawn from $\mathbb{Z}_q$

$\underline{\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k[[\,]])_{k\in[m]}):}$

$(\mathcal{M}_k)_{k\in[m]}$:
  $\mathsf{pk}^{(k)}, \mathsf{sk}^{(k)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$; publish $\mathsf{pk}^{(k)}$
$\mathsf{pk}_{\mathsf{EG}}, (\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}]])_{k\in[m]} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k[[\,]])_{k\in[m]})$
$\mathsf{pk}_{\mathsf{Pa}}, (\mathcal{M}_k[[\mathsf{sk}_{\mathsf{Pa}}^{(k)}]])_{k\in[m]} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k[[\,]])_{k\in[m]})$
output $\mathsf{mpk} := ((\mathsf{pk}^{(k)})_{k\in[m]}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}})$,
  $(\mathcal{M}_k[[\mathsf{msk}^{(k)} := (\mathsf{sk}^{(k)}, \mathsf{sk}_{\mathsf{EG}}^{(k)}, \mathsf{sk}_{\mathsf{Pa}}^{(k)})]])_{k\in[m]}$

$\underline{\mathsf{Enc}(\mathsf{mpk} := ((\mathsf{pk}^{(k)})_{k\in[m]}, \cdot, \mathsf{pk}_{\mathsf{Pa}}), v \in \mathbb{Z}_q):}$ // run by $S_1, \dots, S_n$

$\epsilon \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, v)$ // interpret v as an element of $\mathbb{Z}_N$
$r \xleftarrow{\$} \mathbb{Z}_q$; $\gamma \leftarrow g_1^v h_1^r$; $\rho_\gamma \leftarrow \mathsf{NIZKPK}\{((v, r)) : \gamma = g_1^v h_1^r\}$
$\epsilon_r \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, r)$ // interpret r as an element of $\mathbb{Z}_N$
$(v^{(k)})_{k\in[m]} \leftarrow \mathsf{Share}_{m,m}(v)$; $(r^{(k)})_{k\in[m]} \leftarrow \mathsf{Share}_{m,m}(r)$
$(\mathsf{ev}^{(k)})_{k\in[m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, v^{(k)}))_{k\in[m]}$
$(\mathsf{er}^{(k)})_{k\in[m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, r^{(k)}))_{k\in[m]}$
output $c := (\epsilon, \gamma, (\mathsf{ev}^{(k)}, \mathsf{er}^{(k)})_{k\in[m]}, \rho_\gamma, \epsilon_r)$

$\mathsf{Mix}(\mathsf{mpk} := (\cdot, \cdot, \mathsf{pk}_{\mathsf{Pa}}), \boldsymbol{c} := (\boldsymbol{\epsilon}_i, \cdot, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k\in[m]}, \cdot, \cdot)_{i\in[n]},$
$\underline{(\mathcal{M}_k[[\mathsf{msk}^{(k)} := (\mathsf{sk}^{(k)}, \cdot, \mathsf{sk}_{\mathsf{Pa}}^{(k)})]])_{k\in[m]}) :}$

$(\mathcal{M}_k)_{k\in[m]}$: $\pi^{(k)} \xleftarrow{\$} \mathsf{Perm}(n)$
$(\boldsymbol{\epsilon}'_j)_{j\in[n]} \leftarrow \mathsf{Shuffle}(\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}, \mathsf{pk}_{\mathsf{Pa}}, (\boldsymbol{\epsilon}_i)_{i\in[n]}, \mathcal{M}_1[[\pi^{(1)}]], \dots, \mathcal{M}_m[[\pi^{(m)}]])$
// $\boldsymbol{\epsilon}'_j = \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\pi(j)})$, where $\pi = \pi^{(m)} \circ \dots \circ \pi^{(1)}$
$(\boldsymbol{v}'_j)_{j\in[n]} \leftarrow \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}((\boldsymbol{\epsilon}'_j)_{j\in[n]}, (\mathcal{M}_k[[\mathsf{sk}_{\mathsf{Pa}}^{(k)}]])_{k\in[m]})$
$(\mathcal{M}_k)_{k\in[m]}$:
  $\boldsymbol{v}^{(k)} \leftarrow \mathbf{E}.\mathbf{Dec}(\mathsf{sk}^{(k)}, \mathbf{ev}^{(k)})$; $\boldsymbol{r}^{(k)} \leftarrow \mathbf{E}.\mathbf{Dec}(\mathsf{sk}^{(k)}, \mathbf{er}^{(k)})$
output $\boldsymbol{v}', (\mathcal{M}_k[[\omega^{(k)} := (\pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)})]])_{k\in[m]}$

$\mathsf{BTraceIn}(\mathsf{mpk} := (\cdot, \mathsf{pk}_{\mathsf{EG}}, \cdot), \boldsymbol{c} := (\cdot, \boldsymbol{\gamma}_i, \cdot, \cdot, \cdot)_{i\in[n]},$
$\boldsymbol{v}' \in \mathbb{Z}_q^n, I \subseteq [n], J \subseteq [n],$
$(\mathcal{M}_k[[\mathsf{msk}^{(k)} := (\cdot, \mathsf{sk}_{\mathsf{EG}}^{(k)}, \cdot), \omega^{(k)} := (\pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)})]])_{k\in[m]},$
$\underline{\mathcal{Q}[[\,]]) :}$

$\mathcal{Q}[[I^*]] \leftarrow \mathsf{DB\text{-}SM}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\gamma}, \boldsymbol{v}', I, J,$
  $(\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]}, \mathcal{Q}[[\,]])$ // see Fig. 8
$\mathcal{Q}[[I_c^*]] \leftarrow \mathsf{DB\text{-}SM}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\gamma}, \boldsymbol{v}', I, [n] \setminus J,$
  $(\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]}, \mathcal{Q}[[\,]])$ // see Fig. 8
$\mathcal{Q}$: **if** $I^* \cup I_c^* \neq I$: **abort**
output $\mathcal{Q}[[\boldsymbol{c}_{I^*}]]$

$\mathsf{BTraceOut}(\mathsf{mpk} := (\cdot, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}), \boldsymbol{c} := (\cdot, \boldsymbol{\gamma}_i, \cdot, \boldsymbol{\epsilon}_{\boldsymbol{r}i}, \boldsymbol{\rho}_{\boldsymbol{\gamma}i})_{i\in[n]},$
$\boldsymbol{v}' \in \mathbb{Z}_q^n, I \subseteq [n], J \subseteq [n],$
$(\mathcal{M}_k[[\mathsf{msk}^{(k)} := (\cdot, \mathsf{sk}_{\mathsf{EG}}^{(k)}, \mathsf{sk}_{\mathsf{Pa}}^{(k)}), \omega^{(k)} := (\pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)})]])_{k\in[m]},$
$\underline{\mathcal{Q}[[\,]]) :}$

$\mathcal{Q}[[J^*]] \leftarrow \mathsf{DB\text{-}RSM}(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma}, \boldsymbol{\rho}_{\boldsymbol{\gamma}}, \boldsymbol{\epsilon}_r, \boldsymbol{v}', I, J,$
  $(\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}, \mathsf{sk}_{\mathsf{Pa}}^{(k)}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]}, \mathcal{Q}[[\,]])$ // see Fig. 9
$\mathcal{Q}[[J_c^*]] \leftarrow \mathsf{DB\text{-}RSM}(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma}, \boldsymbol{\rho}_{\boldsymbol{\gamma}}, \boldsymbol{\epsilon}_r, \boldsymbol{v}', [n] \setminus I, J,$
  $(\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}, \mathsf{sk}_{\mathsf{Pa}}^{(k)}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]}, \mathcal{Q}[[\,]])$ // see Fig. 9
$\mathcal{Q}$: **if** $J^* \cup J_c^* \neq J$: **abort**
output $\mathcal{Q}[[\boldsymbol{v}'_{J^*}]]$

Figure 7: Our construction of a traceable mixnet.

Participants: Mix-servers $(\mathcal{M}_k)_{k\in[m]}$, Querier $\mathcal{Q}$
Common input: $(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\gamma} = (\boldsymbol{\gamma}_i)_{i\in[n]}, \boldsymbol{v}' = (\boldsymbol{v}'_j)_{j\in[n]}, I, J)$ **s.t.:**
  $\boldsymbol{\gamma}_i \in \mathbb{G}_1, \boldsymbol{v}'_j \in \mathbb{Z}_q, I, J \subseteq [n]$
$\mathcal{M}_k$**'s input:** $(\mathsf{sk}_{\mathsf{EG}}^{(k)}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)})$ **s.t. letting**
  $\boldsymbol{v} := \sum_{k\in[m]} \boldsymbol{v}^{(k)}, \boldsymbol{r} := \sum_{k\in[m]} \boldsymbol{r}^{(k)}, \pi := \pi^{(m)} \circ \dots \circ \pi^{(1)}$**:**
  1) $\forall i \in [n] : \boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}$
  2) $\forall j \in [n] : \boldsymbol{v}'_j := \boldsymbol{v}_{\pi(j)}$
$\mathcal{Q}$**'s output:** $I^* := \{i \in I \mid \boldsymbol{v}_i = \boldsymbol{v}'_j$ **for some** $j \in J\}$

Stage 1:
Signature generation
$\mathcal{Q}$: $x \xleftarrow{\$} \mathbb{Z}_q$; $y \leftarrow g_2^x$
  $\boldsymbol{\sigma}' \leftarrow (\boldsymbol{\sigma}'_j)_{j\in[n]}$, where $\boldsymbol{\sigma}'_j := \begin{cases} g_1^{\frac{1}{x+\boldsymbol{v}'_j}} & \text{if } j \in J \\ g_1 & \text{otherwise} \end{cases}$
  $\boldsymbol{\epsilon}'_\sigma \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\sigma}')$
  publish $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$
Shuffling
$\boldsymbol{\epsilon}_\sigma \leftarrow \mathsf{Shuffle}(\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}, \mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}'_\sigma, \mathcal{M}_m[[(\pi^{(m)})^{-1}]], \dots, \mathcal{M}_1[[(\pi^{(1)})^{-1}]])$
// $\boldsymbol{\epsilon}_{\sigma_i} = \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\sigma}_i)$; $\boldsymbol{\sigma}_i := g_1^{\frac{1}{x+\boldsymbol{v}_i}}$ if $\pi^{-1}(i) \in J$ else g_1
Homomorphic blinding
$(\mathcal{M}_k)_{k\in[m]}$:
  $\boldsymbol{b}^{(k)} \xleftarrow{\$} \mathbb{Z}_q^n$
  publish $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k)}})$ // $\tilde{\boldsymbol{\epsilon}}_{\sigma_i}^{(k)} = \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\sigma}_i^{\boldsymbol{b}_i^{(k)}})$
$(\mathcal{M}_k)_{k\in[m]}$: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \prod_{k\in[m]} \tilde{\boldsymbol{\epsilon}}_\sigma^{(k)}$ // $\tilde{\boldsymbol{\epsilon}}_{\sigma_i} = \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\sigma}_i^{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}})$
Threshold decryption
$\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, (\mathcal{M}_k[[\mathsf{sk}_{\mathsf{EG}}^{(k)}]])_{k\in[m]})$
// $\tilde{\boldsymbol{\sigma}}_i = \boldsymbol{\sigma}_i^{\boldsymbol{b}_i} = g_1^{\frac{\boldsymbol{b}_i}{x+\boldsymbol{v}_i}}$ if $\pi^{-1}(i) \in J$ else $g_1^{\boldsymbol{b}_i}$
publish $\tilde{\boldsymbol{\sigma}}$

Stage 2:
$\underline{I^* \leftarrow \emptyset}$
for $i \in I$:
  $\mathcal{Q}[[res]] \leftarrow \mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, \mathcal{M}_k[[\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)}]], \mathcal{Q}[[\,]])$
   where $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$
  $\mathcal{Q}$: **if** $res = 1$: $I^* \leftarrow I^* \cup \{i\}$
endfor
output $\mathcal{Q}[[I^*]]$

Figure 8: Protocol DB-SM **(see summary in Figure 6b).**

would not be perfectly hiding. To circumvent this issue, we follow an approach similar to [46, 47]. First, we ensure that N is much larger than q so that all additions remain integer additions and do not wrap around N (this is anyway the case since N is usually a 2048-bit modulus, q is of 254 bits and we perform a small number of homomorphic additions per ciphertext). Second, we pad blinding factors $\boldsymbol{b}_{\boldsymbol{c}j}, \boldsymbol{b}_{\boldsymbol{r}j} \in \mathbb{Z}_q$ for $\boldsymbol{\epsilon}'_{\boldsymbol{c}_j}, \boldsymbol{\epsilon}'_{\mathbf{r}_j}$ by much larger offsets $q\boldsymbol{\chi}_{\boldsymbol{c}_j}, q\boldsymbol{\chi}_{\boldsymbol{r}_j} \in \mathbb{Z}_{q(q-1)}$ so that the padded blinding factors $\boldsymbol{b}_{\boldsymbol{c}j} + q\boldsymbol{\chi}_{\boldsymbol{c}_j}, \boldsymbol{b}_{\boldsymbol{r}j} + q\boldsymbol{\chi}_{\boldsymbol{r}_j}$ are identically distributed to uniform samples from $\mathbb{Z}_{q^2}$ and provide an almost perfect integer blinding for the messages $\boldsymbol{c}_{\pi(j)} \in \mathbb{Z}_q$ and $\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} \in \mathbb{Z}_{2q}$. The offsets are removed by reducing decryptions $\tilde{\boldsymbol{c}}''_j, \tilde{\boldsymbol{r}}''_j$ of $\boldsymbol{\epsilon}'_{\boldsymbol{c}_j}, \boldsymbol{\epsilon}'_{\mathbf{r}_j}$ modulo q.

In stage 2, for each $j \in J$, a DPK is given that proves that the mix-servers know shares of $\boldsymbol{b}_{S_j}, \boldsymbol{b}_{\boldsymbol{c}_j}, \boldsymbol{b}_{\boldsymbol{r}_j}$ such that $(\tilde{S}'_j g_1^{-\boldsymbol{b}_{S_j}}, \tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}_j} \bmod q, \tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}_j} \bmod q)$ is a valid BBS+ signature on message $\boldsymbol{v}'_j$ under $\mathcal{Q}$'s verification key. The actual DPK used in Figure 9 is designed to follow the format of Section 3.1.5. Since each mix-server knows shares of $\boldsymbol{b}_{S\,j}, \boldsymbol{b}_{\boldsymbol{c}\,j}, \boldsymbol{b}_{\boldsymbol{r}\,j}, \delta_0$ and gets shares of δ_1 and δ_2 via Beaver's Mult algorithm, these DPKs can be given efficiently.

4.3.1 HBC to malicious security. Now we highlight steps to maintain secrecy even when the corrupted parties maliciously deviate from the protocol (see Appendix B for detailed steps). Any party that publishes an encryption - senders during Enc, the querier, or the mix-servers during Mix or BTraceIn/BTraceOut protocols - must provide proofs of knowledge of underlying plaintexts. This is to avoid attacks where re-encryptions of honest senders' ciphertexts are published to get them decrypted. Moreover, before participating in threshold decryption protocols, honest mix-servers must verify

Participants: Mix-servers $(\mathcal{M}_k)_{k\in[m]}$, Querier Q

Common input: $(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma} = (\boldsymbol{\gamma}_i)_{i\in[n]}, \boldsymbol{\rho_\gamma} = (\boldsymbol{\rho}_{\boldsymbol{\gamma}_i})_{i\in[n]},$
$\boldsymbol{\epsilon_r} = (\boldsymbol{\epsilon_r}_i)_{i\in[n]}, \boldsymbol{v}' = (\boldsymbol{v}'_j)_{j\in[n]}, I, J)$ **s.t.:**
$\boldsymbol{\gamma}_i \in \mathbb{G}_1, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} = \mathsf{NIZKPK}\{(v,r) : \boldsymbol{\gamma}_i = g_1^v h_1^r\}, \boldsymbol{\epsilon}_{r_i} \in \mathbb{C}(\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}})$
$\boldsymbol{v}'_j \in \mathbb{Z}_q, I, J \subseteq [n]$

$\mathcal{M}_k$**'s input:** $(\mathsf{sk}^{(k)}_{\mathsf{EG}}, \mathsf{sk}^{(k)}_{\mathsf{Pa}}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)})$ **s.t. letting**
$\boldsymbol{v} := \sum_{k\in[m]} \boldsymbol{v}^{(k)}, \boldsymbol{r} := \sum_{k\in[m]} \boldsymbol{r}^{(k)}, \pi := \pi^{(m)} \circ \cdots \circ \pi^{(1)}$**:**
1) $\forall i \in [n] : \boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}$ and $\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$
2) $\forall j \in [n] : \boldsymbol{v}'_j := \boldsymbol{v}_{\pi(j)}$

Q**'s output:** $J^* := \{j \in J \mid \boldsymbol{v}'_j = v_i$ **for some** $i \in I\}$

Stage 1:

Signature generation

Q: **for each** $i \in I$: **abort** if $\mathsf{NIZKVer}(\boldsymbol{\gamma}_i, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}) \neq 1$
$x \xleftarrow{\$} \mathbb{Z}_q; y \leftarrow f_2^x ; \boldsymbol{c} \xleftarrow{\$} \mathbb{Z}_q^n; \hat{\boldsymbol{r}} \xleftarrow{\$} \mathbb{Z}_q^n$
$S \leftarrow (S_i)_{i\in[n]}$ where $S_i := \begin{cases} (f_1 h_1^{\hat{\boldsymbol{r}}_i} \boldsymbol{\gamma}_i)^{\frac{1}{x+\boldsymbol{c}_i}} & \text{if } i \in I \\ f_1^0 & \text{otherwise} \end{cases}$
$(\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, S), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{c}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \hat{\boldsymbol{r}}))$
publish $y, \hat{\boldsymbol{\sigma}} := (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), \boldsymbol{\epsilon}_{\hat{\boldsymbol{\sigma}}} := (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$

$(\mathcal{M}_k)_{k\in[m]}$: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon_r}$

// $(\boldsymbol{\epsilon}_{S_i}, \boldsymbol{\epsilon}_{\boldsymbol{c}_i}, \boldsymbol{\epsilon}_{\mathbf{r}_i})$ encrypt $((f_1 g_1^{\boldsymbol{v}_i} h_1^{\hat{\boldsymbol{r}}_i + \boldsymbol{r}_i})^{\frac{1}{x+\boldsymbol{c}_i}}, \boldsymbol{c}_i, \hat{\boldsymbol{r}}_i + \boldsymbol{r}_i)$ if $i \in I$
// else $(f_1^0, \boldsymbol{c}_i, \hat{\boldsymbol{r}}_i + \boldsymbol{r}_i)$

Shuffling

$(\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon_c}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}') \leftarrow \mathsf{Shuffle}((\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}, \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}, \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}), (\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \mathsf{pk}_{\mathsf{Pa}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon}_{\mathbf{r}}),$
$\mathcal{M}_1[[\pi^{(1)}]], \ldots, \mathcal{M}_m[[\pi^{(m)}]])$ // shuffle all three components

Homomorphic blinding

$(\mathcal{M}_k)_{k\in[m]}$:
$(\boldsymbol{b}^{(k)}_S, \boldsymbol{b}^{(k)}_{\boldsymbol{c}}, \boldsymbol{b}^{(k)}_{\boldsymbol{r}}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n); \boldsymbol{\chi}^{(k)}_{\boldsymbol{c}}, \boldsymbol{\chi}^{(k)}_{\boldsymbol{r}} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$
publish $(\boldsymbol{\epsilon}^{(k)}_{\boldsymbol{b}S}, \boldsymbol{\epsilon}^{(k)}_{\boldsymbol{bc}}, \boldsymbol{\epsilon}^{(k)}_{\boldsymbol{br}}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}^{(k)}_S}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}^{(k)}_{\boldsymbol{c}} + q\boldsymbol{\chi}^{(k)}_{\boldsymbol{c}}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}^{(k)}_{\boldsymbol{r}} + q\boldsymbol{\chi}^{(k)}_{\boldsymbol{r}}))$

$(\mathcal{M}_k)_{k\in[m]}$: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k\in[m]} \boldsymbol{\epsilon}^{(k)}_{\boldsymbol{b}S}, \boldsymbol{\epsilon_c}{}' \prod_{k\in[m]} \boldsymbol{\epsilon}^{(k)}_{\boldsymbol{bc}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k\in[m]} \boldsymbol{\epsilon}^{(k)}_{\boldsymbol{br}})$

Threshold decryption

$\tilde{S}' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', (\mathcal{M}_k[[\mathsf{sk}^{(k)}_{\mathsf{EG}}]])_{k\in[m]})$
$\tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', (\mathcal{M}_k[[\mathsf{sk}^{(k)}_{\mathsf{Pa}}]])_{k\in[m]})$; $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}''\, \mathbf{mod}\ \boldsymbol{q}$
$\tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', (\mathcal{M}_k[[\mathsf{sk}^{(k)}_{\mathsf{Pa}}]])_{k\in[m]})$; $\tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}''\, \mathbf{mod}\ \boldsymbol{q}$
publish $\tilde{\boldsymbol{\sigma}}' := (\tilde{S}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}')$
// $\tilde{S}'_j = (f_1 g_1^{\boldsymbol{v}_{\pi(j)}} h_1^{\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)}})^{\frac{1}{x+\boldsymbol{c}_{\pi(j)}}} g_1^{\boldsymbol{b}_{Sj}}$ if $\pi(j) \in I$ else $g_1^{\boldsymbol{b}_{Sj}}$
// $\quad = (f_1 g_1^{\boldsymbol{v}'_j} h_1^{\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)}})^{\frac{1}{x+\boldsymbol{c}_{\pi(j)}}} g_1^{\boldsymbol{b}_{Sj}}$ if $\pi(j) \in I$ else $g_1^{\boldsymbol{b}_{Sj}}$
// $\tilde{\boldsymbol{c}}'_j = \boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}j}$; $\tilde{\boldsymbol{r}}'_j = \hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}j}$
// where $\boldsymbol{b}_{S_j} := \sum_{k\in[m]} \boldsymbol{b}^{(k)}_{S_j}; \boldsymbol{b}_{\boldsymbol{c}_j} := \sum_{k\in[m]} \boldsymbol{b}^{(k)}_{\boldsymbol{c}_j}; \boldsymbol{b}_{\boldsymbol{r}_j} := \sum_{k\in[m]} \boldsymbol{b}^{(k)}_{\boldsymbol{r}_j}$

Stage 2:

Q: $J^* \leftarrow \emptyset$
$(\mathcal{M}_k)_{k\in[m]}, Q$: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$
for $j \in J$:
$(\mathcal{M}_k)_{k\in[m]}$:
$\delta_0^{(k)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k)}, \delta_2^{(k)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}^{(k)}_{S_j}, \boldsymbol{b}^{(k)}_{\boldsymbol{c}_j}), \mathsf{Mult}(\delta_0^{(k)}, \boldsymbol{b}^{(k)}_{\boldsymbol{c}_j}))$
publish $\mathfrak{z}_1^{(k)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k)}} \mathfrak{h}_3^{\delta_0{}^{(k)}}$
$(\mathcal{M}_k)_{k\in[m]}, Q$: $\mathfrak{z}_1 \leftarrow \prod_{k\in[m]} \mathfrak{z}_1^{(k)}$; $\mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$
$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2)$; $\mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$
$Q[[res]] \leftarrow \mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS+}_j},$
$(\mathcal{M}_k[[\boldsymbol{b}^{(k)}_{S_j}, \boldsymbol{b}^{(k)}_{\boldsymbol{c}_j}, \boldsymbol{b}^{(k)}_{\boldsymbol{r}_j}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}]])_{k\in[m]}, Q[[]])$
where $p_{\mathsf{BBS+}_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{\boldsymbol{r}_j}} \mathfrak{h}_2^{\delta_1})$
Q: **if** $res = 1$: $J^* \leftarrow J^* \cup \{j\}$
endfor
output $Q[[J^*]]$

Figure 9: Protocol DB-RSM **(see summary in Figure 6c).**

that all encryptions are correctly created and consistently shuffled using the same permutation in DB-RSM as in Mix and the inverse permutation in DB-SM (say, using techniques of [77, 83]). Mix-servers should also verify that the querier's signatures are valid for all elements in the requested set and invalid for the complement set. The DPKs can be converted using Fiat-Shamir heuristic [42].

5 SECURITY ANALYSIS

THEOREM 1 (COMPLETENESS). *Let* Π_{TM} *be the protocol of Figure 7.* Π_{TM} *is complete (Definition 2). (Proof in Appendix C.1).*

THEOREM 2 (SOUNDNESS). *Under the DL assumption in* $\mathbb{G}_1$ *and n-SDH assumption in* $(\mathbb{G}_1, \mathbb{G}_2)$ *[14],* Π_{TM} *is sound (Definition 3).*

Proof sketch (full proof in Appendix C.2): If an adversary won $\mathsf{Exp}_{\mathsf{soundness}}$ then Q must have output either 1) a wrong $\boldsymbol{c}_{I^*}$ in a $\mathsf{BTraceIn}(I, J)$ query; or 2) a wrong $\boldsymbol{v}'_{J^*}$ in a $\mathsf{BTraceOut}(I, J)$ query. Case 1 implies that either *a)* $\boldsymbol{c}_{I^*}$ included a $\boldsymbol{c}_i \in \boldsymbol{c}_I$ not encrypting a plaintext in $\boldsymbol{v}'_J$ or *b)* it excluded a $\boldsymbol{c}_i \in \boldsymbol{c}_I$ encrypting a plaintext in $\boldsymbol{v}'_J$. Case 1a directly reduces to breaking the soundness of DB-SM for $\boldsymbol{\gamma}_i$ against set $\boldsymbol{v}'_J$. Further, since each $\boldsymbol{c}_i \in \boldsymbol{c}_I \setminus \boldsymbol{c}_{I^*}$ must be in $\boldsymbol{c}_{I^*_c}$ for Q to not abort, case 1b also reduces to breaking the soundness of DB-SM (because if $\boldsymbol{c}_i \notin \boldsymbol{c}_{I^*}$ encrypts a plaintext in $\boldsymbol{v}'_J$, it does not encrypt a plaintext in $\boldsymbol{v}'_{[n]\setminus J}$ due to distinctness of $\boldsymbol{v}'_j$s, but $\boldsymbol{c}_i \in \boldsymbol{c}_{I^*_c}$). Case 2 similarly reduces to breaking the soundness of DB-RSM.

Soundness of DB-SM *for* I, J*:* The DPK proves that $(\mathcal{M}_k)_{k\in[m]}$ know blinding factors to unblind $\tilde{\boldsymbol{\sigma}}_i$ to a valid BB signature on the value $\boldsymbol{v}_i$ committed by $\boldsymbol{\gamma}_i$ under Q's fresh public key. Since Q issued valid signatures only for plaintexts in $\boldsymbol{v}'_J$, passing the DPK if $\boldsymbol{v}_i \notin \boldsymbol{v}'_J$ requires $(\mathcal{M}_k)_{k\in[m]}$ to forge a BB signature under Q's public key, which is hard under the stated assumptions.

Soundness of DB-RSM *for* I, J*:* The DPK proves that $(\mathcal{M}_k)_{k\in[m]}$ know blinding factors to unblind $\tilde{\boldsymbol{\sigma}}'_j$ to a valid BBS+ signature on $\boldsymbol{v}'_j$ under Q's fresh public key. Since Q issued valid signatures only for commitments in $\boldsymbol{\gamma}_I$, deriving a signature on a plaintext not committed in any commitment in $\boldsymbol{\gamma}_I$ reduces to breaking the soundness of the BBS+ scheme for obtaining signatures on committed values, which is hard under the stated assumptions.

THEOREM 3 (SECRECY). *Under the IND-CPA security of* E*, the DDH assumption in* $\mathbb{G}_1$ *and the DCR assumption [73],* Π_{TM} *protects secrecy (Definition 4) against HBC adversaries in the random oracle model.*

Proof sketch (full proof in Appendix C.3): We need to show that for any pair of values v_{i_0}, v_{i_1}, no PPT adversary controlling $(\mathcal{M}_k)_{k\neq k^*}$, $(S_i)_{i\notin\{i_0,i_1\}}$ and Q can distinguish between world 0 where S_{i_0} sends v_{i_0} and S_{i_1} sends v_{i_1} and world 1 where this order is reversed, if the assert conditions in all $\mathsf{OTraceIn}$ and $\mathsf{OTraceOut}$ calls of $\mathsf{Exp}_{\mathsf{secrecy}}$ are respected (and all adversarial parties are honest-but-curious). We do this by simplifying world b for each $b \in \{0, 1\}$ by the following sequence of indistinguishability arguments:

- For each $i \in \{i_0, i_1\}$, NIZK $\boldsymbol{\rho}_{\boldsymbol{\gamma}_i}$ can be simulated, shares $\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}$ given to $(\mathcal{M}_k)_{k\neq k^*}$ can be replaced by random elements drawn from $\mathbb{Z}_q$, and encryptions $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ can be replaced by encryptions of 0 by the IND-CPA security of E.
- Threshold decryption protocols for $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ do not reveal any information beyond the decryption output, which can be

obtained by permuting the list of input values, where S_{i_0} and S_{i_1}'s input values are v_{i_b} and $v_{i_{1-b}}$ respectively. With the decryption oracles now eliminated, all $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ encryptions/re-encryptions can be replaced by encryptions of dummy values, by the IND-CPA security of $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ under the DDH and DCR assumptions respectively.

- The only information leaked during a DPK for S_{i_0}/S_{i_1}'s commitment/plaintext is whether the DPK passed or not, but the assert conditions ensure that this information is the same in both the worlds. Thus, these DPKs can be simulated by a ZK simulator that does not know which specific world it is in.
- For $i \in \{i_0, i_1\}$, $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ can be replaced by encryptions of 0 since the encrypted values $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ are not used anywhere anymore (they were used in the DPKs/encryptions earlier). Similarly, $\boldsymbol{\gamma}_{i_0}, \boldsymbol{\gamma}_{i_1}$ can now be replaced with commitments of 0.
- The blinded signatures corresponding to S_{i_0} or S_{i_1}'s commitments or plaintexts during DB-SM and DB-RSM can now be replaced with random group elements because they are not used anywhere anymore and the blinding factors chosen by $\mathcal{M}_{k^*}$ are random (note: for the c, r components of the BBS+ signature, this holds because of flooding with large blinding factors).

The two worlds obtained now are indistinguishable because now only the mixnet output list $\boldsymbol{v}'$ depends on v_{i_0} and v_{i_1} and this list is identically distributed in the two worlds because of the uniformly chosen permutation $\pi^{(k^*)}$ by $\mathcal{M}_{k^*}$.

Theorem 4 (Output secrecy). *Under the same assumptions as Theorem 3, Π_{TM} protects output secrecy (Definition 5). (Proof in Appendix C.4).*

In Appendix C.5, we also sketch a proof that after applying the additional steps mentioned in Section 4.3.1 and Appendix B, our construction protects secrecy against general malicious adversaries.

5.1 Privacy risk analysis of query outputs

Theorem 3 guarantees that a traceable mixnet does not reveal any information beyond the output of the BTraceIn/BTraceOut queries. To evaluate the privacy risk impact of these outputs themselves, we provide a mechanism in Appendix D to statically analyse information leaked by a given set of queries. Specifically, given a set Q of proposed TraceIn/TraceOut queries in an application, the mechanism outputs information potentially leaked by them: *a*) for each $i \in [n]$, the smallest set J_{min_i} representing potential plaintexts that $\boldsymbol{c}_i$ might encrypt and *b*) for each $j \in [n]$, the smallest set I_{min_j} representing potential ciphertexts that $\boldsymbol{v}'_j$ might decrypt from ($J_{\mathsf{min}_i} = [n]$ denotes that queries in Q reveal no additional information about $\boldsymbol{c}_i$; likewise for I_{min_j}). The $J_{\mathsf{min}}/I_{\mathsf{min}}$ information can then directly be used to analyse application-level security.

6 IMPLEMENTATION AND BENCHMARKS

We implemented a proof-of-concept for our traceable mixnet construction, with the primary goal of evaluating its runtime query performance. Given this focus, we mainly implemented the DB-SM and DB-RSM protocols. This allows us to directly estimate the total time for BTraceIn or BTraceOut queries as that of two DB-SM or DB-RSM invocations. However, an optimisation where the DB-SM calls for J and $[n] \setminus J$ (similarly for DB-RSM) are combined to a single call as follows considerably improves this estimate: $\mathcal{Q}$ generates signature key pairs x, y for J and x_c, y_c for $[n] \setminus J$ and sends signatures using key x_c for $j \notin J$ instead of invalid signatures, thus avoiding extraneous shuffling/decryption of invalid signatures.

We used standard threshold ElGamal encryption [34] for $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and Damgård et al.'s [29] optimised threshold Paillier encryption for $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$.[2] For key generation, we implemented a simplified protocol where a trusted dealer distributes key shares to the mix-servers. Secure distributed key generation is trivial for $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ but requires a special protocol [30] for $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$. However, our simplification is justified as key generation is a one-time setup step that can be executed ahead of time and does not affect runtime query performance.

We also implemented all the steps mentioned in Section 4.3.1 and Appendix B to evaluate performance in the realistic malicious setting. We used the following techniques to optimise these steps: 1) standard Σ-protocol techniques to let senders efficiently prove knowledge of their uploaded ciphertexts and mix-servers prove knowledge of blinding factors during homomorphic blinding; 2) batch-verification techniques of [7, 41] to let each mix-server efficiently verify the querier's signatures/quasi-signatures and the correctness of other mix-servers' decryption shares during threshold decryption; and 3) *permutation commitment*-based techniques of [77, 83] to let each mix-server efficiently prove that they created their shuffles consistently during Mix, DB-SM and DB-RSM.

We implemented all sender proofs of knowledge required to achieve our soundness and secrecy requirements. However, we did not implement proofs of well-formedness of input ciphertexts (Section 2): proofs that $\boldsymbol{\gamma}_i$ commit the same value as encrypted by $\boldsymbol{\epsilon}_i$ and that $\boldsymbol{\epsilon}_i, \boldsymbol{\epsilon}_{r_i}$ encrypt values in the range $[0, q]$.[3] This is primarily due to our focus on runtime query performance and because these costs are incurred by individual senders and mix-servers as and when senders send their data. Our proofs of knowledge incur $\sim$0.15 seconds cost per-sender for both the sender and the mix-servers. With standard Σ-protocol techniques and FO commitments [43] for proving equality of committed and encrypted values and [16, 64] for range proofs, the total time is expected to remain <1 s per sender.

Finally, we did not implement the authenticated broadcast channel. Since our uploaded datasets are moderate in size and the round complexity is small, we do not expect this to be a bottleneck.

Our implementation [1] is based on the Charm library [3] with the PBC backend [66] for pairing operations. We chose the BN254 curve [6, 56] to instantiate pairing groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, which gives a group order q of 254 bits.[4] We attempted to minimise the number of pairing operations wherever possible and used pre-computation of powers of fixed bases to speed-up exponentiation operations.

We ran all our benchmarks on an Intel(R) Xeon(R) W-1270 CPU @ 3.40GHz with 64 GB RAM on a single core. Figure 10 shows the performance of our DB-SM/DB-RSM implementation in the worst

[2] We slightly simplified [29] by skipping the factorial trick used for general t-out-of-m threshold cryptography in RSA groups, focusing on the m-out-of-m case.

[3] Senders do not need to prove distinctness of their encrypted messages because duplicate values can be detected at the mixnet output and the offending senders could be identified (by, say, a BTraceIn query) on demand.

[4] With NFS attacks [60], the security of BN254 curves has dropped to 100-110 bits [39, 67]. We are limited to BN254 because of our chosen Charm library, but we estimate that the overall performance hit in per-mix-server time on switching to a more secure curve BLS12-381 [39] is <1.3x, given that BLS12-381 operations are <2x slower than BN254 [26] and that curve operations predominantly only affect our stage 2 DPKs.

$\mathcal{M}_k$ and Q times (sec) in DB-SM and DB-RSM for different n and m						
	$m = 2$			$m = 4$		
	$n = 10^3$	$n = 10^4$	$n = 10^5$	$n = 10^3$	$n = 10^4$	$n = 10^5$
DB-SM - $\mathcal{M}_k$	35	340	3400	49	490	4916
DB-SM (HBC) - $\mathcal{M}_k$	23	225	2268	23	227	2274
DB-SM - Q	20	190	2000	20	200	2000
collab-zkSNARK-SM	4120	42960	446800	4120	42960	446800
DB-RSM - $\mathcal{M}_k$	168	1600	20000	240	2400	32000
DB-RSM (HBC) - $\mathcal{M}_k$	122	1188	11981	129	1291	12949
DB-RSM - Q	62	610	6200	62	620	6200
collab-zkSNARK-RSM	4160	43100	448920	4160	43100	448920

Detailed breakdown for $n = 10^4$ and $m = 4$	
- Size of n input ciphertexts	200 MB
- ($\mathcal{M}_k$) Mixing (Fig. 7):	343 s
DB-SM (Fig. 8):	
- (Q) Generating n BB signatures/encryptions	8.3 s
- ($\mathcal{M}_k$) Verifying n BB signatures/encryptions	15 s
- ($\mathcal{M}_k$) Re-encryption of encrypted signatures	7.3 s
- ($\mathcal{M}_k$) Proof-of-shuffle of encrypted signatures	139 s
- ($\mathcal{M}_k$) Homomorphic blinding of encrypted signatures	130 s
- ($\mathcal{M}_k$) Threshold decryption of encrypted signatures	22 s
- ($\mathcal{M}_k$) Generating n DPK proofs for p_{BB}	170 s
- (Q) Verifying n DPK proofs for p_{BB}	190 s
- Size of n BB signatures	0.3 MB
- Size of n DPK proofs for p_{BB}	3.8 MB
DB-RSM (Fig. 9):	
- (Q) Verifying n PoKs of commitments	7.9 s
- (Q) Generating n BBS+ quasi-signatures	23 s
- ($\mathcal{M}_k$) Verifying n BBS+ quasi-signatures/encryptions	27 s
- ($\mathcal{M}_k$) Re-encryption of encrypted signatures	16 s
- ($\mathcal{M}_k$) Proof-of-shuffle of encrypted signatures	349 s
- ($\mathcal{M}_k$) Homomorphic blinding of encrypted signatures	880 s
- ($\mathcal{M}_k$) Threshold decryption of encrypted signatures	441 s
- ($\mathcal{M}_k$) Generating n DPK proofs for $p_{\mathrm{BBS+}}$	673 s
- (Q) Verifying n DPK proofs for $p_{\mathrm{BBS+}}$	590 s
- Size of n BBS+ signatures	0.9 MB
- Size of n DPK proofs for $p_{\mathrm{BBS+}}$	11 MB

Figure 10: Performance of DB-SM and DB-RSM (n: number of input ciphertexts, m: number of mix-servers). $\mathcal{M}_k$ and Q denote per-mix-server and querier times respectively; collab-zkSNARK denotes estimated per-prover times in collaborative zkSNARKs [72].

case when $I = J = [n]$, with the overall mix-server and querier times for different n and m at the top and the detailed breakdown for $n = 10000$ ciphertexts and $m = 4$ mix-servers at the bottom. All reported values are averages over 3 runs. The deviation from the average in any run is <1.5%. We report per-mix-server times, which accurately capture real-world latencies as the heavy operations like proofs-of-shuffle, homomorphic blinding, threshold decryption and stage 2 DPKs can be run in parallel by the mix-servers. The only sequential operation is re-encryption, but it is a negligible fraction of other steps. We also report timings for the HBC case to highlight the overhead introduced by the malicious security steps.

Our ZKPs are practical for offline batch processing tasks, finishing within an hour and requiring moderate amount of data to be published for 10000 ciphertexts. They scale linearly with n. The scale-up with m is constant for the HBC mix-server time and for the querier, but in the malicious case, each mix-server needs to verify other mix-servers' output, which leads to a ∼1.4x increase from $m = 2$ to $m = 4$. Our main bottlenecks are expensive pairing and exponentiation computations in our DPKs and expensive Paillier operations in DB-RSM. The additional proofs for malicious security add an overhead of roughly 1.5-2x over the HBC case where these steps are skipped.

There exists a high degree of task parallelism in our construction, since DPKs in stage 2 are independent of each other and stage 1 operations incur at most constant communication overhead. Thus, we expect significant speedups if each mix-server and the querier are given multiple cores. With a moderate parallel cluster of 100 nodes, recovery for an election with 10^6 votes can thus be performed within a few hours. Further, if the signer in stage 1 of our ZKPs could be a separate trusted entity different than the querier and pre-sign all set entries, then our ZKPs become completely non-interactive and only incur stage 2 costs for verification.

Comparison. The only technique comparable to our distributed setting is collaborative zkSNARKs [72]. We indirectly estimate our performance against them by employing a thumbrule given by [72] that the per-prover time in a collaborative zkSNARK, assuming each prover already has a share of the SNARK witness, is ∼2x the prover time in the corresponding single-prover zkSNARK. Thus, we implemented zkSNARKs for $\rho_{\text{SM-Acc}}$ and $\rho_{\text{RSM-Acc}}$ via Merkle accumulators (see Section 1.2.4). We used the ZoKrates toolchain [38] and the Groth16 proof system [50]. We used the Baby Jubjub curve [81], which has similar order as BN254 and allows efficient computation of Merkle hashes for commitments for $\rho_{\text{RSM-Acc}}$. For creating Merkle hashes, we used the Poseidon hash function [49].

Figure 10 (top) also shows the per-prover times in a collaborative zkSNARK approach estimated as above (averaged over 3 runs with deviation <1.1%). We find the prover time for one $\rho_{\text{SM-Acc}}$ or $\rho_{\text{RSM-Acc}}$ proof against a set of size 10000 is ∼2.15 s, which when scaled to 10000 commitments takes ∼21500 s. From this, the per-prover time in collaborative zkSNARKs is estimated to be ∼43000 s, as shown in the column for $n = 10^4$ (for any m). This estimate is conservative as it does not count the time taken to securely distribute shares of the SNARK witness among the collaborating provers. This makes our DB-SM and DB-RSM proofs ∼86x and ∼18x faster than collaborative zkSNARKs. We note that our verification times (200 s and 620 s for $n = 10000$) are slower than zkSNARKs' ∼50 s, but the dominant prover times in zkSNARKs imply that our techniques still bring drastic overall improvements.

We also ran Benarroch et al.'s [12] official implementation [52], which proves $\rho_{\text{SM-Acc}}$ for a single prover (but not $\rho_{\text{RSM-Acc}}$). This takes ∼2200 s for 10000 $\rho_{\text{SM-Acc}}$ proofs, excluding the $O(n^2)$ time taken to generate the RSA accumulator witnesses (see Section 1.2.4).

7 CONCLUSION

We introduced and formalised the notion of traceable mixnets, extending traditional mixnets to provably answer useful subset queries in zero knowledge. We also proposed a traceable mixnet construction using novel distributed ZKPs of set membership and reverse set membership, which are useful in other settings too. We implemented these ZKPs and showed that they are significantly faster than the state-of-the-art techniques. Nevertheless, our current implementation is practical only for offline batch processing such as recovery in elections. Constructing traceable mixnets for real-time privacy applications is a challenging open problem.

ACKNOWLEDGMENTS

We wish to thank Rohit Vaish, Kabir Tomer and Mahesh Sreekumar Rajasree for their helpful discussions and comments, and Aarav Varshney for assistance in setting up the benchmarks. Our gratitude also extends to the anonymous reviewers whose suggestions significantly improved the manuscript's presentation.

This research was partially supported by the Pankaj Gupta Chair in Privacy and Decentralisation. Prashant Agrawal received additional support from the Pankaj Jalote Doctoral Grant, and Abhinav Nakarmi was supported by a research grant from the MPhasis F1 Foundation.

REFERENCES

[1] Prashant Agrawal, Abhinav Nakarmi, Mahabir Prasad Jhanwar, Subodh Sharma, and Subhashis Banerjee. 2023. Benchmarks for traceable mixnets. https://github.com/agrawalprash/traceable-mixnets/tree/v1.

[2] Prashant Agrawal, Kabir Tomer, Abhinav Nakarmi, Mahabir Prasad Jhanwar, Subodh Vishnu Sharma, and Subhashis Banerjee. 2023. OpenVoting: recoverability from failures in dual voting. In *E-VOTE-ID*. 18–34.

[3] Joseph A. Akinyele, Christina Garman, Ian Miers, Matthew W. Pagano, Michael Rushanan, Matthew Green, and Aviel D. Rubin. 2013. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering* 3, 2 (2013), 111–128.

[4] Man Ho Au, Willy Susilo, and Yi Mu. 2006. Constant-size dynamic k-TAA. In *Security and Cryptography for Networks*. 111–125.

[5] Michael Backes, Jeremy Clark, Aniket Kate, Milivoj Simeonovski, and Peter Druschel. 2014. BackRef: accountability in anonymous communication networks. In *ACNS*. 380–400.

[6] Paulo S. L. M. Barreto and Michael Naehrig. 2006. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography*. 319–331.

[7] Mihir Bellare, Juan A Garay, and Tal Rabin. 1998. Fast batch verification for modular exponentiation and digital signatures. In *EUROCRYPT*. 236–250.

[8] Josh Benaloh. 2008. Administrative and public verifiability: can we have both? *EVT* 8 (2008), 1–10.

[9] Josh Benaloh and Michael de Mare. 1993. One-way accumulators: a decentralized alternative to digital signatures. In *EUROCRYPT*. 274–285.

[10] Josh Benaloh and Dwight Tuinstra. 1994. Receipt-free secret-ballot elections. In *ACM symposium on Theory of Computing*. 544–553.

[11] Josh C Benaloh and Moti Yung. 1986. Distributing the power of a government to enhance the privacy of voters. In *ACM symposium on Principles of Distributed Computing*. 52–62.

[12] Daniel Benarroch, Matteo Campanelli, Dario Fiore, Kobi Gurkan, and Dimitris Kolonelos. 2021. Zero-knowledge proofs for set membership: efficient, succinct, modular. In *Intl. Conf. Financial Cryptography and Data Security*. 393–414.

[13] David Bernhard, Véronique Cortier, David Galindo, Olivier Pereira, and Bogdan Warinschi. 2015. SoK: a comprehensive analysis of game-based ballot privacy definitions. In *IEEE S&P*. 499–516.

[14] Dan Boneh and Xavier Boyen. 2004. Short signatures without random oracles. In *EUROCRYPT*. 56–73.

[15] Dan Boneh, Benedikt Bünz, and Ben Fisch. 2019. Batching techniques for accumulators with applications to IOPs and stateless blockchains. In *CRYPTO*. 561–586.

[16] Fabrice Boudot. 2000. Efficient proofs that a committed number lies in an interval. In *International Conference on the Theory and Applications of Cryptographic Techniques*. 431–444.

[17] Xavier Boyen, Thomas Haines, and Johannes Müller. 2020. A verifiable and practical lattice-based decryption mix net with external auditing. In *European Symposium on Research in Computer Security*. 336–356.

[18] Jan Camenisch. 1998. *Group signature schemes and payment systems based on the discrete logarithm problem.* Ph. D. Dissertation. ETH Zurich.

[19] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. 2008. Efficient protocols for set membership and range proofs. In *ASIACRYPT*.

[20] Jan Camenisch and Anna Lysyanskaya. 2004. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO*. 56–72.

[21] Matteo Campanelli, Dario Fiore, Semin Han, Jihye Kim, Dimitris Kolonelos, and Hyunok Oh. 2022. Succinct zero-knowledge batch proofs for set accumulators. In *CCS*. 455–469.

[22] David Chaum and Eugène van Heyst. 1991. Group signatures. In *EUROCRYPT*. 257–265.

[23] David L. Chaum. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* 24, 2 (1981), 84–90.

[24] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. 2020. Marlin: preprocessing zkSNARKs with universal and updatable SRS. In *EUROCRYPT*. 738–768.

[25] Joris Claessens, Claudia Diaz, Caroline Goemans, Jos Dumortier, Bart Preneel, and Joos Vandewalle. 2003. Revocable anonymous access to the Internet? *Internet Research* 13, 4 (2003), 242–258.

[26] Consensys. 2021. Benchmarking pairing-friendly elliptic curves libraries. https://hackmd.io/@gnark/eccbench. Accessed on November 27, 2023.

[27] Ronald Cramer, Ivan Damgård, and Ueli Maurer. 2000. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT*. 316–334.

[28] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. 1994. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*. 174–187.

[29] Ivan Damgård, Mads Jurik, and Jesper Buus Nielsen. 2010. A generalization of Paillier's public-key system with applications to electronic voting. *Intl. Jr. Inf. Security* 9 (2010), 371–385.

[30] Ivan Damgård and Maciej Koprowski. 2001. Practical threshold RSA signatures without a trusted dealer. In *EUROCRYPT*. 152–165.

[31] Anupam Datta, Divya Sharma, and Arunesh Sinha. 2012. Provable de-anonymization of large datasets with sparse dimensions. In *Principles of Security and Trust*. Springer, 229–248.

[32] Pankaj Dayama, Arpita Patra, Protik Paul, Nitin Singh, and Dhinakaran Vinayagamurthy. 2022. How to prove any NP statement jointly? Efficient distributed-prover zero-knowledge protocols. *PETS* 2 (2022), 517–556.

[33] Stéphanie Delaune, Steve Kremer, and Mark Ryan. 2009. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security* 17, 4 (2009), 435–487.

[34] Yvo Desmedt. 1994. Threshold cryptography. *European Tr. Telecommunications* 5, 4 (1994), 449–458.

[35] Yvo Desmedt and Yair Frankel. 1989. Threshold cryptosystems. In *CRYPTO*, Vol. 435. 307–315.

[36] Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. 2004. Tor: The second-generation onion router.. In *USENIX Security*, Vol. 4. 303–320.

[37] Cynthia Dwork. 2006. Differential privacy. In *Automata, Languages and Programming*. Springer, 1–12.

[38] Jacob Eberhardt and Stefan Tai. 2018. ZoKrates - scalable privacy-preserving off-chain computations. In *IEEE iThings*. 1084–1091.

[39] Ben Edgington. 2023. BLS12-381 for the rest of us. https://hackmd.io/@benjaminion/bls12-381. Accessed on November 27, 2023.

[40] Taher ElGamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Tr. Information theory* 31, 4 (1985), 469–472.

[41] Anna Lisa Ferrara, Matthew Green, Susan Hohenberger, and Michael Østergaard Pedersen. 2009. Practical short signature batch verification. In *CT-RSA*. 309–324.

[42] Amos Fiat and Adi Shamir. 1986. How to prove yourself: practical solutions to identification and signature problems. In *CRYPTO*. 186–194.

[43] Eiichiro Fujisaki and Tatsuaki Okamoto. 1997. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*. 16–30.

[44] Jun Furukawa. 2005. Efficient and verifiable shuffling and shuffle-decryption. *IEICE Tr. Fundamentals of Electronics, Communications and Computer Sciences* 88, 1 (2005), 172–188.

[45] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. 2019. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. *Cryptology ePrint Archive* (2019).

[46] Rosario Gennaro and Steven Goldfeder. 2018. Fast multiparty threshold ECDSA with fast trustless setup. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 1179–1194.

[47] Rosario Gennaro, Steven Goldfeder, and Arvind Narayanan. 2016. Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security. In *ACNS*. 156–174.

[48] Shafi Goldwasser and Silvio Micali. 1982. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*. 365–377.

[49] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. 2021. Poseidon: a new hash function for zero-knowledge proof systems. In *USENIX Security*.

[50] Jens Groth. 2016. On the size of pairing-based non-interactive arguments. In *EUROCRYPT*. 305–326.

[51] Jens Groth and Markulf Kohlweiss. 2015. One-out-of-many proofs: or how to leak a secret and spend a coin. In *EUROCRYPT*. 253–280.

[52] Kobi Gurkan. 2021. CPSNARKs-Set. https://github.com/kobigurk/cpsnarks-set. [Accessed Mar 21, 2023].

[53] Thomas Haines and Johannes Müller. 2020. SoK: techniques for verifiable mix nets. In *CSF*. 49–64.

[54] Rawane Issa, Nicolas Alhaddad, and Mayank Varia. 2022. Hecate: abuse reporting in secure messengers with sealed sender. In *USENIX Security*. 2335–2352.

[55] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. 2002. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX Security*. 339–353.

[56] John Johnson. 2021. BN254 for the rest of us. https://hackmd.io/@jpw/bn254. Accessed on April 14, 2023.

[57] Marcel Keller, Gert Læssøe Mikkelsen, and Andy Rupp. 2012. Efficient threshold zero-knowledge with applications to user-centric protocols. In *Intl. Conf. Information Theoretic Security*. 147–166.

[58] Shahram Khazaei, Tal Moran, and Douglas Wikström. 2012. A mix-net from any CCA2 secure cryptosystem. In *Intl. Conf. Theory and Application of Cryptology and Information Security*. 607–625.

[59] Shahram Khazaei and Douglas Wikström. 2013. Randomized partial checking revisited. In *CT-RSA*. 115–128.

[60] Taechan Kim and Razvan Barbulescu. 2016. Extended tower number field sieve: A new complexity for the medium prime case. In *Annual international cryptology conference*. Springer, 543–571.

[61] Ralf Küsters, Johannes Müller, Enrico Scapin, and Tomasz Truderung. 2016. sElect: A lightweight verifiable remote voting system. In *CSF*. 341–354.

[62] Ralf Küsters and Tomasz Truderung. 2016. Security analysis of re-encryption RPC mix nets. In *EuroS&P*. 227–242.

[63] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2014. Formal analysis of Chaumian mix nets with randomized partial checking. In *IEEE S&P*. 343–358.

[64] Yehuda Lindell. 2017. Fast secure two-party ECDSA signing. In *CRYPTO*. 613–644.

[65] David Lundin and Peter Y. A. Ryan. 2008. Human readable paper verification of Prêt à Voter. In *ESORICS*. 379–395.

[66] Ben Lynn. 2013. PBC library (pbc-0.5.14). https://crypto.stanford.edu/pbc/. [Accessed June 10, 2019].

[67] Alfred Menezes, Palash Sarkar, and Shashank Singh. 2016. Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography. In *International Conference on Cryptology in Malaysia*. 83–108.

[68] Ralph C Merkle. 1987. A certified digital signature. *CRYPTO* (1987), 218–238.

[69] Arvind Narayanan and Vitaly Shmatikov. 2008. Robust de-anonymization of large sparse datasets. In *IEEE S&P*. 111–125.

[70] C Andrew Neff. 2001. A verifiable secret shuffle and its application to e-voting. In *CCS*. 116–125.

[71] Lan Nguyen. 2005. Accumulators from bilinear pairings and applications. In *CT-RSA*. 275–292.

[72] Alex Ozdemir and Dan Boneh. 2022. Experimenting with collaborative zk-SNARKs: zero-knowledge proofs for distributed secrets. In *USENIX Security*. 4291–4308.

[73] Pascal Paillier. 1999. Public-key cryptosystems based on composite degree residuosity classes. In *Intl. Conf. Theory and Applications of Cryptographic Techniques*. 223–238.

[74] Torben P. Pedersen. 1991. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*. 129–140.

[75] Ronald L Rivest, Adi Shamir, and Yael Tauman. 2001. How to leak a secret. In *ASIACRYPT*. 552–565.

[76] Bruce Schneier. 2007. *Applied cryptography: protocols, algorithms, and source code in C*. John Wiley & Sons.

[77] Björn Terelius and Douglas Wikström. 2010. Proofs of restricted shuffles. In *Intl. Conf. Cryptology in Africa*. 100–113.

[78] Nirvan Tyagi, Paul Grubbs, Julia Len, Ian Miers, and Thomas Ristenpart. 2019. Asymmetric message franking: content moderation for metadata-private end-to-end encryption. In *CRYPTO*. 222–250.

[79] Nirvan Tyagi, Ian Miers, and Thomas Ristenpart. 2019. Traceback for end-to-end encrypted messaging. In *CCS*. 413–430.

[80] Luis Von Ahn, Andrew Bortz, Nicholas J Hopper, and Kevin O'Neill. 2006. Selectively traceable anonymity. In *PETS*. 208–222.

[81] Barry WhiteHat, Jordi Baylina, and Marta Bellés. 2020. Baby Jubjub elliptic curve. *Ethereum Improvement Proposal, EIP-2494* 29 (2020).

[82] Douglas Wikström. 2005. A sender verifiable mix-net and a new proof of a shuffle. In *Intl. Conf. Theory and Application of Cryptology and Information Security*. 273–292.

[83] Douglas Wikström. 2009. A commitment-consistent proof of a shuffle. In *Australasian Conf. Information Security and Privacy*. 407–421.

[84] Bin Yang, Issei Sato, and Hiroshi Nakagawa. 2015. Bayesian differential privacy on correlated data. In *SIGMOD*. 747–762.

A SINGLE PROVER REVERSE SET MEMBERSHIP

Figure 11 shows the ZKP of reverse set membership in the single prover case. Note that in this protocol, the prover knows commitment openings for each $\gamma \in \Phi$ (not only for the commitment committing v).

Participants: Prover $\mathcal{P}$, Verifier $\mathcal{V}$
Common input: $\Phi \in 2^{\mathbb{G}_1}$(set of commitments), $v \in \mathbb{Z}_q$
$\mathcal{P}$'s input: For each $\gamma_v \in \Phi$ (Φ indexed by v): (v, r_v) s.t. $\gamma_v = g_1^v h_1^{r_v}$
$\mathcal{V}$'s output: 1 if $\mathcal{V}$ accepts else 0

Stage 1:
$\mathcal{P}$: **for each** $\gamma \in \Phi$: $\rho_\gamma \leftarrow \mathsf{NIZKPK}\{(v, r) : \gamma = g_1^v h_1^r\}$
send $\{\rho_\gamma \mid \gamma \in \Phi\}$ to $\mathcal{V}$
$\mathcal{V}$:
for each $\gamma \in \Phi$:
abort if $\mathsf{NIZKVer}(\gamma, \rho_\gamma) \neq 1$
$x \xleftarrow{\$} \mathbb{Z}_q; y \leftarrow f_2^x$
$c_\gamma \xleftarrow{\$} \mathbb{Z}_q; \hat{r}_\gamma \xleftarrow{\$} \mathbb{Z}_q; S_\gamma \leftarrow (f_1 h_1^{\hat{r}_\gamma} \gamma)^{\frac{1}{x+c_\gamma}}$
endfor
send $y, \hat{\sigma}_\gamma := (S_\gamma, c_\gamma, \hat{r}_\gamma)$ to $\mathcal{P}$
$\mathcal{P}$:
Find $\gamma_v \in \Phi$ s.t. $\gamma_v = g_1^v h_1^{r_v}$ // $O(1)$ look-up if Φ is indexed by v
$\hat{\sigma}_v := (S_v, c_v, \hat{r}_v) := \hat{\sigma}_{\gamma_v}$
$(b_S, b_c, b_r) \xleftarrow{\$} (\mathbb{Z}_q \times \mathbb{Z}_q \times \mathbb{Z}_q)$
$(\tilde{S}_v, \tilde{c}_v, \tilde{r}_v) \leftarrow (S_v g_1^{b_S}, c_v + b_c \bmod q, \hat{r}_v + r_v + b_r \bmod q)$
send $\tilde{\sigma}_v := (\tilde{S}_v, \tilde{c}_v, \tilde{r}_v)$ to $\mathcal{V}$

Stage 2:
// $\mathcal{P}$ proves (in ZK) knowledge of a BBS+ signature
// $(\tilde{S}_v g_1^{-b_S}, \tilde{c}_v - b_c \bmod q, \tilde{r}_v - b_r \bmod q)$ on v, i.e.,
// $e(\tilde{S}_v g_1^{-b_S}, y f_2^{\tilde{c}_v - b_c}) = e(f_1 g_1^v h_1^{\tilde{r}_v - b_r}, f_2)$ (Sec. 3.1.2):
$\mathcal{P}, \mathcal{V}$:
$\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$
$\mathfrak{z}_2 \leftarrow e(\tilde{S}_v, y f_2^{\tilde{c}_v}) / e(f_1 g_1^v h_1^{\tilde{r}_v}, f_2)$
$\mathfrak{g}_1 \leftarrow e(\tilde{S}_v, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}_v})$
$\mathcal{P}$:
$\delta_0 \xleftarrow{\$} \mathbb{Z}_q; \delta_1 \leftarrow b_S b_c; \delta_2 \leftarrow \delta_0 b_c$
send $\mathfrak{z}_1 := \mathfrak{h}_2^{b_S} \mathfrak{h}_3^{\delta_0}$ to $\mathcal{V}$
$\mathcal{P}, \mathcal{V}$:
$res \leftarrow \mathsf{PK}\{(b_S, b_c, b_r, \delta_0, \delta_1, \delta_2) : (\mathfrak{z}_1 = \mathfrak{h}_2^{b_S} \mathfrak{h}_3^{\delta_0}) \wedge$
$(1_{\mathbb{G}_T} = \mathfrak{z}_1^{-b_c} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2}) \wedge (\mathfrak{z}_2 = \mathfrak{g}_1^{b_c} \mathfrak{g}_2^{b_S} \mathfrak{h}_1^{b_r} \mathfrak{h}_2^{\delta_1})\}$
$\mathcal{V}$:
output res

Figure 11: ZKP of reverse set membership $\rho_{\mathsf{RSM}}(\Phi, v) := \mathsf{PK}\{(r) : \gamma = g_1^v h_1^r \wedge \gamma \in \Phi\}$, if the prover knows openings for each $\gamma \in \Phi$.

B FROM HONEST-BUT-CURIOUS TO MALICIOUS MODEL

In this section, we mention steps required to derive secrecy (Definition 4) in the general case when the adversary allows the corrupted parties (all-but-two senders, all-but-one mix-servers and the querier) to deviate from the protocol:

- Each sender S_i must attach NIZK proofs of knowledge $(\boldsymbol{\rho}_{\epsilon_i}, \boldsymbol{\rho}_{\epsilon_{r_i}}, (\boldsymbol{\rho}_{\mathrm{ev}_i}^{(k)}, \boldsymbol{\rho}_{\mathrm{er}_i}^{(k)})_{k \in [m]})$ of plaintexts encrypted by encryptions $(\boldsymbol{\epsilon}_i, \boldsymbol{\epsilon}_{r_i}, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]})$ sent by it. Each $(\mathcal{M}_k)_{k \in [m]}$ must verify these proofs before processing anything.
- For encryptions $\boldsymbol{\epsilon}'_{\boldsymbol{\sigma}}$ in DB-SM and $\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}}$ in DB-RSM, the querier must publish their randomnesses and each $(\mathcal{M}_k)_{k \in [m]}$ must verify that they were created correctly. Each $(\mathcal{M}_k)_{k \in [m]}$ must also verify that $\boldsymbol{Q}$ gave valid signatures/quasi-signatures for each element in the requested set and invalid ones for its complement. Note that this also involves verifying that $\boldsymbol{c}, \hat{\boldsymbol{r}}$ in DB-RSM contain only elements in the range $[0, q]$.

- Each $(\mathcal{M}_k)_{k\in[m]}$ must provide proofs of correct shuffle in all the shuffle protocols. In a proof of shuffle, $\mathcal{M}_k$ for a given input ciphertext list $\boldsymbol{\epsilon}$ and output ciphertext list $\boldsymbol{\epsilon}'$ proves that $\boldsymbol{\epsilon}'$ is a permutation and re-encryption of $\boldsymbol{\epsilon}$ under a permutation that is consistent across Mix, DB-SM and DB-RSM protocols (i.e., the permutation used during DB-RSM is the same as that used during Mix and the permutation used during DB-SM is the inverse of it). Such proofs can be given efficiently using the permutation-commitment based techniques of [77, 83]. Each $(\mathcal{M}_k)_{k\in[m]}$ must verify proofs given by other mix-servers before participating in the corresponding threshold decryption protocols.
- Each $(\mathcal{M}_k)_{k\in[m]}$ must provide proofs of knowledge of the blinding factors of homomorphically blinded signatures. During DB-SM, this involves giving proofs $\boldsymbol{\rho}^{(k)}_{\tilde{\sigma}_i}$ for each $\tilde{\boldsymbol{\epsilon}}^{(k)}_{\sigma_i}$ published by $\mathcal{M}_k$ as a blinding of $\boldsymbol{\epsilon}_{\sigma_i}$ with blinding factor $\boldsymbol{b}_i^{(k)}$. Note that $\boldsymbol{\rho}^{(k)}_{\tilde{\sigma}_i} :=$ NIZKPK$\{(r, \boldsymbol{b}_i^{(k)}) : \tilde{c}_0 = g_1^r c_0^{\boldsymbol{b}_i^{(k)}} \wedge \tilde{c}_1 = \mathsf{pk}_{\mathsf{EG}}^r c_1^{\boldsymbol{b}_i^{(k)}}\}$, where $\boldsymbol{\epsilon}_{\boldsymbol{\sigma} i}$ and $\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\sigma} i}^{(k)}$ are parsed as ElGamal ciphertexts (c_0, c_1) and $(\tilde{c}_0, \tilde{c}_1)$ respectively. During DB-RSM, this involves giving NIZK proofs $(\boldsymbol{\rho}^{(k)}_{bS_j}, \boldsymbol{\rho}^{(k)}_{bc_j}, \boldsymbol{\rho}^{(k)}_{br_j})$ for each $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}{}_j^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}{}_j^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}{}_j^{(k)})$, which are encryptions of $g_1^{\boldsymbol{bs}_j^{(k)}}$ under $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and of $\boldsymbol{b}'_{\boldsymbol{c}}{}_j^{(k)} := \boldsymbol{b}_{\boldsymbol{c}}{}_j^{(k)} + q\boldsymbol{\chi}_{\boldsymbol{c}}{}_j^{(k)}$ and $\boldsymbol{b}'_{\boldsymbol{r}}{}_j^{(k)} := \boldsymbol{b}_{\boldsymbol{r}}{}_j^{(k)} + q\boldsymbol{\chi}_{\boldsymbol{r}}{}_j^{(k)}$ under $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ respectively. Here, $\boldsymbol{\rho}^{(k)}_{bS_j} :=$ NIZKPK$\{(r, \boldsymbol{bs}_j^{(k)}) : \tilde{c}_0 = g_1^r \wedge \tilde{c}_1 = g_1^{\boldsymbol{bs}_j^{(k)}} \mathsf{pk}_{\mathsf{EG}}^r\}$, where $\boldsymbol{\epsilon}_{\boldsymbol{bS}}{}_j^{(k)}$ is parsed as the ElGamal ciphertext $(\tilde{c}_0, \tilde{c}_1)$ and $\boldsymbol{\rho}^{(k)}_{bc_j}, \boldsymbol{\rho}^{(k)}_{br_j}$ are proofs of knowledge of the plaintexts encrypted by $\boldsymbol{\epsilon}_{\boldsymbol{bc}}{}_j^{(k)}$, $\boldsymbol{\epsilon}_{\boldsymbol{br}}{}_j^{(k)}$ respectively.
- During the $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$.TDec and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$.TDec protocols, each $(\mathcal{M}_k)_{k\in[m]}$ must provide proofs that they produced correct decryption shares. Each $(\mathcal{M}_k)_{k\in[m]}$ should proceed with stage 2 only if these proofs pass.
- During the DB-RSM protocol, each $(\mathcal{M}_k)_{k\in[m]}$ must provide a proof of knowledge of the opening of $\mathfrak{z}_1{}^{(k^*)}$. Each $(\mathcal{M}_k)_{k\in[m]}$ should proceed only if the proof passes.

Note that the DPKs in DB-SM and DB-RSM already employ the Fiat-Shamir heuristic [42] which makes them general ZKPs in the random oracle model (see Section 3.1.5).

In Section C.5, we sketch a proof that our construction with the above steps protects secrecy (Definition 4) against general malicious adversaries.

C PROOFS

C.1 Proof for Theorem 1

It can be inspected that when all the parties are honest, inputs to protocols DB-SM and DB-RSM satisfy the preconditions mentioned in Figures 8 and 9, respectively. This is also true for the reruns of DB-SM and DB-RSM in the BTraceIn/BTraceOut calls against the complement sets. Thus, by Lemma 1, I^* and I^*_c obtained by $\mathcal{Q}$ in a BTraceIn call satisfy $I^* = \{i \in I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_J\}$ and $I^*_c = \{i \in I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_{[n]\setminus J}\}$. Also, the correctness of Mix implies that for each $i \in I \subseteq [n]$, $\boldsymbol{v}_i \in \{\boldsymbol{v}'_j \mid j \in [n]\}$, i.e., $\boldsymbol{v}_i \in \boldsymbol{v}'_J \cup \boldsymbol{v}'_{[n]\setminus J}$ for any $J \subseteq [n]$. Thus, $I^* \cup I^*_c = I$, which implies that $\mathcal{Q}$ does not abort and outputs a $\boldsymbol{c}_{I^*}$ that satisfies the first condition of $\mathsf{Exp}_{\mathsf{completeness}}$ (Figure 3). By a similar argument using Lemma 2, it follows that $\mathcal{Q}$ does not abort in a BTraceOut call and outputs a $\boldsymbol{v}'_{J^*}$ that satisfies the second condition of $\mathsf{Exp}_{\mathsf{completeness}}$.

Lemma 1. *If inputs* $(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\gamma}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{sk}^{(k)}_{\mathsf{EG}}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]})$ *of a* DB-SM *invocation satisfy the preconditions mentioned in Figure 8 and all the parties are honest then* $\mathcal{Q}$ *outputs* $I^* = \{i \in I \mid \exists j \in J : \sum_{k\in[m]} \boldsymbol{v}_i^{(k)} = \boldsymbol{v}'_j\}$.

Proof. Note that for honest mix-servers, the i^{th} DPK passes iff $\mathcal{M}_k$ uses $(\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})$ such that $(\boldsymbol{v}_i, \boldsymbol{r}_i, \boldsymbol{b}_i) := (\sum_{k\in[m]} \boldsymbol{v}_i^{(k)}, \sum_{k\in[m]} \boldsymbol{r}_i^{(k)}, \sum_{k\in[m]} \boldsymbol{b}_i^{(k)})$ satisfy the predicate p_{BB_i}. The equation $\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}$ of p_{BB_i} is trivially satisfied by the correctness of $(\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)})$. Next, note that $\tilde{\boldsymbol{\sigma}}_i = (\boldsymbol{\sigma}'_{\pi^{-1}(i)})^{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}}$, by the homomorphism of $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$. Let $j \in [n]$ be the index to which index i is mapped after permutation, i.e., $i = \pi(j)$ or equivalently $\pi^{-1}(i) = j$. Correctness of input conditions implies $\boldsymbol{v}'_j = \sum_{k\in[m]} \boldsymbol{v}^{(k)}_{\pi(j)} = \sum_{k\in[m]} \boldsymbol{v}_i^{(k)}$. Thus, $\tilde{\boldsymbol{\sigma}}_i = (\boldsymbol{\sigma}'_j)^{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}}$, which equals $g_1^{\frac{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}}{x+\boldsymbol{v}'_j}} = g_1^{\frac{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}}{x+\sum_{k\in[m]} \boldsymbol{v}_i^{(k)}}}$ if $j \in J$ and $g_1^{\sum_{k\in[m]} \boldsymbol{b}_i^{(k)}}$ if $j \notin J$. In the first case, the second equation of p_{BB_i} passes; in the second case, it fails. Since the DPK is run only for $i \in I$, I^* is exactly as claimed. ◻

Lemma 2. *If inputs* $(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma}, \boldsymbol{\rho_\gamma}, \boldsymbol{\epsilon_r}, \boldsymbol{v}', I, J, (\mathcal{M}_k[[\mathsf{sk}^{(k)}_{\mathsf{EG}}, \mathsf{sk}^{(k)}_{\mathsf{Pa}}, \pi^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)}]])_{k\in[m]})$ *of a* DB-RSM *invocation satisfy the preconditions mentioned in Figure 9 and all the parties are honest then* $\mathcal{Q}$ *outputs* $J^* = \{j \in J \mid \exists i \in I : \sum_{k\in[m]} \boldsymbol{v}_i^{(k)} = \boldsymbol{v}'_j\}$.

Proof. As in Lemma 1, let i be s.t. $i = \pi(j)$. By correctness of inputs, $\boldsymbol{v}'_j = \sum_{k\in[m]} \boldsymbol{v}_i^{(k)}$, $\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \sum_{k\in[m]} \boldsymbol{r}_i^{(k)})$. Then, it can be inspected that the following equalities hold:

- $\tilde{\boldsymbol{S}}'_j = (f_1 g_1^{\sum_{k\in[m]} \boldsymbol{v}_i^{(k)}} h_1^{\hat{\boldsymbol{r}}_i + \sum_{k\in[m]} \boldsymbol{r}_i^{(k)}})^{\frac{1}{x+\boldsymbol{c}_i}} g_1^{\sum_{k\in[m]} \boldsymbol{bs}_j^{(k)}}$ if $i \in I$ else f_1^0,
- $\tilde{\boldsymbol{c}}'_j = \boldsymbol{c}_i + \sum_{k\in[m]} \boldsymbol{b}_{\boldsymbol{c}}{}_j^{(k)}$,
- $\tilde{\boldsymbol{r}}'_j = \hat{\boldsymbol{r}}_i + \sum_{k\in[m]} \boldsymbol{r}_i^{(k)} + \sum_{k\in[m]} \boldsymbol{b}_{\boldsymbol{r}}{}_j^{(k)}$

Therefore, $(\tilde{\boldsymbol{S}}'_j g_1^{-\boldsymbol{b}_{\boldsymbol{S}_j}}, \tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}_j}, \tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}_j})$ satisfies the BBS+ verification equation $e(\tilde{\boldsymbol{S}}'_j g_1^{-\boldsymbol{b}_{\boldsymbol{S}_j}}, y f_2^{\tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}_j}}) = e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}_j}}, f_2)$ if $i \in I$, where

$(\boldsymbol{b}_{S\,j}, \boldsymbol{b}_{\boldsymbol{c}\,j}, \boldsymbol{b}_{\boldsymbol{r}\,j}) := (\sum_{k\in[m]} \boldsymbol{b}_{S\,j}^{(k)}, \sum_{k\in[m]} \boldsymbol{b}_{\boldsymbol{c}\,j}^{(k)}, \sum_{k\in[m]} \boldsymbol{b}_{\boldsymbol{r}\,j}^{(k)})$. So:

$$
\begin{aligned}
& e(\tilde{S}'_j g_1^{-\boldsymbol{b}_{S\,j}}, y f_2^{\tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}\,j}}) = e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}\,j}}, f_2) \\
\Leftrightarrow\ & \frac{e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}\,j}})}{e(g_1^{\boldsymbol{b}_{S\,j}}, y f_2^{\tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}\,j}})} = e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}\,j}}, f_2) \\
\Leftrightarrow\ & \frac{e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j})}{e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)} = \\
& e(\tilde{S}'_j, f_2)^{\boldsymbol{b}_{\boldsymbol{c}\,j}} e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})^{\boldsymbol{b}_{S\,j}} e(h_1, f_2)^{-\boldsymbol{b}_{\boldsymbol{r}\,j}} e(g_1, f_2)^{-\boldsymbol{b}_{S\,j}\boldsymbol{b}_{\boldsymbol{c}\,j}} \\
\Leftrightarrow\ & \mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{\boldsymbol{c}\,j}} \mathfrak{g}_2^{\boldsymbol{b}_{S\,j}} \mathfrak{h}_1^{\boldsymbol{b}_{\boldsymbol{r}\,j}} \mathfrak{h}_2^{\boldsymbol{b}_{S\,j}\boldsymbol{b}_{\boldsymbol{c}\,j}}
\end{aligned}
\tag{4}
$$

Thus, with the mix-servers holding shares of $(\boldsymbol{b}_{S\,j}, \boldsymbol{b}_{\boldsymbol{c}\,j}, \boldsymbol{b}_{\boldsymbol{r}\,j})$ and $\delta_1 := \boldsymbol{b}_{S\,j}\boldsymbol{b}_{\boldsymbol{c}\,j}$ and $\delta_2 := \delta_0\delta_1$, all equations of the predicate $p_{\mathsf{BBS+}_j}$ of the DPK are satisfied if $i \in I$. If $i \notin I$, the last equation of $p_{\mathsf{BBS+}_j}$ is not satisfied. The claim follows. □

C.2 Proof for Theorem 2

Suppose for contradiction that there is a PPT adversary $\mathcal{A}$ such that $\mathsf{Exp}_{\mathsf{soundness}}$ (Figure 4) outputs 1 with non-negligible probability. Note first that Q always outputs $I^* \subseteq I$ in DB-SM and $J^* \subseteq J$ in DB-RSM. Thus, $\boldsymbol{c}_{I^*} \subseteq \boldsymbol{c}_I$ and $\boldsymbol{v}'_{J^*} \subseteq \boldsymbol{v}'_J$. We now consider the following cases:

Case 1: $\boldsymbol{c}_{I^*} \neq \{\boldsymbol{c}_i \in \boldsymbol{c}_I \mid \boldsymbol{v}_i \in \boldsymbol{v}'_J\}$. This leads to the following sub-cases:

- *Case 1.1:* $\exists \boldsymbol{c}_i \in \boldsymbol{c}_{I^*} : \boldsymbol{v}_i \notin \boldsymbol{v}'_J$. As per Figure 7, $\boldsymbol{c}_i$ contains $\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}$ for some $\boldsymbol{r}_i \in \mathbb{Z}_q$. Since $\boldsymbol{c}_i \in \boldsymbol{c}_{I^*}$, $\boldsymbol{\gamma}_i \in \boldsymbol{\gamma}_{I^*}$. Thus, by Lemma 3, a PPT extractor can extract a tuple (j^*, r^*) such that $\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}'_{j^*}} h_1^{r^*}$ and $\boldsymbol{v}'_{j^*} \in \boldsymbol{v}'_J$. The requirement $\boldsymbol{v}_i \notin \boldsymbol{v}'_J$ thus implies that $\boldsymbol{v}_i \neq \boldsymbol{v}'_{j^*}$. This allows producing two different openings $(\boldsymbol{v}'_{j^*}, r^*)$ and $(\boldsymbol{v}_i, \boldsymbol{r}_i)$ for Pedersen commitment $\boldsymbol{\gamma}_i$, which is a contradiction under the discrete logarithm assumption in $\mathbb{G}_1$.
- *Case 1.2:* $\exists \boldsymbol{c}_i \in \boldsymbol{c}_{I \setminus I^*} : \boldsymbol{v}_i \in \boldsymbol{v}'_J$. Note that since Q produces a $\boldsymbol{c}_{I^*}$ and does not abort, it must be that $I^* \cup I^*_{\mathsf{c}} = I$ during the BTraceIn call. Thus, $I^*_{\mathsf{c}} = I \setminus I^*$. Further, since all $\boldsymbol{v}'_i$s are distinct, $\boldsymbol{v}_i \in \boldsymbol{v}'_J \implies \boldsymbol{v}_i \notin \boldsymbol{v}'_{[n]\setminus J}$. Thus, this case can be restated as follows: $\exists \boldsymbol{c}_i \in \boldsymbol{c}_{I^*_{\mathsf{c}}} : \boldsymbol{v}_i \notin \boldsymbol{v}'_{[n]\setminus J}$. Thus, by applying Lemma 3 for the second DB-SM call in BTraceIn and proceeding as the previous case, we conclude that this case is not possible.

Case 2: $\boldsymbol{v}'_{J^*} \neq \{\boldsymbol{v}'_j \in \boldsymbol{v}'_J \mid \boldsymbol{v}'_j \in \boldsymbol{v}_I\}$. This leads to the following sub-cases:

- *Case 2.1:* $\exists \boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*} : \boldsymbol{v}'_j \notin \boldsymbol{v}_I$. Since $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$, by Lemma 4, a PPT extractor can extract a tuple (i^*, r^*) such that $\boldsymbol{\gamma}_{i^*} = g_1^{\boldsymbol{v}'_j} h_1^{r^*}$ and $i^* \in I$. Since $i^* \in I$, the requirement $\boldsymbol{v}'_j \notin \boldsymbol{v}_I$ implies $\boldsymbol{v}'_j \neq \boldsymbol{v}_{i^*}$. As per Figure 7, $\boldsymbol{\gamma}_{i^*} \leftarrow g_1^{\boldsymbol{v}_{i^*}} h_1^{\boldsymbol{r}_{i^*}}$ for some $\boldsymbol{r}_{i^*} \in \mathbb{Z}_q$. This allows producing two different openings $(\boldsymbol{v}'_j, r^*)$ and $(\boldsymbol{v}_{i^*}, \boldsymbol{r}_{i^*})$ for $\boldsymbol{\gamma}_{i^*}$, which leads to a contradiction.
- *Case 2.2:* $\exists \boldsymbol{v}'_j \in \boldsymbol{v}'_{J\setminus J^*} : \boldsymbol{v}'_j \in \boldsymbol{v}_I$. Note that since Q produces a $\boldsymbol{v}'_{J^*}$ and does not abort, it must be that $J^* \cup J^*_{\mathsf{c}} = J$ during the BTraceIn call. Thus, $J^*_{\mathsf{c}} = J \setminus J^*$. Further, since all $\boldsymbol{v}_i$s are distinct, $\boldsymbol{v}'_j \in \boldsymbol{v}_I \implies \boldsymbol{v}'_j \notin \boldsymbol{v}_{[n]\setminus I}$. Thus, this case can be restated as follows: $\exists \boldsymbol{v}'_j \in \boldsymbol{v}_{J^*_{\mathsf{c}}} : \boldsymbol{v}'_j \notin \boldsymbol{v}_{[n]\setminus I}$. Thus, by applying Lemma 4 for the second DB-RSM call in BTraceOut and proceeding as the previous case, we conclude that this case is not possible.

1	$\underline{\mathcal{E}(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma}, \boldsymbol{\rho}_{\boldsymbol{\gamma}}, \boldsymbol{v}', I, J)}$:	
2	$\mathcal{E}$:	**for each** $i \in I$**:** $(\boldsymbol{v}_i, \boldsymbol{r}_i) \leftarrow \mathcal{E}_1(\boldsymbol{\gamma}_i, \boldsymbol{\rho}_{\boldsymbol{\gamma}\,i})$
3	$\mathcal{E} \to C$:	$(\boldsymbol{v}_i)_{i\in I}$ (BBS+ signature queries)
4	$C \to \mathcal{E}$:	$y, (\boldsymbol{\sigma}_i)_{i\in I}$
5	$\mathcal{E}$:	**for each** $i \in [n]$**:**
6		**if** $i \in I$**:** $(S_i, \boldsymbol{c}_i, \mathbf{r}_i) := \boldsymbol{\sigma}_i$; $\hat{\boldsymbol{r}}_i \leftarrow \mathbf{r}_i - \boldsymbol{r}_i$
7		**else:** $S_i \leftarrow f_1^0$; $\boldsymbol{c}_i \xleftarrow{\$} \mathbb{Z}_q$; $\hat{\boldsymbol{r}}_i \xleftarrow{\$} \mathbb{Z}_q$
8		**endfor**
9		$(\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}}) \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{S}), \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{c}),$ $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \hat{\boldsymbol{r}}))$
10	$\mathcal{E} \to \mathcal{A}$:	$y, \hat{\boldsymbol{\sigma}} := (\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}}), \boldsymbol{\epsilon}_{\hat{\sigma}} := (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$
11	$\mathcal{A} \to \mathcal{E}$:	$\tilde{\boldsymbol{\sigma}}' := (\tilde{\boldsymbol{S}}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}')$
12	$\mathcal{E}$:	**for each** $j \in J$**:**
13		Compute $\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2$ from $\tilde{\boldsymbol{\sigma}}'_j := (\tilde{S}'_j, \tilde{\boldsymbol{c}}'_j, \tilde{\boldsymbol{r}}'_j)$
14		$(\boldsymbol{b}_{S\,j}, \boldsymbol{b}_{\boldsymbol{c}\,j}, \boldsymbol{b}_{\boldsymbol{r}\,j}, \delta_0, \delta_1, \delta_2) \leftarrow \mathcal{E}_2(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2)$
15		$\boldsymbol{\sigma}'_j := (\tilde{S}'_j g_1^{-\boldsymbol{b}_{S\,j}}, \tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{\boldsymbol{c}\,j}, \tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{\boldsymbol{r}\,j})$
16		**for each** $i \in I$**:**
17		**if** $(\boldsymbol{v}_i = \boldsymbol{v}'_j)$**:** output $(i, \boldsymbol{r}_i)$
18		**endfor**
19		output $(\boldsymbol{v}'_j, \boldsymbol{\sigma}'_j)$
20		**endfor**

Figure 12: Extractor $\mathcal{E}$.

Lemma 3. *If* Q *participates in the* DB-SM *protocol with common input* $(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\gamma}, \boldsymbol{v}', I, J)$ *and outputs* I^**, then for all PPT adversaries* $\mathcal{A}$ *controlling* $(\mathcal{M}_k)_{k\in[m]}$ *and for all* $\boldsymbol{\gamma}_i \in \boldsymbol{\gamma}_{I^*}$*, there exists a PPT extractor* $\mathcal{E}$ *that outputs a* (j^*, r^*) *such that* $\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}'_{j^*}} h_1^{r^*} \wedge \boldsymbol{v}'_{j^*} \in \boldsymbol{v}'_J$.

Proof. Note that the verification steps of Q in DB-SM for a given $\boldsymbol{\gamma}_i \in \boldsymbol{\gamma}_I$ are exactly those of the verifier of the single-prover ZKP of set membership [19] for commitment $\boldsymbol{\gamma}_i$ against set $\phi := \boldsymbol{v}'_J$ when extended to our asymmetric pairing setting. Thus, for each $\boldsymbol{\gamma}_i \in \boldsymbol{\gamma}_{I^*}$, by the special soundness of the single-prover ZKP [19] under the n-Strong Diffie-Hellman assumption in $(\mathbb{G}_1, \mathbb{G}_2)$, a PPT extractor $\mathcal{E}'$ can extract v^*, r^* such that $\boldsymbol{\gamma}_i = g_1^{v^*} h_1^{r^*} \wedge v^* \in \phi$. $\mathcal{E}$ simply runs $\mathcal{E}'$, finds j^* such that $v^* = \boldsymbol{v}'_{j^*}$ and outputs (j^*, r^*). □

Lemma 4. *If* Q *participates in the* DB-RSM *protocol with common input* $(\mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\gamma}, \boldsymbol{\rho}_{\boldsymbol{\gamma}}, \boldsymbol{\epsilon}_{\boldsymbol{r}}, \boldsymbol{v}', I, J)$ *and outputs* J^**, then for all PPT adversaries* $\mathcal{A}$ *controlling* $(\mathcal{M}_k)_{k\in[m]}$ *and for all* $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$*, there exists a PPT extractor* $\mathcal{E}$ *that outputs an* (i^*, r^*) *such that* $\boldsymbol{\gamma}_{i^*} = g_1^{\boldsymbol{v}'_j} h_1^{r^*} \wedge i^* \in I$.

Proof. We construct an algorithm $\mathcal{E}$ that for each $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$ either outputs a desired tuple (i^*, r^*) or forges a BBS+ signature, using extractors $\mathcal{E}_1$ for the NIZK proofs of knowledge of commitment openings $\boldsymbol{\rho}_{\gamma_i}$ and $\mathcal{E}_2$ for the DPKs for $p_{\mathsf{BBS+}_j}$. On one end, $\mathcal{E}$ interacts with adversary $\mathcal{A}$ controlling $(\mathcal{M}_k)_{k\in[m]}$ in the DB-RSM protocol; on the other, with the challenger C of the BBS+ signature unforgeability game. See Figure 12.

For each $i \in I$, $\mathcal{E}$ first extracts opening $(\boldsymbol{v}_i, \boldsymbol{r}_i)$ of commitment $\boldsymbol{\gamma}_i$ from the NIZK proof $\boldsymbol{\rho}_{\boldsymbol{\gamma}\,i}$ using extractor $\mathcal{E}_1$ (line 2). It then obtains BBS+ public key y and signatures for each $(\boldsymbol{v}_i)_{i\in I}$ from the BBS+ challenger C (lines 3-4), derives quasi-signatures from them using

$\boldsymbol{r}_i$ (line 6) and forwards the quasi-signatures and their encrypted versions to $\mathcal{A}$, along with invalid quasi-signatures for $i \notin I$ — similar to $\mathcal{Q}$ (lines 5-10). $\mathcal{A}$ responds with blinded permuted signatures $\tilde{\boldsymbol{\sigma}}'$ (line 11), as $(\mathcal{M}_k)_{k\in[m]}$ do in the real protocol at the end of stage 1. In stage 2, for each $j \in J$, $\mathcal{E}$ extracts blinding factors for the blinded signature $\tilde{\boldsymbol{\sigma}}'_j$ using extractor $\mathcal{E}_2$ (lines 13-14), from which it obtains an unblinded signature $\boldsymbol{\sigma}'_j$ (line 15). $\mathcal{E}$ then attempts to find an opening $\boldsymbol{v}_i$ extracted by $\mathcal{E}_1$ for some commitment $\boldsymbol{\gamma}_i \in \boldsymbol{\gamma}_I$ such that $\boldsymbol{v}_i = \boldsymbol{v}'_j$, and returns the corresponding tuple $(i, \boldsymbol{r}_i)$ (lines 16-18). If no such $\boldsymbol{v}_i$ exists, it outputs the message-signature tuple $(\boldsymbol{v}'_j, \boldsymbol{\sigma}'_j)$ (line 19).

Note that $\mathcal{Q}$ produced some J^* and did not abort. In this case, the view produced by $\mathcal{E}$ to $\mathcal{A}$ is identical to that produced by $\mathcal{Q}$ to $(\mathcal{M}_k)_{k\in[m]}$. Further, for some $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$, if $\boldsymbol{v}_i = \boldsymbol{v}'_j$ for some $i \in I$, $(i, \boldsymbol{r}_i)$ output in line 17 is a desired tuple since $\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} = g_1^{\boldsymbol{v}'_j} h_1^{\boldsymbol{r}_i}$ and $i \in I$ (the first equality follows by the soundness of the proof of knowledge of commitment openings; the second because $\boldsymbol{v}_i = \boldsymbol{v}'_j$). If no such $\boldsymbol{v}_i$ exists, we show that the tuple $(\boldsymbol{v}'_j, \boldsymbol{\sigma}'_j)$ output in line 19 is a valid BBS+ signature forgery:

- Since for all $i \in I$, $\boldsymbol{v}_i \neq \boldsymbol{v}'_j$, a BBS+ signature for $\boldsymbol{v}'_j$ was not queried from C in line 3.
- Since $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$, the DPK for $p_{\mathsf{BBS+}_j}$ must have passed. Thus, by the soundness of DPKs:

$$\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{s\,j}} \mathfrak{h}_3^{\delta_0} \tag{5}$$

$$\mathfrak{z}_1^{\boldsymbol{b}_{c\,j}} = \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2}, \tag{6}$$

$$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c\,j}} \mathfrak{g}_2^{\boldsymbol{b}_{s\,j}} \mathfrak{h}_1^{\boldsymbol{b}_{r\,j}} \mathfrak{h}_2^{\delta_1} \tag{7}$$

From Equations 5 and 6, $\mathfrak{h}_2^{\boldsymbol{b}_{s\,j}\boldsymbol{b}_{c\,j}} \mathfrak{h}_3^{\delta_0 \boldsymbol{b}_{c\,j}} = \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2}$. It must be that $\delta_1 = \boldsymbol{b}_{s\,j}\boldsymbol{b}_{c\,j}$, otherwise two different openings (δ_1, δ_2) and $(\boldsymbol{b}_{s\,j}\boldsymbol{b}_{c\,j}, \delta_0 \boldsymbol{b}_{c\,j})$ for the Pedersen commitment $\mathfrak{z}_1^{\boldsymbol{b}_{c\,j}}$ can be produced. Equation 7 thus implies $\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c\,j}} \mathfrak{g}_2^{\boldsymbol{b}_{s\,j}} \mathfrak{h}_1^{\boldsymbol{b}_{r\,j}} \mathfrak{h}_2^{\boldsymbol{b}_{s\,j}\boldsymbol{b}_{c\,j}}$, which implies that $\boldsymbol{\sigma}'_j := (\tilde{S}'_j g_1^{-\boldsymbol{b}_{s\,j}}, \tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{c\,j}, \tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{r\,j})$ satisfies the BBS+ signature verification equation on message $\boldsymbol{v}'_j$ under public key y: $e(\tilde{S}'_j g_1^{-\boldsymbol{b}_{s\,j}}, y f_2^{\tilde{\boldsymbol{c}}'_j - \boldsymbol{b}_{c\,j}}) = e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j - \boldsymbol{b}_{r\,j}}, f_2)$ (see Equation 4; Lemma 2).

Since at most n signature queries could have been made, forging a BBS+ signature is not possible under the n-Strong Diffie Hellman assumption in $(\mathbb{G}_1, \mathbb{G}_2)$ [4]. Thus, for each $\boldsymbol{v}'_j \in \boldsymbol{v}'_{J^*}$, some $i \in I$ such that $\boldsymbol{v}_i = \boldsymbol{v}'_j$ must exist and a desired tuple $(i, \boldsymbol{r}_i)$ must have been produced. □

C.3 Proof for Theorem 3

We prove the theorem by considering the following sequence of hybrid experiments (the complete hybrid experiments are shown in Figures 13 to 32; solid boxes denote lines changed in a given hybrid and dashed boxes denote lines that change in the next hybrid):

- E_1 (Figure 13): E_1 is the original secrecy game $\mathsf{Exp}_{\mathsf{secrecy}}$ instantiated for our protocol.
- E_2 (Figure 14): In E_2, NIZKPKs $\boldsymbol{\rho}_{\boldsymbol{\gamma}_{i_0}}$ and $\boldsymbol{\rho}_{\boldsymbol{\gamma}_{i_1}}$ are simulated. E_2 is indistinguishable from E_1 in the random oracle model because of the ZK property of the NIZK proof.
- E_3 (Figure 15): In E_3, shares of $\boldsymbol{v}_i, \boldsymbol{r}_i$ for $i \in \{i_0, i_1\}$ are drawn as $(\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)})_{k\neq k^*} \xleftarrow{\$} \mathbb{Z}_q$ and $\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k\neq k^*} \boldsymbol{v}_i^{(k)}$, $\boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k\neq k^*} \boldsymbol{r}_i^{(k)}$. E_3 is indistinguishable from E_2 because the additive secret sharing is information-theoretically secure.
- E_4 (Figure 16): In E_4, instead of decrypting $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ for $i \in \{i_0, i_1\}$ during Mix, $\boldsymbol{v}_i^{(k^*)}$, $\boldsymbol{r}_i^{(k^*)}$ are obtained by directly using their corresponding values in the Enc call for i_0/i_1. E_4 is indistinguishable from E_3 because of correctness of decryption.
- E_5 (Figure 17): In E_5, instead of decrypting $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ for $i \in [n] \setminus \{i_0, i_1\}$ during Mix, $\boldsymbol{v}_i^{(k^*)}$, $\boldsymbol{r}_i^{(k^*)}$ are obtained by rerunning the Enc algorithm on input $\boldsymbol{v}_i$ for sender S_i using the randomnesses from the random tape issued to $\mathcal{A}$. E_5 is indistinguishable from E_4 because of correctness of decryption.
- E_6 (Figure 18): In E_6, encrypted shares $\mathbf{ev}_{i_0}^{(k^*)}, \mathbf{er}_{i_0}^{(k^*)}, \mathbf{ev}_{i_1}^{(k^*)}, \mathbf{er}_{i_1}^{(k^*)}$ in the Enc calls for i_0, i_1 are replaced by encryptions of 0. E_6 is indistinguishable from E_5 by the IND-CPA security of E.
- E_7 (Figure 19): In E_7, responses of $\mathcal{M}_k^*$ in $\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{TDec}$ and $\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}$ protocols are simulated by first obtaining the correct decryption m of the given ciphertext c using ideal decryption oracles $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}}$, $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}$ and then simulating $\mathcal{M}_k^*$'s responses using c, m and secret keys for $(\mathcal{M}_k)_{k\neq k^*}$. E_7 is indistinguishable from E_6 by the security of the threshold decryption protocols of $\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}$ and $\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}$. For $\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}$, it follows straightforwardly because given $c = (c_0, c_1) \in \mathbb{C}(\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}})$ and $m = c_1 / \prod_{k\in[m]} c_0^{\mathsf{sk}_{\mathsf{EG}}^{(k)}}$ obtained from $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}}$, the simulator can output $\dfrac{c_1}{m \prod_{k\neq k^*} c_0^{\mathsf{sk}_{\mathsf{EG}}^{(k)}}} = \dfrac{c_1 \prod_{k\in[m]} c_0^{\mathsf{sk}_{\mathsf{EG}}^{(k)}}}{c_1 \prod_{k\neq k^*} c_0^{\mathsf{sk}_{\mathsf{EG}}^{(k)}}} = c_0^{\mathsf{sk}_{\mathsf{EG}}^{(k^*)}}$, which is exactly the decryption share output by the real $\mathcal{M}_{k^*}$. For $\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}$, Damgård et al. provide a proof (see Theorem 4; [29]).
- E_8 (Figure 20): In E_8, instead of using $\boldsymbol{v}'$ given by $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}$ during Mix, it is set as $\boldsymbol{v}' \leftarrow (\boldsymbol{v}_{\pi(j)})_{j\in[n]}$. Here, $\pi \leftarrow \pi^{(m)} \circ \cdots \circ \pi^{(1)}$ is obtained using the experimenter-selected $\pi^{(k^*)}$ and $(\pi^{(k)})_{k\neq k^*}$ obtained from the random tape issued to $\mathcal{A}$, $(\boldsymbol{v}_i)_{i\in\{i_0,i_1\}}$ are obtained from their values in Enc calls for i_0, i_1, and $(\boldsymbol{v}_i)_{i\in[n]\setminus\{i_0,i_1\}}$ are obtained from $\mathcal{A}$'s input for senders $(S_i)_{i\in[n]\setminus\{i_0,i_1\}}$. E_8 is indistinguishable from E_7 by the correctness of $\mathsf{Shuffle}$ and $\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}$ protocols.
- E_9 (Figure 21): In E_9, instead of using $\tilde{\boldsymbol{\sigma}}$ given by $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}}$ during $\mathsf{DB\text{-}SM}$, it is set as $\tilde{\boldsymbol{\sigma}} \leftarrow ((\boldsymbol{\sigma}'_{\pi^{-1}(i)})^{\boldsymbol{b}_i})_{i\in[n]}$. Here $\boldsymbol{\sigma}'$ denotes BB signatures sent by $\mathcal{A}$ at the beginning of $\mathsf{DB\text{-}SM}$, π is as obtained in E_8 and $\boldsymbol{b}_i := \sum_{k\in[m]} \boldsymbol{b}_i^{(k)}$ is obtained using the experimenter-selected $\boldsymbol{b}_i^{(k^*)}$ and $(\boldsymbol{b}_i^{(k)})_{k\neq k^*}$ obtained from $\mathcal{A}$'s random tape. E_9 is indistinguishable from E_8 by the correctness of $\mathsf{Shuffle}$ and $\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{TDec}$ protocols.
- E_{10} (Figure 22): In E_{10}, instead of using $\tilde{\boldsymbol{S}}', \tilde{\boldsymbol{c}}'', \tilde{\boldsymbol{r}}''$ given by $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}}$ and $\mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}$ during $\mathsf{DB\text{-}RSM}$, they are set as $\tilde{\boldsymbol{S}}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b}_{s\,j}})_{j\in[n]}$, $\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{c\,j}')_{j\in[n]}$, $\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{r\,j}')_{j\in[n]}$. Here,

$(\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}})$ denote BBS+ quasi-signatures sent by $\mathcal{A}$ at the beginning of DB-RSM, π is as obtained in E_8, $\boldsymbol{b_S}_j \leftarrow \sum_{k\in[m]} \boldsymbol{b_S}_j^{(k)}$, $\boldsymbol{b_c}'_j \leftarrow \sum_{k\in[m]} \boldsymbol{b'_c}_j^{(k)} \bmod N$, $\boldsymbol{b_r}'_j \leftarrow \sum_{k\in[m]} \boldsymbol{b'_r}_j^{(k)} \bmod N$ are obtained using experimenter-selected $\boldsymbol{b_S}_j^{(k^*)}, \boldsymbol{b'_c}_j^{(k^*)} := \boldsymbol{b_c}_j^{(k^*)} + q\boldsymbol{\chi_c}_j^{(k^*)}, \boldsymbol{b'_r}_j^{(k^*)} := \boldsymbol{b_r}_j^{(k^*)} + q\boldsymbol{\chi_r}_j^{(k^*)}$ and $(\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b'_c}_j^{(k)} := \boldsymbol{b_c}_j^{(k)} + q\boldsymbol{\chi_c}_j^{(k)}, \boldsymbol{b'_r}_j^{(k)} := \boldsymbol{b_r}_j^{(k)} + q\boldsymbol{\chi_r}_j^{(k)})_{k\neq k^*}$ computed using $\mathcal{A}$'s random tape, $(\boldsymbol{r}_i)_{i\in\{i_0,i_1\}}$ are obtained from the Enc calls for i_0, i_1 and $(\boldsymbol{r}_i)_{i\in[n]\setminus\{i_0,i_1\}}$ are obtained from the random tape given to $\mathcal{A}$. E_{10} is indistinguishable from E_9 by the correctness of Shuffle, $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$.TDec and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$.TDec protocols.

- E_{11} (Figure 23): In E_{11}, all encryptions/re-encryptions under $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ are replaced by encryptions of 0. E_{11} is indistinguishable to E_{10} by the IND-CPA security of $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ under the DCR assumption [73].
- E_{12} (Figure 24): In E_{12}, all encryptions/re-encryptions under $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ are replaced by encryptions of g_1^0. E_{12} is indistinguishable to E_{11} by the IND-CPA security of $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ under the DDH assumption.
- E_{13} (Figure 25): In E_{13}, $\mathcal{M}_{k^*}$'s responses during the DPKs in DB-SM are simulated using shares of $(\mathcal{M}_k)_{k\neq k^*}$, as follows:
  - For $i \in \{i_0, i_1\}$, $\mathcal{M}_{k^*}$'s responses are simulated completely if $\pi^{-1}(i_0) \in J$; otherwise only the responses corresponding to the first equation $\boldsymbol{\gamma}_i = g_1^{\sum_{k\in[m]} \boldsymbol{v}_i^{(k)}} h_1^{\sum_{k\in[m]} \boldsymbol{r}_i^{(k)}}$ are simulated. For simulating the DPK completely, $(\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)})_{k\neq k^*}$ are obtained from their values in the Enc calls for i_0/i_1 and $(\boldsymbol{b}_i^{(k)})_{k\neq k^*}$ are chosen as the ones obtained in E_9. For simulating only the first component of the DPK, $\mathcal{M}_{k^*}$'s responses corresponding to the first equation are simulated using $(\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)})_{k\neq k^*}$ obtained from the Enc calls for i_0/i_1, but $\mathcal{M}_{k^*}$'s strategy is followed for the second component except that a freshly sampled $\boldsymbol{b}_i^{(k^*)}$ is used instead.
  - For $i \in [n] \setminus \{i_0, i_1\}$, the DPK is not simulated, i.e., values $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}$ are used exactly as $\mathcal{M}_{k^*}$ does.

  E_{13} is indistinguishable from E_{12} because *a*) by the correctness of the blinded signatures, the DPK for i_0 passes iff $j_0 = \pi^{-1}(i_0) \in J$ (otherwise the blinded signature is invalid), *b*) by the assert condition in the OTraceIn call, the DPK for i_1 passes iff the DPK for i_0 passes and *c*) even when these DPKs do not pass, their first component always passes.
- E_{14} (Figure 27): In E_{14}, $\mathcal{M}_{k^*}$'s responses during the DPKs in DB-RSM are simulated using shares of $(\mathcal{M}_k)_{k\neq k^*}$, as follows:
  - For $j \in \{j_0, j_1\}$ where j_0, j_1 are s.t. $v_0 = \boldsymbol{v}'_{j_0}$ and $v_1 = \boldsymbol{v}'_{j_1}$, the DPK is simulated completely if $\pi(j_0) \in I$; otherwise only its first two components are simulated. For simulating the DPK completely, $(\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)}, \boldsymbol{b_r}_j^{(k)})_{k\neq k^*}$ are chosen as the ones obtained in E_{10}, $(\delta_0^{(k)})_{k\neq k^*}$ are obtained from the random tape issued to $\mathcal{A}$ and $(\delta_1^{(k)}, \delta_2^{(k)})_{k\neq k^*}$ are obtained by applying the Mult algorithm to inputs $(\delta_0^{(k)}, \boldsymbol{b_S}_j^{(k)})$ and $(\delta_0^{(k)}, \boldsymbol{b_c}_j^{(k)})$ respectively. For simulating only the first two components, $\mathcal{M}_{k^*}$'s responses corresponding to the first two equations are simulated using $(\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)})_{k\neq k^*}$ obtained as above whereas for the third equation, $\mathcal{M}_{k^*}$'s strategy is followed except with a freshly sampled $\boldsymbol{b_r}_j^{(k^*)}$.
  - For $j \in [n] \setminus \{j_0, j_1\}$, the DPK is not simulated, i.e., values $(\boldsymbol{b_S}_j^{(k^*)}, \boldsymbol{b_c}_j^{(k^*)}, \boldsymbol{b_r}_j^{(k^*)}, \delta_0^{(k^*)}, \delta_1^{(k^*)}, \delta_2^{(k^*)})$ are used exactly as $\mathcal{M}_{k^*}$ does.

  E_{14} is indistinguishable from E_{13} because *a*) by the correctness of the blinded signatures, the DPK for j_0 passes iff $i_0 = \pi(j_0) \in I$ (otherwise the blinded signature is invalid), *b*) by the assert condition in the OTraceOut call, the DPK for j_1 passes iff the DPK for j_0 passes, and *c*) even when these DPKs do not pass, their first two equations do pass.
- E_{15} (Figure 28): In E_{15}, $\mathfrak{z}_1^{(k^*)}$ in DB-RSM is chosen uniformly at random from $\mathbb{Z}_q$. E_{15} is indistinguishable from E_{14} because of perfect blinding using $\delta_0^{(k^*)}$.
- E_{16} (Figure 29): In E_{16}, shares $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ during the Enc calls for i_0, i_1 are set as 0. E_{16} is indistinguishable from E_{15} because these shares are not used anymore.
- E_{17} (Figure 30): In E_{17}, commitments $\boldsymbol{\gamma}_{i_0}, \boldsymbol{\gamma}_{i_1}$ in above Enc calls commit 0. E_{17} is indistinguishable from E_{16} because Pedersen commitments are perfectly hiding and the committed values are not used anywhere.
- E_{18} (Figure 31): In E_{18}, $\tilde{\boldsymbol{\sigma}}_{i_0}$, $\tilde{\boldsymbol{\sigma}}_{i_1}$ during DB-SM are replaced by randomly drawn elements from $\mathbb{G}_1$. E_{18} is indistinguishable from E_{17} because $\boldsymbol{b}_{i_0}^{(k^*)}, \boldsymbol{b}_{i_1}^{(k^*)}$ used in computing $\tilde{\boldsymbol{\sigma}}_{i_0}$, $\tilde{\boldsymbol{\sigma}}_{i_1}$ are chosen uniformly at random from $\mathbb{Z}_q$.
- E_{19} (Figure 32): In E_{19}, $\tilde{\boldsymbol{S}}'_{j_0}$, $\tilde{\boldsymbol{S}}'_{j_1}$ during DB-RSM are replaced by randomly drawn elements from $\mathbb{G}_1$; $\tilde{\boldsymbol{c}}''_{j_0}$, $\tilde{\boldsymbol{c}}''_{j_1}$ are computed as $\tilde{\boldsymbol{c}}'''_{j_0} + \sum_{k\neq k^*} \boldsymbol{b'_c}_{j_0}^{(k)}$, $\tilde{\boldsymbol{c}}'''_{j_1} + \sum_{k\neq k^*} \boldsymbol{b'_c}_{j_1}^{(k)}$ where $\tilde{\boldsymbol{c}}'''_{j_0}, \tilde{\boldsymbol{c}}'''_{j_1} \xleftarrow{\$} \mathbb{Z}_{q+q^2}$ and $(\boldsymbol{b'_c}_{j_0}^{(k)}, \boldsymbol{b'_c}_{j_1}^{(k)})_{k\neq k^*}$ are the ones obtained in E_{10}; and $\tilde{\boldsymbol{r}}''_{j_0}$, $\tilde{\boldsymbol{r}}''_{j_1}$ are computed as $\tilde{\boldsymbol{r}}'''_{j_0} + \sum_{k\neq k^*} \boldsymbol{b'_r}_{j_0}^{(k)}, \tilde{\boldsymbol{r}}'''_{j_1} + \sum_{k\neq k^*} \boldsymbol{b'_r}_{j_1}^{(k)}$ where $\tilde{\boldsymbol{r}}'''_{j_0}, \tilde{\boldsymbol{r}}'''_{j_1} \xleftarrow{\$} \mathbb{Z}_{2q+q^2}$ and $(\boldsymbol{b'_r}_{j_0}^{(k)}, \boldsymbol{b'_r}_{j_1}^{(k)})_{k\neq k^*}$ are the ones obtained in E_{10}. E_{19} is indistinguishable from E_{18} because $\boldsymbol{b_S}_{j_0}^{(k^*)}, \boldsymbol{b_S}_{j_1}^{(k^*)}$ used in computing $\tilde{\boldsymbol{S}}'_{j_0}, \tilde{\boldsymbol{S}}'_{j_1}$ are chosen uniformly at random from $\mathbb{Z}_q$ and $\boldsymbol{b'_c}_{j_0}^{(k^*)}, \boldsymbol{b'_c}_{j_1}^{(k^*)}$ (resp. $\boldsymbol{b'_r}_{j_0}^{(k^*)}, \boldsymbol{b'_r}_{j_1}^{(k^*)}$) used in computing $\tilde{\boldsymbol{c}}''_{j_0}$, $\tilde{\boldsymbol{c}}''_{j_1}$ (resp. $\tilde{\boldsymbol{r}}''_{j_0}$, $\tilde{\boldsymbol{r}}''_{j_1}$) are chosen uniformly at random from an exponentially larger space than the corresponding messages $\boldsymbol{c}_{\pi(j_0)}, \boldsymbol{c}_{\pi(j_1)}$ (resp. $\mathbf{r}_{\pi(j_0)}, \mathbf{r}_{\pi(j_1)}$).

In E_{19}, $\mathcal{A}$'s view is identical for both $b = 0$ and $b = 1$ because in this experiment only $\boldsymbol{v}'$ sent at the end of the Mix protocol depends on v_{i_0}, v_{i_1} and $\boldsymbol{v}'$ is identically distributed for any choice of b because $\pi^{(k^*)}$ is uniformly distributed over $\mathsf{Perm}(n)$. Thus, by a standard hybrid argument, the theorem holds.

C.4 Proof for Theorem 4

In this case, the proof proceeds exactly as above except that in experiment E_{13}, DPKs for both i_0, i_1 are always simulated completely and in experiment E_{14}, DPKs for both j_0, j_1 are always simulated completely. E_{13} and E_{14} are indistinguishable from their previous experiments because in this case, even though $\mathcal{A}$ is not restricted by the constraints at OTraceIn/OTraceOut calls, $\mathcal{A}$ does not control Q anymore and thus does not even learn whether a DPK passed or not by the property of the DPKs (see Section 3.1.5).

C.5 Secrecy in the malicious model

In this section, we sketch a proof that when the additional steps of Appendix B are applied then our construction protects secrecy (Definition 4) even against general malicious adversaries. We assume that secure MPC protocols are used for Beaver triples generation and for $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ distributed key generation.

The proof essentially follows the structure of the proof in Appendix C.3 with the following differences:

- In E_5, $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ for $i \in [n] \setminus \{i_0, i_1\}$ are obtained by extracting them from proofs $(\boldsymbol{\rho}^{(k)}_{\mathsf{ev}_i}, \boldsymbol{\rho}^{(k)}_{\mathsf{er}_i})_{k\in[m]}$; the corresponding proofs for $i \in \{i_0, i_1\}$ are simulated. E_5 is indistinguishable from E_4 by knowledge soundness of $\boldsymbol{\rho}^{(k)}_{\mathsf{ev}_i}, \boldsymbol{\rho}^{(k)}_{\mathsf{er}_i}$ for $i \in [n] \setminus \{i_0, i_1\}$ and their zero-knowledgeness for $i \in \{i_0, i_1\}$.
- In E_8, $(\pi^{(k)})_{k\neq k^*}$ are obtained by extracting them from proofs of shuffle produced by $(\mathcal{M}_k)_{k\neq k^*}$ and $(\boldsymbol{v}_i)_{i\in[n]\setminus\{i_0,i_1\}}$ are obtained by extracting them from proofs $\boldsymbol{\rho}_{\epsilon_i}$ for $i \in [n] \setminus \{i_0, i_1\}$. Proofs of shuffle for $\mathcal{M}_{k^*}$ and proofs $\boldsymbol{\rho}_{\epsilon_i}$ for $i \in \{i_0, i_1\}$ are simulated. E_8 is indistinguishable to E_7 by *a*) knowledge soundness of proofs of shuffles for $(\mathcal{M}_k)_{k\neq k^*}$, *b*) knowledge soundness of $\boldsymbol{\rho}_{\epsilon_i}$ for $i \in [n] \setminus \{i_0, i_1\}$, *c*) zero-knowledgeness of proofs of shuffle for $\mathcal{M}_{k^*}$, *d*) zero-knowledgeness of $\boldsymbol{\rho}_{\epsilon_i}$ for $i \in \{i_0, i_1\}$, and *e*) correctness of threshold decryption protocol $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}$.
- In E_9, $(\boldsymbol{b}_i^{(k)})_{k\neq k^*}$ are obtained by extracting them from $(\boldsymbol{\rho}^{(k)}_{\tilde{\sigma}_i})_{k\neq k^*}$. The corresponding proofs for $k = k^*$ are simulated. E_9 is indistinguishable from E_8 by *a*) verification of correct construction of $\boldsymbol{\epsilon}'_\sigma$ from $\boldsymbol{\sigma}'$, *b*) knowledge soundness of proofs of shuffle that prove that each $(\mathcal{M}_k)_{k\neq k^*}$ shuffled encrypted signatures in this phase correctly using the inverse of the permutation $\pi^{(k)}$ it applied during mixing, *c*) knowledge soundness of $(\boldsymbol{\rho}^{(k)}_{\tilde{\sigma}_i})_{k\neq k^*}$, *d*) zero-knowledgeness of proofs of shuffle for $k = k^*$, *e*) zero-knowledgeness of $\boldsymbol{\rho}^{(k^*)}_{\tilde{\sigma}_i}$, and *f*) correctness of threshold decryption protocol $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}$.
- In E_{10}, $(\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b'_c}_j^{(k)}, \boldsymbol{b'_r}_j^{(k)})_{k\neq k^*}$ are obtained by extracting them from $(\boldsymbol{\rho}^{(k)}_{bS_j}, \boldsymbol{\rho}^{(k)}_{bc_j}, \boldsymbol{\rho}^{(k)}_{br_j})$ respectively and $\boldsymbol{r}_{\pi(j)}$ is obtained by extracting them from $\boldsymbol{\rho}_{\epsilon_{r_{\pi(j)}}}$. Proofs $(\boldsymbol{\rho}^{(k^*)}_{bS_j}, \boldsymbol{\rho}^{(k^*)}_{bc_j}, \boldsymbol{\rho}^{(k^*)}_{br_j})$ and $\boldsymbol{\rho}_{\epsilon_{r_{i_0}}}, \boldsymbol{\rho}_{\epsilon_{r_{i_1}}}$ are simulated. E_{10} is indistinguishable from E_9 by *a*) verification of correct construction of $(\boldsymbol{\epsilon_S}, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon_{\hat{r}}})$ from $(\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}})$, *b*) knowledge soundness of the proofs of shuffle that prove that each $\mathcal{M}_k$ shuffled encrypted signatures in this phase correctly using $\pi^{(k)}$ applied during mixing, *c*) knowledge soundness of $(\boldsymbol{\rho}^{(k)}_{bS_j}, \boldsymbol{\rho}^{(k)}_{bc_j}, \boldsymbol{\rho}^{(k)}_{br_j})$, *d*) zero-knowledgeness of proofs of shuffle for $k = k^*$, *e*) zero-knowledgeness of $\boldsymbol{\rho}^{(k^*)}_{bc_j}, \boldsymbol{\rho}^{(k^*)}_{br_j}$ and $\boldsymbol{\rho}_{\epsilon_{r_{i_0}}}, \boldsymbol{\rho}_{\epsilon_{r_{i_1}}}$, and *f*) correctness of threshold decryption protocols $\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{TDec}$ and $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}$.
- In E_{13}, the correctness of blinded signatures is ensured through the additional verification that signatures $\boldsymbol{\sigma}'_j$ for each $j \in J$ are valid BB signatures on $\boldsymbol{v}'_j$ and are invalid signatures for $j \notin J$.
- In E_{14}, the correctness of blinded signatures is ensured through the additional verification that quasi-signatures $(\boldsymbol{S}_i, \boldsymbol{c}_i, \hat{\boldsymbol{r}}_i)$ for each $i \in I$ are valid BBS+ quasi signatures using commitment $\boldsymbol{\gamma}_i$ and invalid for $i \notin I$. Note that correctness of quasi-signatures implies correctness of full BBS+ signatures for $i \in \{i_0, i_1\}$ because encryptions $\boldsymbol{\epsilon}_{r_i}$ are generated by honest senders. Further, in this case, $(\delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)})_{k\neq k^*}$ are obtained by extracting $\delta_0^{(k)}$ from proofs of knowledge of the opening of $(\mathfrak{z}_1^{(k)})_{k\neq k^*}$ and computing $(\delta_1^{(k)}, \delta_2^{(k)})_{k\neq k^*}$ using the Mult algorithm on inputs $(\delta_0^{(k)}, \boldsymbol{b_S}_j^{(k)})$ and $(\delta_0^{(k)}, \boldsymbol{b_c}_j^{(k)})$ respectively, where $\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)} := (\boldsymbol{b'_c}_j^{(k)} \bmod q)$ are the ones obtained in E_{10} (note that this requires Beaver triples held by $(\mathcal{M}_k)_{k\neq k^*}$ which are provided by the ideal functionality for the MPC protocol for Beaver triple generation).

D PRIVACY RISK ANALYSIS OF TRACEIN/TRACEOUT QUERIES

In this section, we provide formal rules to analyse the privacy risk impact of allowing a given set of TraceIn/TraceOut queries to a querier in an application. Since the goal of the analysis is to apriori decide whether to allow the queries or not, we assume that we only have query inputs and not their outputs. Further, we assume that no query's input depends on other queries' outputs. Given this, we adopt a static analysis approach where we conservatively estimate the information *potentially* leaked by the queries. If the actual information leaked by a query depends on its output, we conservatively assume that the query potentially leaks the information corresponding to both the outputs.

Consider a mixnet with input ciphertexts $(\boldsymbol{c}_i)_{i\in[n]}$ and output plaintexts $(\boldsymbol{v}'_j)_{j\in[n]}$. Let Q denote the set of proposed allowed queries containing elements of the form $(\mathsf{TraceIn}, i, J)$ and $(\mathsf{TraceOut}, I, j)$ for $i, j \in [n]$ and $I, J \subseteq [n]$ representing the TraceIn(i, J) and TraceOut(I, j) queries respectively ($\mathsf{BTraceIn}$/$\mathsf{BTraceOut}$ queries can be compiled to this format). Let $K(i, J)$ denote the fact that the querier potentially knows that ciphertext $\boldsymbol{c}_i$ encrypts a plaintext in set $\boldsymbol{v}'_J$. Let $K(I, j)$ (with an index set as the first argument and an index as the second argument) denote the fact that the querier potentially knows that plaintext $\boldsymbol{v}'_j$ is encrypted in some ciphertext in set $\boldsymbol{c}_I$. We therefore have the following rules:

- **R0.1:** $\forall i \in [n] : K(i, [n])$.
- **R0.2:** $\forall j \in [n] : K([n], j)$.
- **R1.1:** $\forall i \in [n], J \subseteq [n] : (\mathsf{TraceIn}, i, J) \in Q \implies K(i, J) \wedge K(i, [n] \setminus J)$.
- **R1.2:** $\forall j \in [n], I \subseteq [n] : (\mathsf{TraceOut}, I, j) \in Q \implies K(I, j) \wedge K([n] \setminus I, j)$.
- **R2.1:** $\forall i \in [n], J, J' \subseteq [n] : K(i, J) \wedge K(i, J') \implies K(i, J \cap J')$.
- **R2.2:** $\forall j \in [n], I, I' \subseteq [n] : K(I, j) \wedge K(I', j) \implies K(I \cap I', j)$.
- **R3.1:** $\forall i, j \in [n], I, J \subseteq [n] : K(i, J) \wedge K(I, j) \wedge i \notin I \wedge j \in J \implies K(i, J \setminus \{j\})$.
- **R3.2:** $\forall i, j \in [n], I, J \subseteq [n] : K(i, J) \wedge K(I, j) \wedge i \in I \wedge j \notin J \implies K(I \setminus \{i\}, j)$.

Rules R0.1 and R0.2 encode the fact that the querier initially knows that each input ciphertext encrypts a plaintext in the output list and each output plaintext is encrypted in a ciphertext in the input list. R1.1 (symmetrically R1.2) encodes that a TraceIn(i, J) query leads to either the querier learning that $\boldsymbol{c}_i$ encrypts a plaintext in $\boldsymbol{v}'_J$ (if the query passed) or that $\boldsymbol{c}_i$ encrypts a plaintext in $\boldsymbol{v}'_{[n]\setminus J}$ (if the query failed). Thus, potentially, the querier may learn both.

R2.1 (symmetrically R2.2) encodes that if the querier knows that $\boldsymbol{c}_i$ encrypts a value in $\boldsymbol{v}'_J$ and in $\boldsymbol{v}'_{J'}$ then it knows that $\boldsymbol{c}_i$ encrypts a value in $\boldsymbol{v}'_{J \cap J'}$. R3.1 (symmetrically R3.2) encodes that if the querier knows that $\boldsymbol{c}_i$ encrypts a value in $\boldsymbol{v}'_J$ but there exists a $j \in J$ that is known to be encrypted in a ciphertext in a set I that does not include i, then it knows that $\boldsymbol{c}_i$ cannot encrypt $\boldsymbol{v}'_j$ because of distinctness of encrypted values.

Given these rules, the goal of our analysis is to output a) for each $i \in [n]$, the smallest set $J_{\min_i}$ such that $K(i, J_{\min_i})$ holds; and b) for each $j \in [n]$, the smallest set $I_{\min_j}$ such that $K(I_{\min_j}, i)$ holds. Note that if for some i, no TraceIn(i, J) query exists in Q then $J_{\min_i} = [n]$ (similarly for $I_{\min_j}$). The $J_{\min}, I_{\min}$ outputs tell exactly what information is potentially leaked by the queries in Q. This information can directly feed to the application-level security analysis. Since each implication rule monotonically decreases the size of either the J set for $K(i, J)$ or the I set for $K(I, j)$, J_{min_i} and I_{min_j} can both be obtained in finite time for a finite set Q using a fixed-point algorithm.

$E_1(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

  $\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

  $\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i};$ $\boxed{\boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathsf{NIZKPK}\{((\boldsymbol{v}_i, \boldsymbol{r}_i)) : \boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}\}}$

  $\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

  $(\boldsymbol{v}_i^{(k)})_{k \in [m]} \leftarrow \mathsf{Share}_{m,m}(\boldsymbol{v}_i); (\boldsymbol{r}_i^{(k)})_{k \in [m]} \leftarrow \mathsf{Share}_{m,m}(\boldsymbol{r}_i)$

  $(\mathbf{ev}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \in [m]}$

  $(\mathbf{er}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \in [m]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \ldots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}(\boldsymbol{\epsilon}'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

for $i \in [n]$**:**

  $\boldsymbol{v}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{ev}_i^{(k^*)}); \boldsymbol{r}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{er}_i^{(k^*)})$

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}_{\pi^{(k^*)^{-1}}(j)}^{(k^*+1)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^{\boldsymbol{n}}$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

  with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathrm{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

  $\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^{\boldsymbol{n}}$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

  $\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

  $\tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

  $\tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

  $\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}{}^{(k^*)}} \mathfrak{h}_3^{\delta_0{}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

  $\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

  $(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

  with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

  $\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{\boldsymbol{r}_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 13: E_1: The original $\mathsf{Exp}_{\mathsf{secrecy}}$ experiment instantiated for our construction.

$E_2(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boxed{\rho_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)}$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$(\boldsymbol{v}_i^{(k)})_{k \in [m]} \leftarrow \mathsf{Share}_{m,m}(\boldsymbol{v}_i); (\boldsymbol{r}_i^{(k)})_{k \in [m]} \leftarrow \mathsf{Share}_{m,m}(\boldsymbol{r}_i)$

$(\mathbf{ev}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \in [m]}$

$(\mathbf{er}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \in [m]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \rho_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}(\boldsymbol{\epsilon}'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

for $i \in [n]$**:**

$\boldsymbol{v}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{ev}_i^{(k^*)}); \boldsymbol{r}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{er}_i^{(k^*)})$

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_\sigma}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E^{th}_{EG}.REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E^{th}_{EG}.TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, c, \hat{r}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_c{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_c{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_c{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_c{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E^{th}_{EG}.Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$
$\mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_c^{(k^*)} + q\boldsymbol{\chi}_c^{(k^*)}), \mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_r^{(k^*)} + q\boldsymbol{\chi}_r^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_c{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_c{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E^{th}_{EG}.TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E^{th}_{Pa}.TDec}(\tilde{\boldsymbol{\epsilon}}_c{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

$\tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E^{th}_{Pa}.TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}''$ **mod** $\boldsymbol{q}$; $\tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}''$ **mod** $\boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0{}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$
$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 14: E_2: Simulating $\rho_{\boldsymbol{\gamma}_{i_0}}$ and $\rho_{\boldsymbol{\gamma}_{i_1}}$.

$E_3(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1:

E: **for** $i \in \{i_0, i_1\}$:

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q;\ \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i};\ \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$\boxed{(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q}$

$\boxed{\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)};\ \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}}$

$(\mathbf{ev}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \in [m]}$

$(\mathbf{er}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \in [m]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$: $\boldsymbol{v}'_j \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}(\boldsymbol{\epsilon}'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

for $i \in [n]$:

$\boldsymbol{v}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{ev}_i^{(k^*)});\ \boldsymbol{r}_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{er}_i^{(k^*)})$

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$: // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, c, \hat{r}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

$\tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q};\ \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1};\ \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1};\ \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$:

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q;\ (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)};\ \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2);\ \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$
$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 15: E_3**: Send random values as shares to** $(\mathcal{M}_k)_{k \neq k^*}$**.**

$E_4(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, v_i)$

$\quad r_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{v_i} h_1^{r_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, r_i)$

$\quad (v_i^{(k)})_{k \neq k^*}, r_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad v_i^{(k^*)} \leftarrow v_i - \sum_{k \neq k^*} v_i^{(k)}; r_i^{(k^*)} \leftarrow r_i - \sum_{k \neq k^*} r_i^{(k)}$

$\quad (\mathbf{ev}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, v_i^{(k)}))_{k \in [m]}$

$\quad (\mathbf{er}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, r_i^{(k)}))_{k \in [m]}$

$E \rightarrow \mathcal{A}$: $\quad \boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$: $v'_j \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}(\boldsymbol{\epsilon}'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$\boxed{\text{Get } v_{i_0}^{(k^*)}, r_{i_0}^{(k^*)}, v_{i_1}^{(k^*)}, r_{i_1}^{(k^*)} \text{ from the Enc calls for } i_0, i_1}$

$\boxed{\textbf{for } i \in [n] \setminus \{i_0, i_1\}\textbf{:}}$

$\quad \boxed{v_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{ev}_i^{(k^*)}); r_i^{(k^*)} \leftarrow \mathsf{E.Dec}(\mathsf{sk}^{(k^*)}, \mathbf{er}_i^{(k^*)})}$

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (v_i^{(k^*)}, r_i^{(k^*)}, b_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

$\quad\quad$ with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{v_i} h_1^{r_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{b_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-v_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\quad \boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

$\quad \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

$\quad \tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

$\quad \tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \textbf{ mod } \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \textbf{ mod } \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\quad \delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\quad \mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0{}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $\quad (\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\quad \mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{v'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\quad \mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$\quad\quad (\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

$\quad\quad$ with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\quad\quad \mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{\boldsymbol{r}_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 16: E_4: Do not use $\mathsf{sk}^{(k^*)}$ for decrypting $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ for $i \in \{i_0, i_1\}$.

$E_5(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\quad \boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^H(\boldsymbol{\gamma}_i)$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$\quad (\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad \boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$\quad (\mathbf{ev}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \in [m]}$

$\quad (\mathbf{er}_i^{(k)})_{k \in [m]} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \in [m]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \dots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$: $\boldsymbol{v}_j' \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}(\boldsymbol{\epsilon}_j', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

$\boxed{\textbf{for } i \in [n] \setminus \{i_0, i_1\}\textbf{:}}$

$\quad \boxed{\text{Get } \boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)} \text{ from } \mathcal{A}\text{'s random tape and input for sender } S_i}$

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}_J' \iff v_1 \in \boldsymbol{v}_J'$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}_\sigma'$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_\sigma}_{\pi^{(k^*)^{-1}}(j)}^{(k^*+1)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

$\qquad$ with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S^{(k^*-1)}, \boldsymbol{\epsilon}_c^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{S}{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_c^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{c}{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S', \boldsymbol{\epsilon}_c', \boldsymbol{\epsilon}_{\mathbf{r}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\quad \boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{bS}^{(k^*)}, \boldsymbol{\epsilon}_{bc}^{(k^*)}, \boldsymbol{\epsilon}_{br}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

$\quad \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_c^{(k^*)} + q\boldsymbol{\chi}_c^{(k^*)}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_r^{(k^*)} + q\boldsymbol{\chi}_r^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{bS}^{(k)}, \boldsymbol{\epsilon}_{bc}^{(k)}, \boldsymbol{\epsilon}_{br}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{bS}^{(k)}, \boldsymbol{\epsilon}_c' \prod_{k \in [m]} \boldsymbol{\epsilon}_{bc}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{br}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_S', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

$\quad \tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_c', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

$\quad \tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\quad \delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}_j', y f_2^{\tilde{\boldsymbol{c}}_j'}) / e(f_1 g_1^{\boldsymbol{v}_j'} h_1^{\tilde{\boldsymbol{r}}_j'}, f_2)$

$\quad \mathfrak{g}_1 \leftarrow e(\tilde{S}_j', f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}_j'})$

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$\qquad (\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

$\qquad$ with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\qquad \mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 17: E_5: Do not use $\mathsf{sk}^{(k^*)}$ for decrypting $\mathbf{ev}_i^{(k^*)}, \mathbf{er}_i^{(k^*)}$ for $i \in [n] \setminus \{i_0, i_1\}$.

$E_6(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\quad \boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$\quad (\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad \boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$\quad \boxed{(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)}$

$\quad \boxed{(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}(\boldsymbol{\epsilon}'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

$\quad$ Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E^{th}_{EG}.REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

$E \leftrightarrow \mathcal{A}$: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathbf{E^{th}_{EG}.TDec}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$ // threshold decryption

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

$\quad\quad$ with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\quad \boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E^{th}_{EG}.Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

$\quad \mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}), \mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

$E \leftrightarrow \mathcal{A}$: $\tilde{S}' \leftarrow \mathbf{E^{th}_{EG}.TDec}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$ // threshold decryption

$\quad \tilde{\boldsymbol{c}}'' \leftarrow \mathbf{E^{th}_{Pa}.TDec}(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

$\quad \tilde{\boldsymbol{r}}'' \leftarrow \mathbf{E^{th}_{Pa}.TDec}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathcal{M}_k)_{k \neq k^*})$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\quad \delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$\quad (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\quad \mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0{}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $\quad (\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\quad \mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\quad \mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\quad \mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$\quad\quad (\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

$\quad\quad$ with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\quad\quad \mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 18: E_6: Replace $\mathbf{ev}_i^{(k^*)}$, $\mathbf{er}_i^{(k^*)}$ for $i \in \{i_0, i_1\}$ by encryptions of zeros.

$E_7(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\epsilon_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\epsilon_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\epsilon_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \epsilon_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\epsilon_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\epsilon^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \epsilon^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: $\boxed{\textbf{for } j \in [n]\textbf{: } \boldsymbol{v}'_j \leftarrow \mathcal{F}^{\mathsf{th}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\epsilon'_j, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})}$

$\mathcal{A} \leftrightarrow E$: $\boxed{\textbf{for } j \in [n]\textbf{: } \mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \epsilon'_j, \boldsymbol{v}'_j)}$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_\sigma}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\boxed{\tilde{\boldsymbol{\sigma}} \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathsf{sk}_{\mathsf{EG}}^{(k)})_{k \neq k^*})}$ // threshold decryption

$\mathcal{A} \leftrightarrow E$: $\boxed{\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})}$

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, c, \hat{r}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}^{(k^*-1)}, \boldsymbol{\epsilon_c}^{(k^*-1)}, \boldsymbol{\epsilon_{\mathbf{r}}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_S}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_S}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_c}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon_c}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_{\mathbf{r}}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon_{\mathbf{r}}}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}', \boldsymbol{\epsilon_c}', \boldsymbol{\epsilon_{\mathbf{r}}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_c^{(k^*)} + q\boldsymbol{\chi}_c^{(k^*)}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_r^{(k^*)} + q\boldsymbol{\chi}_r^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon_S}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon_c}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon_{\mathbf{r}}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

E: $\boxed{\tilde{S}' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_S', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathsf{sk}_{\mathsf{EG}}^{(k)})_{k \neq k^*})}$ // threshold decryption

$\boxed{\tilde{c}'' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_c', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})}$

$\boxed{\tilde{r}'' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})}$

$E \leftrightarrow \mathcal{A}$: $\boxed{\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')}$

$\boxed{\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c', \tilde{c}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{r}'')}$

E: $\tilde{c}' \leftarrow \tilde{c}'' \textbf{mod } q; \tilde{r}' \leftarrow \tilde{r}'' \textbf{mod } q$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$(\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{c}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{r}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$
$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 19: E_7: Simulate threshold decryption protocols using ideal decryption oracles.

$E_8(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \to \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \to E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \to E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^{H}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \to \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \to E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \to \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \dots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \to E$: $\boldsymbol{\epsilon}'$

E: $\boxed{\text{Get } (\pi^{(k)})_{k \neq k^*} \text{ using } \mathcal{A}\text{'s random tape; } \pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}}$

E: $\boxed{\textbf{for } j \in [n]\textbf{: } \boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}_{\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \to E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \to E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \to E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \to \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_\sigma}_{\pi^{(k^*)^{-1}}(j)}^{(k^*+1)}))_{j \in [n]}$

$\mathcal{A} \to E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \to \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \to E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\sigma}} \leftarrow \mathcal{F}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathbf{th}}(\tilde{\boldsymbol{\epsilon}}_\sigma, \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathsf{sk}_{\mathsf{EG}}^{(k)})_{k \neq k^*})$ // threshold decryption

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \to E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \to E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \to E$: $\boldsymbol{\epsilon_S}^{(k^*-1)}, \boldsymbol{\epsilon_c}^{(k^*-1)}, \boldsymbol{\epsilon_r}^{(k^*-1)}$ // shuffling

$E \to \mathcal{A}$: $\boldsymbol{\epsilon_S}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon_S}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \to \mathcal{A}$: $\boldsymbol{\epsilon_c}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon_c}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$E \to \mathcal{A}$: $\boldsymbol{\epsilon_r}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon_r}_{\pi^{(k^*)}(i)}^{(k^*-1)}))_{i \in [n]}$

$\mathcal{A} \to E$: $\boldsymbol{\epsilon_S}', \boldsymbol{\epsilon_c}', \boldsymbol{\epsilon_r}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \to \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$
$\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_c^{(k^*)} + q\boldsymbol{\chi}_c^{(k^*)}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_r^{(k^*)} + q\boldsymbol{\chi}_r^{(k^*)}))$

$\mathcal{A} \to E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_c' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: $\tilde{S}' \leftarrow \mathcal{F}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathbf{th}}(\tilde{\boldsymbol{\epsilon}}_S', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathsf{sk}_{\mathsf{EG}}^{(k)})_{k \neq k^*})$ // threshold decryption

$\tilde{\mathbf{c}}'' \leftarrow \mathcal{F}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathbf{th}}(\tilde{\boldsymbol{\epsilon}}_c', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})$

$\tilde{\mathbf{r}}'' \leftarrow \mathcal{F}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathbf{th}}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')$

$\mathcal{S}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c', \tilde{\mathbf{c}}''); \mathcal{S}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\mathbf{r}}'')$

E: $\tilde{\mathbf{c}}' \leftarrow \tilde{\mathbf{c}}'' \textbf{ mod } \boldsymbol{q}; \tilde{\mathbf{r}}' \leftarrow \tilde{\mathbf{r}}'' \textbf{ mod } \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$(\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \to \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b_S}_j^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \to E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\mathbf{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\mathbf{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\mathbf{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$
$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 20: E_8: Obtain v' without decryption.

$E_9(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{v}_i)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{r}_i)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\boxed{\text{Get } (\boldsymbol{b}_i^{(k)})_{k \neq k^*} \text{ from } \mathcal{A}\text{'s random tape.}}$ // threshold decryption

$\boxed{\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{r}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{r}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_c{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_c{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_c{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_c{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_c^{(k^*)} + q\boldsymbol{\chi}_c^{(k^*)}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_r^{(k^*)} + q\boldsymbol{\chi}_r^{(k^*)}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_c{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_c{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

E: $\tilde{S}' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_S{}', \mathsf{sk}_{\mathsf{EG}}^{(k^*)}, (\mathsf{sk}_{\mathsf{EG}}^{(k)})_{k \neq k^*})$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_c{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})$

$\tilde{\boldsymbol{r}}'' \leftarrow \mathcal{F}^{\mathbf{th}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \mathsf{sk}_{\mathsf{Pa}}^{(k^*)}, (\mathsf{sk}_{\mathsf{Pa}}^{(k)})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S{}', \tilde{S}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c{}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$(\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{c}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 21: E_9**: Obtain** $\tilde{\sigma}$ **during** DB-SM **without decryption.**

$E_{10}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k\neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k\neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k\neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i\in[n]\setminus\{i_0,i_1\}}$

E: **if** $(b=0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boxed{\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, v_i)}$

$r_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{v_i} h_1^{r_i}; \rho_{\gamma_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boxed{\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, r_i)}$

$(v_i^{(k)})_{k\neq k^*}, r_i^{(k)})_{k\neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$v_i^{(k^*)} \leftarrow v_i - \sum_{k\neq k^*} v_i^{(k)}; r_i^{(k^*)} \leftarrow r_i - \sum_{k\neq k^*} r_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k\neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, v_i^{(k)}))_{k\neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k\neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, r_i^{(k)}))_{k\neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k\in[m]}, \rho_{\gamma_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i\in[n]}$

E: $\boxed{\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}^{(k^*-1)})}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k\neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \cdots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $v'_j \leftarrow (v_{\pi(j)})_{j\in[n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, v'_j)$

Get $v_{i_0}^{(k^*)}, r_{i_0}^{(k^*)}, v_{i_1}^{(k^*)}, r_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $v_i^{(k^*)}, r_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in v'_J \iff v_1 \in v'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j\in[n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_\sigma^{\boldsymbol{b}^{(k^*)}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k\neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k\neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k\neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'{}_{\pi^{-1}(i)}^{\,b_i^{(k^*)} + \sum_{k\neq k^*} b_i^{(k)}})_{i\in[n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\sigma}_i, y), p_{\mathsf{BB}_i}, (v_i^{(k^*)}, r_i^{(k^*)}, b_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k\neq k^*}, \mathcal{Q}^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{v_i} h_1^{r_i} \wedge e(\tilde{\sigma}_i, y) = e(g_1, g_2)^{b_i} e(\tilde{\sigma}_i, g_2)^{-v_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{\epsilon}_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i\in[n]}$

$E \rightarrow \mathcal{A}$: $\boxed{\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i\in[n]}}$

$E \rightarrow \mathcal{A}$: $\boxed{\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i\in[n]}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),$

$\boxed{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)}))}$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k\neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k\in[m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k\in[m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k\in[m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

E: $\boxed{\text{Get } (\boldsymbol{r}_i)_{i\in[n]\setminus\{i_0,i_1\}}, (\boldsymbol{b}_S{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)})_{k\neq k^*} \text{ from } \mathcal{A}\text{'s random tape.}}$

$\boxed{\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k\neq k^*} \boldsymbol{b}_{S_j}^{(k)}})_{j\in[n]}}$ // threshold decryption

$\boxed{\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{c}_j}^{(k^*)} + \sum_{k\neq k^*} \boldsymbol{b}'_{\boldsymbol{c}_j}{}^{(k)})_{j\in[n]}}$

$\boxed{\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}_j}^{(k^*)} + q\boldsymbol{\chi}_{\boldsymbol{r}_j}^{(k^*)} + \sum_{k\neq k^*} \boldsymbol{b}'_{\boldsymbol{r}_j}{}^{(k)})_{j\in[n]}}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S{}', \tilde{S}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$(\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k\neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k\in[m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{c}'_j}) / e(f_1 g_1^{v'_j} h_1^{\tilde{r}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k\neq k^*}, \mathcal{Q}^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 22: E_{10}: Obtain $\tilde{S}', \tilde{c}'', \tilde{r}''$ during DB-RSM without decryption.

$E_{11}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boxed{\epsilon_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)}$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{v_i} h_1^{r_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boxed{\epsilon_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)}$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\epsilon_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \epsilon_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\epsilon^{(k^*-1)} := (\epsilon_i^{(k^*-1)})_{i \in [n]}$

E: $\boxed{\epsilon^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)}$

$E \rightarrow \mathcal{A}$: $\epsilon^{(k^*)} \leftarrow (\epsilon^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \epsilon^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: ϵ'

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \epsilon'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \epsilon'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\epsilon_{\boldsymbol{\sigma}}{}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boxed{\epsilon_{\boldsymbol{\sigma}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \epsilon_{\boldsymbol{\sigma}}{}^{(k^*+1)}_{\pi^{(k^*)^{-1}}(j)}))_{j \in [n]}}$

$\mathcal{A} \rightarrow E$: ϵ_σ

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\boxed{\tilde{\epsilon}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \epsilon_\sigma^{\boldsymbol{b}^{(k^*)}})}$

$\mathcal{A} \rightarrow E$: $(\tilde{\epsilon}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\epsilon}_\sigma \leftarrow \tilde{\epsilon}_\sigma^{(k^*)} (\tilde{\epsilon}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\epsilon}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\boldsymbol{\gamma}_i, \tilde{\boldsymbol{\sigma}}_i, y), p_{\mathsf{BB}_i}, (\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}, \boldsymbol{b}_i^{(k^*)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BB}_i} := (\boldsymbol{\gamma}_i = g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i} \wedge e(\tilde{\boldsymbol{\sigma}}_i, y) = e(g_1, g_2)^{\boldsymbol{b}_i} e(\tilde{\boldsymbol{\sigma}}_i, g_2)^{-\boldsymbol{v}_i})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\epsilon_S, \epsilon_{\boldsymbol{c}}, \epsilon_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\epsilon_{\mathbf{r}} \leftarrow \epsilon_{\hat{\boldsymbol{r}}} \epsilon_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\epsilon_S{}^{(k^*-1)}, \epsilon_{\boldsymbol{c}}{}^{(k^*-1)}, \epsilon_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boxed{\epsilon_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, \epsilon_S{}^{(k^*-1)}_{\pi^{(k^*)}(i)}))_{i \in [n]}}$

$E \rightarrow \mathcal{A}$: $\boxed{\epsilon_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}}$

$E \rightarrow \mathcal{A}$: $\boxed{\epsilon_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}}$

$\mathcal{A} \rightarrow E$: $\epsilon_S{}', \epsilon_{\boldsymbol{c}}{}', \epsilon_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $\boxed{(\epsilon_{\boldsymbol{b}S}^{(k^*)}, \epsilon_{\boldsymbol{b}\boldsymbol{c}}^{(k^*)}, \epsilon_{\boldsymbol{b}\boldsymbol{r}}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^{\boldsymbol{b}_S^{(k^*)}}),}$

$\boxed{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))}$

$\mathcal{A} \rightarrow E$: $(\epsilon_{\boldsymbol{b}S}^{(k)}, \epsilon_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \epsilon_{\boldsymbol{b}\boldsymbol{r}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\epsilon}_S{}', \tilde{\epsilon}_{\boldsymbol{c}}{}', \tilde{\epsilon}_{\mathbf{r}}{}') \leftarrow (\epsilon_S{}' \prod_{k \in [m]} \epsilon_{\boldsymbol{b}S}^{(k)}, \epsilon_{\boldsymbol{c}}{}' \prod_{k \in [m]} \epsilon_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \epsilon_{\mathbf{r}}{}' \prod_{k \in [m]} \epsilon_{\boldsymbol{b}\boldsymbol{r}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b}_S{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_{S_j}^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{c}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{c}_j})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{r}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{r}_j})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\epsilon}_S{}', \tilde{S}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\epsilon}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\epsilon}_{\mathbf{r}}{}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$(\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, Q^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{\boldsymbol{c}_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{\boldsymbol{r}_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 23: E_{11}: Send encryptions of zeros in place of all $\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}$ encryptions.

$E_{12}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^H(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \dots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}_{\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: ${\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boxed{{\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\boxed{\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})}$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow ({\boldsymbol{\sigma}'}_{\pi^{-1}(i)}^{\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

E: $r_{v_i}^{(k^*)}, r_{r_i}^{(k^*)}, r_{b_i}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$E \rightarrow \mathcal{A}$: $a_{\gamma_i}^{(k^*)} \leftarrow g_1^{r_{v_i}^{(k^*)}} h_1^{r_{r_i}^{(k^*)}}; a_{\tilde{\sigma}_i}^{(k^*)} \leftarrow e(g_1, g_2)^{r_{b_i}^{(k^*)}} e(\tilde{\sigma}_i, g_2)^{-r_{v_i}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(a_{\gamma_i}^{(k)}, a_{\tilde{\sigma}_i}^{(k)})_{k \neq k^*}$

E: $\mathrm{c} \leftarrow H(\gamma_i, \tilde{\sigma}_i, y, \prod_{k \in [m]} a_{\gamma_i}^{(k)}, \prod_{k \in [m]} a_{\tilde{\sigma}_i}^{(k)})$

$E \rightarrow \mathcal{A}$: $(z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, z_{b_i}^{(k^*)}) \leftarrow (r_{v_i}^{(k^*)} - \mathrm{c}v_i^{(k^*)}, r_{r_i}^{(k^*)} - \mathrm{c}r_i^{(k^*)}, r_{b_i}^{(k^*)} - \mathrm{c}b_i^{(k^*)})$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathrm{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_r$

$\mathcal{A} \rightarrow E$: ${\boldsymbol{\epsilon}_S}^{(k^*-1)}, {\boldsymbol{\epsilon}_{\boldsymbol{c}}}^{(k^*-1)}, {\boldsymbol{\epsilon}_{\mathbf{r}}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boxed{{\boldsymbol{\epsilon}_S}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}}$

$E \rightarrow \mathcal{A}$: ${\boldsymbol{\epsilon}_{\boldsymbol{c}}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: ${\boldsymbol{\epsilon}_{\mathbf{r}}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: ${\boldsymbol{\epsilon}_S}', {\boldsymbol{\epsilon}_{\boldsymbol{c}}}', {\boldsymbol{\epsilon}_{\mathbf{r}}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $\boxed{(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),}$

$\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $({\tilde{\boldsymbol{\epsilon}}_S}', {\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}}', {\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}}') \leftarrow ({\boldsymbol{\epsilon}_S}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, {\boldsymbol{\epsilon}_{\boldsymbol{c}}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, {\boldsymbol{\epsilon}_{\mathbf{r}}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, ({\boldsymbol{b}_S}^{(k)}, {\boldsymbol{b}'_{\boldsymbol{c}}}^{(k)}, {\boldsymbol{b}'_{\boldsymbol{r}}}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{{\boldsymbol{b}_S}_j^{(k^*)} + \sum_{k \neq k^*} {\boldsymbol{b}_S}_j^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + {\boldsymbol{b}_{\boldsymbol{c}}}_j^{(k^*)} + q{\boldsymbol{\chi}_{\boldsymbol{c}}}_j^{(k^*)} + \sum_{k \neq k^*} {\boldsymbol{b}'_{\boldsymbol{c}}}_j^{(k)})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + {\boldsymbol{b}_{\boldsymbol{r}}}_j^{(k^*)} + q{\boldsymbol{\chi}_{\boldsymbol{r}}}_j^{(k^*)} + \sum_{k \neq k^*} {\boldsymbol{b}'_{\boldsymbol{r}}}_j^{(k)})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, {\tilde{\boldsymbol{\epsilon}}_S}', \tilde{S}')$

$\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, {\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}}', \tilde{\boldsymbol{c}}''); \boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, {\tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{{\boldsymbol{b}_S}_j^{(k^*)}} \mathfrak{h}_3^{{\delta_0}^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$

$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \boldsymbol{Q}^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{{\boldsymbol{b}_S}_j} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-{\boldsymbol{b}_{\boldsymbol{c}}}_j} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$

$\mathfrak{z}_2 = \mathfrak{g}_1^{{\boldsymbol{b}_{\boldsymbol{c}}}_j} \mathfrak{g}_2^{{\boldsymbol{b}_S}_j} \mathfrak{h}_1^{{\boldsymbol{b}_{\boldsymbol{r}}}_j} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 24: E_{12}: Send encryptions of g_1^0 in place of all $\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}$ encryptions.

$E_{13}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\epsilon_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\epsilon_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\epsilon_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \epsilon_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E^{th}_{EG}.Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g_1^0})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E^{th}_{EG}.TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: **for** $i \in I$**:** // stage 2

$\boxed{\textbf{if } i \in \{i_0, i_1\} \textbf{ and } v_0 \in \boldsymbol{v}'_J\textbf{:}}$ // simulate the entire DPK

$\boxed{z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, z_{b_i}^{(k^*)}, \mathsf{c} \xleftarrow{\$} \mathbb{Z}_q}$

$E \rightarrow \mathcal{A}$: $\boxed{a_{\gamma_i}^{(k^*)} \leftarrow g_1^{z_{v_i}^{(k^*)}} h_1^{z_{r_i}^{(k^*)}} (\gamma_i / \prod_{k \neq k^*} g_1^{v_i^{(k)}} g_1^{r_i^{(k)}})^{\mathsf{c}}}$

$E \rightarrow \mathcal{A}$: $\boxed{a_{\tilde{\sigma}_i}^{(k^*)} \leftarrow e(g_1, g_2)^{z_{b_i}^{(k^*)}} e(\tilde{\sigma}_i, g_2)^{-z_{v_i}^{(k^*)}} \left(\frac{e(\tilde{\sigma}_i, y)}{\prod_{k \neq k^*} e(g_1, g_2)^{b_i^{(k)}} e(\tilde{\sigma}_i, g_2)^{-v_i^{(k)}}}\right)^{\mathsf{c}}}$

$\mathcal{A} \rightarrow E$: $\boxed{(a_{\gamma_i}^{(k)}, a_{\tilde{\sigma}_i}^{(k)})_{k \neq k^*}}$

E: $\boxed{\text{Set } H(\gamma_i, \tilde{\sigma}_i, y, \prod_{k \in [m]} a_{\gamma_i}^{(k)}, \prod_{k \in [m]} a_{\tilde{\sigma}_i}^{(k)}) = \mathsf{c}}$

$E \rightarrow \mathcal{A}$: $\boxed{z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, z_{b_i}^{(k^*)}}$

$\boxed{\textbf{else if } i \in \{i_0, i_1\} \textbf{ and } v_0 \notin \boldsymbol{v}'_J\textbf{:}}$ // simulate the first component of the DPK

$\boxed{z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, r_{v_i}^{(k^*)}, r_{b_i}^{(k^*)}, \mathsf{c} \xleftarrow{\$} \mathbb{Z}_q}$

$E \rightarrow \mathcal{A}$: $\boxed{a_{\gamma_i}^{(k^*)} \leftarrow g_1^{z_{v_i}^{(k^*)}} h_1^{z_{r_i}^{(k^*)}} (\gamma_i / \prod_{k \neq k^*} g_1^{v_i^{(k)}} g_1^{r_i^{(k)}})^{\mathsf{c}}}$

$E \rightarrow \mathcal{A}$: $\boxed{a_{\tilde{\sigma}_i}^{(k^*)} \leftarrow e(g_1, g_2)^{r_{b_i}^{(k^*)}} e(\tilde{\sigma}_i, g_2)^{-r_{v_i}^{(k^*)}}}$

$\mathcal{A} \rightarrow E$: $\boxed{(a_{\gamma_i}^{(k)}, a_{\tilde{\sigma}_i}^{(k)})_{k \neq k^*}}$

E: $\boxed{\text{Set } H(\gamma_i, \tilde{\sigma}_i, y, \prod_{k \in [m]} a_{\gamma_i}^{(k)}, \prod_{k \in [m]} a_{\tilde{\sigma}_i}^{(k)}) = \mathsf{c}}$

$\boxed{b_i^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; \quad z_{b_i}^{(k^*)} \leftarrow r_{b_i}^{(k^*)} - \mathsf{c} b_i^{(k^*)}}$

$E \rightarrow \mathcal{A}$: $\boxed{z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, z_{b_i}^{(k^*)}}$

$\boxed{\textbf{else:}}$ // proceed as $\mathcal{M}_{k^*}$

E: $\boxed{r_{v_i}^{(k^*)}, r_{r_i}^{(k^*)}, r_{b_i}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q}$

$E \rightarrow \mathcal{A}$: $\boxed{a_{\gamma_i}^{(k^*)} \leftarrow g_1^{r_{v_i}^{(k^*)}} h_1^{r_{r_i}^{(k^*)}}; a_{\tilde{\sigma}_i}^{(k^*)} \leftarrow e(g_1, g_2)^{r_{b_i}^{(k^*)}} e(\tilde{\sigma}_i, g_2)^{-r_{v_i}^{(k^*)}}}$

$\mathcal{A} \rightarrow E$: $\boxed{(a_{\gamma_i}^{(k)}, a_{\tilde{\sigma}_i}^{(k)})_{k \neq k^*}}$

E: $\boxed{\mathsf{c} \leftarrow H(\gamma_i, \tilde{\sigma}_i, y, \prod_{k \in [m]} a_{\gamma_i}^{(k)}, \prod_{k \in [m]} a_{\tilde{\sigma}_i}^{(k)})}$

$E \rightarrow \mathcal{A}$: $\boxed{(z_{v_i}^{(k^*)}, z_{r_i}^{(k^*)}, z_{b_i}^{(k^*)}) \leftarrow (r_{v_i}^{(k^*)} - \mathsf{c} v_i^{(k^*)}, r_{r_i}^{(k^*)} - \mathsf{c} r_i^{(k^*)}, r_{b_i}^{(k^*)} - \mathsf{c} b_i^{(k^*)}}$

E: **endfor**

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon_S}, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon_{\hat{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathsf{r}} \leftarrow \boldsymbol{\epsilon_{\hat{r}}} \boldsymbol{\epsilon_r}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}^{(k^*-1)}, \boldsymbol{\epsilon_c}^{(k^*-1)}, \boldsymbol{\epsilon_r}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_S}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_c}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_r}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}', \boldsymbol{\epsilon_c}', \boldsymbol{\epsilon_r}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E^{th}_{EG}.Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g_1^0}),$
$\mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E^{th}_{Pa}.Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{\epsilon}}_r') \leftarrow (\boldsymbol{\epsilon_S}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon_c}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon_r}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b_S}^{(k)}, \boldsymbol{b}_c'^{(k)}, \boldsymbol{b}_r'^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{\boldsymbol{S}}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b_S}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b_S}_j^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b_c}_j^{(k^*)} + q \boldsymbol{\chi_c}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{c}j}{}^{(k)})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b_r}_j^{(k^*)} + q \boldsymbol{\chi_r}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{r}j}{}^{(k)})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E^{th}_{EG}.TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{S}}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E^{th}_{Pa}.TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E^{th}_{Pa}.TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_r', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \bmod \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \bmod \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{c_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b_S}_j^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $(\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{c}'_j}) / e(f_1 g_1^{v'_j} h_1^{\tilde{r}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}'_j})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{DPK}((\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2), p_{\mathsf{BBS}+_j},$
$(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)}, \delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}), (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*}, \mathcal{Q}^{\mathcal{A}})$

with $p_{\mathsf{BBS}+_j} := (\mathfrak{z}_1 = \mathfrak{h}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_3^{\delta_0} \wedge 1_{\mathbb{G}_T} = \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}} \mathfrak{h}_2^{\delta_1} \mathfrak{h}_3^{\delta_2} \wedge$
$\mathfrak{z}_2 = \mathfrak{g}_1^{\boldsymbol{b}_{c_j}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}} \mathfrak{h}_2^{\delta_1})$

E: **endfor**

Figure 25: E_{13}**: Simulate DPKs in** DB-SM **call.**

$E_{13}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad \boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^{H}(\boldsymbol{\gamma}_i)$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad (\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad \boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$\quad (\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$\quad (\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\quad \boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \dots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}_{\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

$\quad$ Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\boldsymbol{\mathcal{A}}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\boxed{\mathcal{S}_{\mathsf{DPK\text{-}DB\text{-}SM}}^{H}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},}$

$\boxed{((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})}$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S', \boldsymbol{\epsilon}_{\boldsymbol{c}}', \boldsymbol{\epsilon}_{\mathbf{r}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$ $\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b_S}^{(k)}, \boldsymbol{b_c'}^{(k)}, \boldsymbol{b_r'}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b_S}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b_S}_j^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b_c}_j^{(k^*)} + q \boldsymbol{\chi_c}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b'_c}_j^{(k)})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b_r}_j^{(k^*)} + q \boldsymbol{\chi_r}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b'_r}_j^{(k)})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\boldsymbol{\mathcal{A}}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')$

$\boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\boldsymbol{\mathcal{A}}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{c}}''); \boldsymbol{\mathcal{S}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\boldsymbol{\mathcal{A}}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

$\quad \delta_0^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; (\delta_1^{(k^*)}, \delta_2^{(k^*)}) \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}), \mathsf{Mult}(\delta_0^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)}))$

$E \rightarrow \mathcal{A}$: $\quad \mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b_S}_j^{(k^*)}} \mathfrak{h}_3^{\delta_0^{(k^*)}}$

$\mathcal{A} \rightarrow E$: $\quad (\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\quad \mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\quad \mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

$\quad r_{bS}^{(k^*)}, r_{bc}^{(k^*)}, r_{br}^{(k^*)}, r_{\delta_0}^{(k^*)}, r_{\delta_1}^{(k^*)}, r_{\delta_2}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q$

$E \rightarrow \mathcal{A}$: $\quad a_{\mathfrak{z}_1}^{(k^*)} \leftarrow \mathfrak{h}_2^{r_{bS}^{(k^*)}} \mathfrak{h}_3^{r_{\delta_0}^{(k^*)}}; a_{\mathfrak{z}'_1}^{(k^*)} \leftarrow \mathfrak{z}_1^{-r_{bc}^{(k^*)}} \mathfrak{h}_2^{r_{\delta_1}^{(k^*)}} \mathfrak{h}_3^{r_{\delta_2}^{(k^*)}}$

$E \rightarrow \mathcal{A}$: $\quad a_{\mathfrak{z}_2}^{(k^*)} \leftarrow \mathfrak{g}_1^{r_{bc}^{(k^*)}} \mathfrak{g}_2^{r_{bS}^{(k^*)}} \mathfrak{h}_1^{r_{br}^{(k^*)}} \mathfrak{h}_2^{r_{\delta_1}^{(k^*)}}$

$E \leftrightarrow \mathcal{A}$: $\quad a_{\mathfrak{z}_1}^{(k^*)}, a_{\mathfrak{z}'_1}^{(k^*)}, a_{\mathfrak{z}_2}^{(k^*)}; (a_{\mathfrak{z}_1}^{(k)}, a_{\mathfrak{z}'_1}^{(k)}, a_{\mathfrak{z}_2}^{(k)})_{k \neq k^*}$

E: $\quad \mathrm{c} \leftarrow H(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2, \prod_{k \in [m]} a_{\mathfrak{z}_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}'_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}_2}^{(k)})$

$\quad z_{bS}^{(k^*)} \leftarrow r_{bS}^{(k^*)} - \mathrm{c}\boldsymbol{b_S}_j^{(k^*)}; z_{bc}^{(k^*)} \leftarrow r_{bc}^{(k^*)} - \mathrm{c}\boldsymbol{b_c}_j^{(k^*)}; z_{br}^{(k^*)} \leftarrow r_{br}^{(k^*)} - \mathrm{c}\boldsymbol{b_r}_j^{(k^*)}$

$\quad z_{\delta_0}^{(k^*)} \leftarrow r_{\delta_0}^{(k^*)} - \mathrm{c}\delta_0^{(k^*)}; z_{\delta_1}^{(k^*)} \leftarrow r_{\delta_1}^{(k^*)} - \mathrm{c}\delta_1^{(k^*)}; z_{\delta_2}^{(k^*)} \leftarrow r_{\delta_2}^{(k^*)} - \mathrm{c}\delta_2^{(k^*)}$

$E \rightarrow \mathcal{A}$: $\quad z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}$

E: **endfor**

Figure 26: Same as E_{13} but DPK simulation steps during DB-SM **are abstracted out.**

$E_{14}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1:

E: **for** $i \in \{i_0, i_1\}$:

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^{H}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}_{\pi^{(k^*)}(1)}^{(k^*)}, \dots, \boldsymbol{\epsilon}_{\pi^{(k^*)}(n)}^{(k^*)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$: $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$: $\mathcal{S}_{\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$:

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}_{\mathsf{DPK\text{-}DB\text{-}SM}}^{H}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S^{(k^*-1)}, \boldsymbol{\epsilon}_c^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_c^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_c{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{bS}^{(k^*)}, \boldsymbol{\epsilon}_{bc}^{(k^*)}, \boldsymbol{\epsilon}_{br}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{bS}^{(k)}, \boldsymbol{\epsilon}_{bc}^{(k)}, \boldsymbol{\epsilon}_{br}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_c{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{bS}^{(k)}, \boldsymbol{\epsilon}_c{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{bc}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{br}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b}_S{}^{(k)}, \boldsymbol{b}'_c{}^{(k)}, \boldsymbol{b}'_r{}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (\boldsymbol{S}_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_{S_j}^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{c_j}^{(k^*)} + q \boldsymbol{\chi}_{c_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{c_j})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{r_j}^{(k^*)} + q \boldsymbol{\chi}_{r_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{r_j})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S{}', \tilde{S}')$

$\mathcal{S}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c{}', \tilde{\boldsymbol{c}}''); \mathcal{S}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}^{\mathcal{A}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \mathbf{mod}\ \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \mathbf{mod}\ \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$:

if $\boldsymbol{v}'_j \in \{v_0, v_1\}$:

E: $\delta_0{}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; \mathfrak{z}_1^{(k^*)} \leftarrow \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k^*)}} \mathfrak{h}_3^{\delta_0{}^{(k^*)}}$

$E \leftrightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)}, (\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{S}'_j, y f_2^{\tilde{c}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{r}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{S}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{c}'_j})$

Get $(\delta_0^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape

$(\delta_1^{(k)}, \delta_2^{(k)})_{k \neq k^*} \leftarrow (\mathsf{Mult}(\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}), \mathsf{Mult}(\delta_0^{(k)}, \boldsymbol{b}_{c_j}^{(k)}))_{k \neq k^*}$

if $i_0 \in I$: // simulate the entire DPK

$z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}, \mathrm{c} \xleftarrow{\$} \mathbb{Z}_q$

$a_{\mathfrak{z}_1}^{(k^*)} \leftarrow \mathfrak{h}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_0}^{(k^*)}} (\mathfrak{z}_1 / \prod_{k \neq k^*} \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k)}} \mathfrak{h}_3^{\delta_0^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}'_1}^{(k^*)} \leftarrow \mathfrak{z}_1^{-z_{bc}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_2}^{(k^*)}} (1_{\mathbb{G}_T} / \prod_{k \neq k^*} \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}} \mathfrak{h}_3^{\delta_2^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}_2}^{(k^*)} \leftarrow \mathfrak{g}_1^{z_{bc}^{(k^*)}} \mathfrak{g}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_1^{z_{br}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} (\mathfrak{z}_2 / \prod_{k \neq k^*} \mathfrak{g}_1^{\boldsymbol{b}_{c_j}^{(k)}} \mathfrak{g}_2^{\boldsymbol{b}_{S_j}^{(k)}} \mathfrak{h}_1^{\boldsymbol{b}_{r_j}^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}})^{\mathrm{c}}$

$E \leftrightarrow \mathcal{A}$: $a_{\mathfrak{z}_1}^{(k^*)}, a_{\mathfrak{z}'_1}^{(k^*)}, a_{\mathfrak{z}_2}^{(k^*)}; (a_{\mathfrak{z}_1}^{(k)}, a_{\mathfrak{z}'_1}^{(k)}, a_{\mathfrak{z}_2}^{(k)})_{k \neq k^*}$

Set $H(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2, \prod_{k \in [m]} a_{\mathfrak{z}_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}'_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}_2}^{(k)}) = \mathrm{c}$

$E \rightarrow \mathcal{A}$: $z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}$

else: // simulate the first two components of the DPK

$z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}, r_{bS}^{(k^*)}, r_{bc}^{(k^*)}, r_{br}^{(k^*)}, r_{\delta_1}^{(k^*)}, \mathrm{c} \xleftarrow{\$} \mathbb{Z}_q$

$a_{\mathfrak{z}_1}^{(k^*)} \leftarrow \mathfrak{h}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_0}^{(k^*)}} (\mathfrak{z}_1 / \prod_{k \neq k^*} \mathfrak{h}_2^{\boldsymbol{b}_{S_j}^{(k)}} \mathfrak{h}_3^{\delta_0^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}'_1}^{(k^*)} \leftarrow \mathfrak{z}_1^{-z_{bc}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_2}^{(k^*)}} (1_{\mathbb{G}_T} / \prod_{k \neq k^*} \mathfrak{z}_1^{-\boldsymbol{b}_{c_j}^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}} \mathfrak{h}_3^{\delta_2^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}_2}^{(k^*)} \leftarrow \mathfrak{g}_1^{r_{bc}^{(k^*)}} \mathfrak{g}_2^{r_{bS}^{(k^*)}} \mathfrak{h}_1^{r_{br}^{(k^*)}} \mathfrak{h}_2^{r_{\delta_1}^{(k^*)}}$

$E \leftrightarrow \mathcal{A}$: $a_{\mathfrak{z}_1}^{(k^*)}, a_{\mathfrak{z}'_1}^{(k^*)}, a_{\mathfrak{z}_2}^{(k^*)}; (a_{\mathfrak{z}_1}^{(k)}, a_{\mathfrak{z}'_1}^{(k)}, a_{\mathfrak{z}_2}^{(k)})_{k \neq k^*}$

Set $H(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2, \prod_{k \in [m]} a_{\mathfrak{z}_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}'_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}_2}^{(k)}) = \mathrm{c}$

$b_{r_j}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; z_{br}^{(k^*)} \leftarrow z_{br}^{(k^*)} - \mathrm{c} b_{r_j}^{(k^*)}$

$E \rightarrow \mathcal{A}$: $z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}$

else:

. . . // proceed as $\mathcal{M}_{k^*}$

E: **endfor**

Figure 27: E_{14}**: Simulate DPKs during** DB-RSM**.**

$E_{15}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_0, v_1)$ **else** $(\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\epsilon_i \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{v_i} h_1^{r_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}_{\mathsf{NIZKPK}}^H(\boldsymbol{\gamma}_i)$

$\epsilon_{r_i} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_g$

$\boldsymbol{v}_i^{(k^*)} \leftarrow \boldsymbol{v}_i - \sum_{k \neq k^*} \boldsymbol{v}_i^{(k)}; \boldsymbol{r}_i^{(k^*)} \leftarrow \boldsymbol{r}_i - \sum_{k \neq k^*} \boldsymbol{r}_i^{(k)}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\epsilon_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \epsilon_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\boldsymbol{\mathcal{S}}^{\mathcal{A}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}SM}}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (\boldsymbol{S}, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon_S}, \boldsymbol{\epsilon_c}, \boldsymbol{\epsilon_{\hat{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}^{(k^*-1)}, \boldsymbol{\epsilon_c}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_S}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{EG}}^{\mathsf{th}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_c}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}_{\mathsf{Pa}}^{\mathsf{th}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}', \boldsymbol{\epsilon_c}', \boldsymbol{\epsilon}_{\mathbf{r}}'$

E: $(\boldsymbol{b}_{\boldsymbol{S}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{S}}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon_S}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon_c}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b_S}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}}'^{(k)}, \boldsymbol{b}_{\boldsymbol{r}}'^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{\boldsymbol{S}}' \leftarrow (\boldsymbol{S}_{\pi(j)} g_1^{\boldsymbol{b_S}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b_S}_j^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b_c}_j^{(k^*)} + q \boldsymbol{\chi_c}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{c}j})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b_r}_j^{(k^*)} + q \boldsymbol{\chi_r}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{r}j})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\boldsymbol{\mathcal{S}}^{\mathcal{A}}_{\mathbf{E}_{\mathbf{EG}}^{\mathbf{th}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{S}}', \tilde{\boldsymbol{S}}')$

$\boldsymbol{\mathcal{S}}^{\mathcal{A}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{c}}''); \boldsymbol{\mathcal{S}}^{\mathcal{A}}_{\mathbf{E}_{\mathbf{Pa}}^{\mathbf{th}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: **for** $j \in J$**:**

if $\boldsymbol{v}'_j \in \{v_0, v_1\}$**:**

E: $\boxed{\mathfrak{z}_1^{(k^*)} \xleftarrow{\$} \mathbb{G}_T}$

$E \leftrightarrow \mathcal{A}$: $\mathfrak{z}_1^{(k^*)}, (\mathfrak{z}_1^{(k)})_{k \neq k^*}$

E: $\mathfrak{z}_1 \leftarrow \prod_{k \in [m]} \mathfrak{z}_1^{(k)}; \mathfrak{z}_2 \leftarrow e(\tilde{\boldsymbol{S}}'_j, y f_2^{\tilde{\boldsymbol{c}}'_j}) / e(f_1 g_1^{\boldsymbol{v}'_j} h_1^{\tilde{\boldsymbol{r}}'_j}, f_2)$

$\mathfrak{g}_1 \leftarrow e(\tilde{\boldsymbol{S}}'_j, f_2); \mathfrak{g}_2 \leftarrow e(g_1, y f_2^{\tilde{\boldsymbol{c}}'_j})$

Get $(\delta_0^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape

$(\delta_1^{(k)}, \delta_2^{(k)})_{k \neq k^*} \leftarrow (\mathsf{Mult}(\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)}), \mathsf{Mult}(\delta_0^{(k)}, \boldsymbol{b_c}_j^{(k)}))_{k \neq k^*}$

if $i_0 \in I$: // simulate the entire DPK

$z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}, \mathrm{c} \xleftarrow{\$} \mathbb{Z}_q$

$a_{\mathfrak{z}_1}^{(k^*)} \leftarrow \mathfrak{h}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_0}^{(k^*)}} (\mathfrak{z}_1 / \prod_{k \neq k^*} \mathfrak{h}_2^{\boldsymbol{b_S}_j^{(k)}} \mathfrak{h}_3^{\delta_0^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}'_1}^{(k^*)} \leftarrow \mathfrak{z}_1^{-z_{bc}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_2}^{(k^*)}} (1_{\mathbb{G}_T} / \prod_{k \neq k^*} \mathfrak{z}_1^{-\boldsymbol{b_c}_j^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}} \mathfrak{h}_3^{\delta_2^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}_2}^{(k^*)} \leftarrow \mathfrak{g}_1^{z_{bc}^{(k^*)}} \mathfrak{g}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_1^{z_{br}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} (\mathfrak{z}_2 / \prod_{k \neq k^*} \mathfrak{g}_1^{\boldsymbol{b_c}_j^{(k)}} \mathfrak{g}_2^{\boldsymbol{b_S}_j^{(k)}} \mathfrak{h}_1^{\boldsymbol{b_r}_j^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}})^{\mathrm{c}}$

$E \leftrightarrow \mathcal{A}$: $a_{\mathfrak{z}_1}^{(k^*)}, a_{\mathfrak{z}'_1}^{(k^*)}, a_{\mathfrak{z}_2}^{(k^*)}; (a_{\mathfrak{z}_1}^{(k)}, a_{\mathfrak{z}'_1}^{(k)}, a_{\mathfrak{z}_2}^{(k)})_{k \neq k^*}$

Set $H(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2, \prod_{k \in [m]} a_{\mathfrak{z}_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}'_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}_2}^{(k)}) = \mathrm{c}$

$E \rightarrow \mathcal{A}$: $z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}$

else: // simulate the first two components of the DPK

$z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}, r_{bS}^{(k^*)}, r_{bc}^{(k^*)}, r_{br}^{(k^*)}, r_{\delta_1}^{(k^*)}, \mathrm{c} \xleftarrow{\$} \mathbb{Z}_q$

$a_{\mathfrak{z}_1}^{(k^*)} \leftarrow \mathfrak{h}_2^{z_{bS}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_0}^{(k^*)}} (\mathfrak{z}_1 / \prod_{k \neq k^*} \mathfrak{h}_2^{\boldsymbol{b_S}_j^{(k)}} \mathfrak{h}_3^{\delta_0^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}'_1}^{(k^*)} \leftarrow \mathfrak{z}_1^{-z_{bc}^{(k^*)}} \mathfrak{h}_2^{z_{\delta_1}^{(k^*)}} \mathfrak{h}_3^{z_{\delta_2}^{(k^*)}} (1_{\mathbb{G}_T} / \prod_{k \neq k^*} \mathfrak{z}_1^{-\boldsymbol{b_c}_j^{(k)}} \mathfrak{h}_2^{\delta_1^{(k)}} \mathfrak{h}_3^{\delta_2^{(k)}})^{\mathrm{c}}$

$a_{\mathfrak{z}_2}^{(k^*)} \leftarrow \mathfrak{g}_1^{r_{bc}^{(k^*)}} \mathfrak{g}_2^{r_{bS}^{(k^*)}} \mathfrak{h}_1^{r_{br}^{(k^*)}} \mathfrak{h}_2^{r_{\delta_1}^{(k^*)}}$

$E \leftrightarrow \mathcal{A}$: $a_{\mathfrak{z}_1}^{(k^*)}, a_{\mathfrak{z}'_1}^{(k^*)}, a_{\mathfrak{z}_2}^{(k^*)}; (a_{\mathfrak{z}_1}^{(k)}, a_{\mathfrak{z}'_1}^{(k)}, a_{\mathfrak{z}_2}^{(k)})_{k \neq k^*}$

Set $H(\mathfrak{g}_1, \mathfrak{g}_2, \mathfrak{h}_1, \mathfrak{h}_2, \mathfrak{h}_3, \mathfrak{z}_1, \mathfrak{z}_2, \prod_{k \in [m]} a_{\mathfrak{z}_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}'_1}^{(k)}, \prod_{k \in [m]} a_{\mathfrak{z}_2}^{(k)}) = \mathrm{c}$

$\boldsymbol{b_r}_j^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q; z_{br}^{(k^*)} \leftarrow z_{br}^{(k^*)} - \mathrm{c}\boldsymbol{b_r}_j^{(k^*)}$

$E \rightarrow \mathcal{A}$: $z_{bS}^{(k^*)}, z_{bc}^{(k^*)}, z_{br}^{(k^*)}, z_{\delta_0}^{(k^*)}, z_{\delta_1}^{(k^*)}, z_{\delta_2}^{(k^*)}$

else:

$\dots$ // proceed as $\mathcal{M}_{k^*}$

E: **endfor**

Figure 28: E_{15}: Use random group element for $\mathfrak{z}_1^{(k^*)}$.

$E_{16}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^{\boldsymbol{v}_i} h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boxed{\boldsymbol{v}_i^{(k^*)} \leftarrow 0; \boldsymbol{r}_i^{(k^*)} \leftarrow 0}$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \cdots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}SM}}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S', \boldsymbol{\epsilon}_{\boldsymbol{c}}', \boldsymbol{\epsilon}_{\mathbf{r}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b}_S^{(k)}, \boldsymbol{b}'^{(k)}_{\boldsymbol{c}}, \boldsymbol{b}'^{(k)}_{\boldsymbol{r}})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_{S_j}^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{c}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{c}_j})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{r}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{r}_j})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \bmod \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \bmod \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: $\boxed{\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}RSM}}(y, \boldsymbol{v}', (\tilde{S}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}'),}$

$\boxed{((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)})_{k \neq k^*})_{j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}},}$

$\boxed{((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)})_{k \in [m]})_{j \notin \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}})}$

Figure 29: E_{16}**: Set** $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ **as zeros (also abstracted out the DPK simulation steps during** DB-RSM**).**

$E_{17}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad \boxed{\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^0 h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)}$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad (\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad \boldsymbol{v}_i^{(k^*)} \leftarrow 0; \boldsymbol{r}_i^{(k^*)} \leftarrow 0$

$\quad (\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$\quad (\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \dots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \dots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

$\quad$ Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\quad \tilde{\boldsymbol{\sigma}} \leftarrow (\boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)})_{i \in [n]}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}SM}}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$\quad ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}^{(k^*-1)}, \boldsymbol{\epsilon_c}^{(k^*-1)}, \boldsymbol{\epsilon_r}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_S}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_c}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_r}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_S}', \boldsymbol{\epsilon_c}', \boldsymbol{\epsilon_r}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_c^{(k^*)}, \boldsymbol{b}_r^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\quad \boldsymbol{\chi}_c^{(k^*)}, \boldsymbol{\chi}_r^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\quad \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bS}}^{(k)}, \boldsymbol{\epsilon}_c' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{bc}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{br}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b_S}^{(k)}, \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\quad \tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b_S}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b_S}_j^{(k)}})_{j \in [n]}$ // threshold decryption

$\quad \tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b_c}_j^{(k^*)} + q \boldsymbol{\chi_c}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{c}}{}_j^{(k)})_{j \in [n]}$

$\quad \tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b_r}_j^{(k^*)} + q \boldsymbol{\chi_r}_j^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{r}}{}_j^{(k)})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')$

$\quad \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_c', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}RSM}}(y, \boldsymbol{v}', (\tilde{S}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}'),$
$\quad ((\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)}, \boldsymbol{b_r}_j^{(k)})_{k \neq k^*})_{j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}},$
$\quad ((\boldsymbol{b_S}_j^{(k)}, \boldsymbol{b_c}_j^{(k)}, \boldsymbol{b_r}_j^{(k)})_{k \in [m]})_{j \notin \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}})$

Figure 30: E_{17}**: Set** $\boldsymbol{\gamma}_{i_0}, \boldsymbol{\gamma}_{i_1}$ **as commitments of zeros.**

$E_{18}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^0 h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$(\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\boldsymbol{v}_i^{(k^*)} \leftarrow 0; \boldsymbol{r}_i^{(k^*)} \leftarrow 0$

$(\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$(\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \cdots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (\boldsymbol{v}_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon_\sigma}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon_\sigma}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

$\boxed{\textbf{for } i \in \{i_0, i_1\}\textbf{: } \tilde{\boldsymbol{\sigma}}_i \xleftarrow{\$} \mathbb{G}_1}$

$\boxed{\textbf{for } i \in [n] \setminus \{i_0, i_1\}\textbf{: } \tilde{\boldsymbol{\sigma}}_i \leftarrow \boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)}}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}SM}}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathrm{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S', \boldsymbol{\epsilon}_{\boldsymbol{c}}', \boldsymbol{\epsilon}_{\mathbf{r}}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}') \leftarrow (\boldsymbol{\epsilon}_S' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}c}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}r}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b}_S^{(k)}, \boldsymbol{b}_c'^{(k)}, \boldsymbol{b}_r'^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\tilde{S}' \leftarrow (S_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_{S_j}^{(k)}})_{j \in [n]}$ // threshold decryption

$\tilde{\boldsymbol{c}}'' \leftarrow (\boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{c}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{c}_j})_{j \in [n]}$

$\tilde{\boldsymbol{r}}'' \leftarrow (\hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{r}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'^{(k)}_{\boldsymbol{r}_j})_{j \in [n]}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S', \tilde{S}')$

$\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \bmod \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \bmod \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}RSM}}(y, \boldsymbol{v}', (\tilde{S}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}'),$
$((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)})_{k \neq k^*})_{j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}},$
$((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{c_j}^{(k)}, \boldsymbol{b}_{r_j}^{(k)})_{k \in [m]})_{j \notin \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}})$

Figure 31: E_{18}: Replace $\tilde{\sigma}_{i_0}, \tilde{\sigma}_{i_1}$ by elements randomly drawn from $\mathbb{G}_1$.

$E_{19}(1^\lambda, k^*, i_0, i_1, b)$:

// ***Keygen:***

E: $\mathsf{pk}^{(k^*)}, \mathsf{sk}^{(k^*)} \leftarrow \mathsf{E.Keygen}(1^\lambda)$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{EG}}, \mathsf{sk}_{\mathsf{EG}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \leftrightarrow \mathcal{A}$: $\mathsf{pk}_{\mathsf{Pa}}, \mathsf{sk}_{\mathsf{Pa}}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Keygen}(1^\lambda, (\mathcal{M}_k^{\mathcal{A}})_{k \neq k^*})$

$E \rightarrow \mathcal{A}$: $\mathsf{pk}^{(k^*)}, \mathsf{pk}_{\mathsf{EG}}, \mathsf{pk}_{\mathsf{Pa}}$

$\mathcal{A} \rightarrow E$: $(\mathsf{pk}^{(k)})_{k \neq k^*}$

$\mathcal{A} \rightarrow E$: $v_0, v_1, (\boldsymbol{c}_i)_{i \in [n] \setminus \{i_0, i_1\}}$

E: **if** $(b = 0)$ **then** $(v_{i_0}, v_{i_1}) := (v_0, v_1)$ **else** $(v_{i_0}, v_{i_1}) := (v_1, v_0)$

// ***Enc calls for*** i_0, i_1***:***

E: **for** $i \in \{i_0, i_1\}$**:**

$\quad \boldsymbol{\epsilon}_i \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad \boldsymbol{r}_i \xleftarrow{\$} \mathbb{Z}_q; \boldsymbol{\gamma}_i \leftarrow g_1^0 h_1^{\boldsymbol{r}_i}; \boldsymbol{\rho}_{\boldsymbol{\gamma}_i} \leftarrow \mathcal{S}^H_{\mathsf{NIZKPK}}(\boldsymbol{\gamma}_i)$

$\quad \boldsymbol{\epsilon}_{r_i} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$\quad (\boldsymbol{v}_i^{(k)})_{k \neq k^*}, \boldsymbol{r}_i^{(k)})_{k \neq k^*} \xleftarrow{\$} \mathbb{Z}_q$

$\quad \boldsymbol{v}_i^{(k^*)} \leftarrow 0; \boldsymbol{r}_i^{(k^*)} \leftarrow 0$

$\quad (\mathbf{ev}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{v}_i^{(k)}))_{k \neq k^*}; \mathbf{ev}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$\quad (\mathbf{er}_i^{(k)})_{k \neq k^*} \leftarrow (\mathsf{E.Enc}(\mathsf{pk}^{(k)}, \boldsymbol{r}_i^{(k)}))_{k \neq k^*}; \mathbf{er}_i^{(k^*)} \leftarrow \mathsf{E.Enc}(\mathsf{pk}^{(k)}, 0)$

$E \rightarrow \mathcal{A}$: $\quad \boldsymbol{c}_i := (\boldsymbol{\epsilon}_i, \boldsymbol{\gamma}_i, (\mathbf{ev}_i^{(k)}, \mathbf{er}_i^{(k)})_{k \in [m]}, \boldsymbol{\rho}_{\boldsymbol{\gamma}_i}, \boldsymbol{\epsilon}_{r_i})$

endfor

// ***Mix:***

E: $\pi^{(k^*)} \xleftarrow{\$} \mathsf{Perm}(n)$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}^{(k^*-1)} := (\boldsymbol{\epsilon}_i^{(k^*-1)})_{i \in [n]}$

E: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow \mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0)$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}^{(k^*)} \leftarrow (\boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(1)}, \ldots, \boldsymbol{\epsilon}^{(k^*)}_{\pi^{(k^*)}(n)})$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}'$

E: Get $(\pi^{(k)})_{k \neq k^*}$ using $\mathcal{A}$'s random tape; $\pi \leftarrow \pi^{(m)} \circ \cdots \circ \pi^{(1)}$

E: **for** $j \in [n]$**:** $\boldsymbol{v}'_j \leftarrow (v_{\pi(j)})_{j \in [n]}$

$\mathcal{A} \leftrightarrow E$: **for** $j \in [n]$**:** $\mathcal{S}^{\mathcal{A}}_{\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \boldsymbol{\epsilon}'_j, \boldsymbol{v}'_j)$

Get $\boldsymbol{v}_{i_0}^{(k^*)}, \boldsymbol{r}_{i_0}^{(k^*)}, \boldsymbol{v}_{i_1}^{(k^*)}, \boldsymbol{r}_{i_1}^{(k^*)}$ from the Enc calls for i_0, i_1

for $i \in [n] \setminus \{i_0, i_1\}$**:**

$\quad$ Get $\boldsymbol{v}_i^{(k^*)}, \boldsymbol{r}_i^{(k^*)}$ from $\mathcal{A}$'s random tape and input for sender S_i

// ***OTraceIn call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceIn}(I, J)$

E: **assert** $v_0 \in \boldsymbol{v}'_J \iff v_1 \in \boldsymbol{v}'_J$

$\mathcal{A} \rightarrow E$: $y, \boldsymbol{\sigma}', \boldsymbol{\epsilon}'_\sigma$ // DB-SM call (repeat for $[n] \setminus J$)

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*+1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{\sigma}}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{j \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_\sigma$

E: $\boldsymbol{b}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_q^n$ // homomorphic blinding

$E \rightarrow \mathcal{A}$: $\tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} \leftarrow \mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}})$

$\mathcal{A} \rightarrow E$: $(\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: $\tilde{\boldsymbol{\epsilon}}_\sigma \leftarrow \tilde{\boldsymbol{\epsilon}}_\sigma^{(k^*)} (\tilde{\boldsymbol{\epsilon}}_\sigma^{(k)})_{k \neq k^*}$

E: Get $(\boldsymbol{b}_i^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape. // threshold decryption

for $i \in \{i_0, i_1\}$**:** $\tilde{\boldsymbol{\sigma}}_i \xleftarrow{\$} \mathbb{G}_1$

for $i \in [n] \setminus \{i_0, i_1\}$**:** $\tilde{\boldsymbol{\sigma}}_i \leftarrow \boldsymbol{\sigma}'^{\,\boldsymbol{b}_i^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_i^{(k)}}_{\pi^{-1}(i)}$

$\mathcal{A} \leftrightarrow E$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_\sigma, \tilde{\boldsymbol{\sigma}})$

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}SM}}(y, \boldsymbol{\gamma}, \tilde{\boldsymbol{\sigma}}, ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \neq k^*})_{i \in \{i_0, i_1\}},$
$\quad ((\boldsymbol{v}_i^{(k)}, \boldsymbol{r}_i^{(k)}, \boldsymbol{b}_i^{(k)})_{k \in [m]})_{i \in [n] \setminus \{i_0, i_1\}})$

// ***OTraceOut call:***

$\mathcal{A} \rightarrow E$: $\mathsf{OTraceOut}(I, J)$

E: **assert** $c_{i_0} \in \boldsymbol{c}_I \iff c_{i_1} \in \boldsymbol{c}_I$

$\mathcal{A} \rightarrow E$: $y, (S, \boldsymbol{c}, \hat{\boldsymbol{r}}), (\boldsymbol{\epsilon}_S, \boldsymbol{\epsilon}_{\boldsymbol{c}}, \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}})$ // DB-RSM call (repeat for $[n] \setminus I$)

E: $\boldsymbol{\epsilon}_{\mathbf{r}} \leftarrow \boldsymbol{\epsilon}_{\hat{\boldsymbol{r}}} \boldsymbol{\epsilon}_{\boldsymbol{r}}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*-1)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*-1)}$ // shuffling

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_S{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{EG}}.\mathsf{REnc}(\mathsf{pk}_{\mathsf{EG}}, g_1^0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\boldsymbol{c}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$E \rightarrow \mathcal{A}$: $\boldsymbol{\epsilon}_{\mathbf{r}}{}^{(k^*)} \leftarrow (\mathsf{E}^{\mathsf{th}}_{\mathsf{Pa}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, 0))_{i \in [n]}$

$\mathcal{A} \rightarrow E$: $\boldsymbol{\epsilon}_S{}', \boldsymbol{\epsilon}_{\boldsymbol{c}}{}', \boldsymbol{\epsilon}_{\mathbf{r}}{}'$

E: $(\boldsymbol{b}_S^{(k^*)}, \boldsymbol{b}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{b}_{\boldsymbol{r}}^{(k^*)}) \xleftarrow{\$} (\mathbb{Z}_q^n \times \mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ // homomorphic blinding

$\quad \boldsymbol{\chi}_{\boldsymbol{c}}^{(k^*)}, \boldsymbol{\chi}_{\boldsymbol{r}}^{(k^*)} \xleftarrow{\$} \mathbb{Z}_{q-1}^n$

$E \rightarrow \mathcal{A}$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k^*)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k^*)}) \leftarrow (\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{EG}}, \boldsymbol{g}_1^{\mathbf{0}}),$
$\quad \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}), \mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{Enc}(\mathsf{pk}_{\mathsf{Pa}}, \mathbf{0}))$

$\mathcal{A} \rightarrow E$: $(\boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k)})_{k \neq k^*}$

E: $(\tilde{\boldsymbol{\epsilon}}_S{}', \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}') \leftarrow (\boldsymbol{\epsilon}_S{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}S}^{(k)}, \boldsymbol{\epsilon}_{\boldsymbol{c}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{c}}^{(k)}, \boldsymbol{\epsilon}_{\mathbf{r}}{}' \prod_{k \in [m]} \boldsymbol{\epsilon}_{\boldsymbol{b}\boldsymbol{r}}^{(k)})$

E: Get $(\boldsymbol{r}_i)_{i \in [n] \setminus \{i_0, i_1\}}, (\boldsymbol{b}_S{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}, \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)})_{k \neq k^*}$ from $\mathcal{A}$'s random tape.

$\boxed{\textbf{for } j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}\textbf{:}}$ // threshold decryption

$\quad \boxed{\tilde{S}'_j \xleftarrow{\$} \mathbb{G}_1; \tilde{\boldsymbol{c}}'''_j \xleftarrow{\$} \mathbb{Z}_{q+q^2}; \tilde{\boldsymbol{r}}'''_j \xleftarrow{\$} \mathbb{Z}_{2q+q^2}}$

$\quad \boxed{\tilde{\boldsymbol{c}}''_j \leftarrow \tilde{\boldsymbol{c}}'''_j + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}_j; \tilde{\boldsymbol{r}}''_j \leftarrow \tilde{\boldsymbol{r}}'''_j + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)}_j}$

$\boxed{\textbf{for } j \in [n] \setminus \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}\textbf{:}}$

$\quad \boxed{\tilde{S}'_j \leftarrow S_{\pi(j)} g_1^{\boldsymbol{b}_{S_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}_{S_j}^{(k)}}}$

$\quad \boxed{\tilde{\boldsymbol{c}}''_j \leftarrow \boldsymbol{c}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{c}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{c}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{c}}{}^{(k)}_j}$

$\quad \boxed{\tilde{\boldsymbol{r}}''_j \leftarrow \hat{\boldsymbol{r}}_{\pi(j)} + \boldsymbol{r}_{\pi(j)} + \boldsymbol{b}_{\boldsymbol{r}_j}^{(k^*)} + q \boldsymbol{\chi}_{\boldsymbol{r}_j}^{(k^*)} + \sum_{k \neq k^*} \boldsymbol{b}'_{\boldsymbol{r}}{}^{(k)}_j}$

$E \leftrightarrow \mathcal{A}$: $\mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{EG}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{EG}}, \tilde{\boldsymbol{\epsilon}}_S{}', \tilde{S}')$
$\quad \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\boldsymbol{c}}{}', \tilde{\boldsymbol{c}}''); \mathcal{S}^{\mathcal{A}}_{\mathbf{E}^{\mathbf{th}}_{\mathbf{Pa}}.\mathbf{TDec}}(\mathsf{pk}_{\mathsf{Pa}}, \tilde{\boldsymbol{\epsilon}}_{\mathbf{r}}{}', \tilde{\boldsymbol{r}}'')$

E: $\tilde{\boldsymbol{c}}' \leftarrow \tilde{\boldsymbol{c}}'' \,\mathbf{mod}\, \boldsymbol{q}; \tilde{\boldsymbol{r}}' \leftarrow \tilde{\boldsymbol{r}}'' \,\mathbf{mod}\, \boldsymbol{q}$

E: $\mathfrak{h}_1 \leftarrow e(h_1, f_2)^{-1}; \mathfrak{h}_2 \leftarrow e(g_1, f_2)^{-1}; \mathfrak{h}_3 \leftarrow f_T$ // stage 2

E: $\mathcal{S}^H_{\mathsf{DPK\text{-}DB\text{-}RSM}}(y, \boldsymbol{v}', (\tilde{S}', \tilde{\boldsymbol{c}}', \tilde{\boldsymbol{r}}'),$
$\quad ((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)})_{k \neq k^*})_{j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}},$
$\quad ((\boldsymbol{b}_{S_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{c}_j}^{(k)}, \boldsymbol{b}_{\boldsymbol{r}_j}^{(k)})_{k \in [m]})_{j \notin \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}})$

Figure 32: E_{19}**: Replace** $\tilde{S}'_j, \tilde{\boldsymbol{c}}''_j, \tilde{\boldsymbol{r}}''_j$ **for** $j \in \{\pi^{-1}(i_0), \pi^{-1}(i_1)\}$ **by randomly drawn elements.**