Synq: Public Policy Analytics Over Encrypted Data

Zachary Espiritu*§, Marilyn George*§, Seny Kamara*†, Lucy Qin†
*MongoDB Research †Brown University
{zachary.espiritu, marilyn.george, seny.kamara}@mongodb.com lq@brown.edu

***Abstract*—Data analytics is a core part of modern decision making, especially in public policy. However, there exists a tension between data privacy and otherwise socially beneficial analytics when data sources contain personal information. We design Synq, a system that supports analytics over encrypted data while accounting for the usability considerations institutions may have when conducting studies that affect public policy. We specifically use an *application-centric* approach and model Synq's design requirements from a large-scale series of studies conducted on the opioid epidemic in Massachusetts. We systematize the design considerations of the public policy context and demonstrate how the combination of design considerations that Synq addresses is novel through a survey of the literature. We then present our protocol which combines structured encryption, somewhat homomorphic encryption, and oblivious pseudorandom functions to support a complex query language that includes filtering (retrieving rows by attribute/value pairs), linking (merging rows from different tables that represent the same individual) and aggregate functions (sum, count, average, variance, regression). We formally express the security of our protocol and show that Synq is efficient in practice while satisfying usability considerations that are critical to deployment in the setting of public policy studies.**

1. Introduction

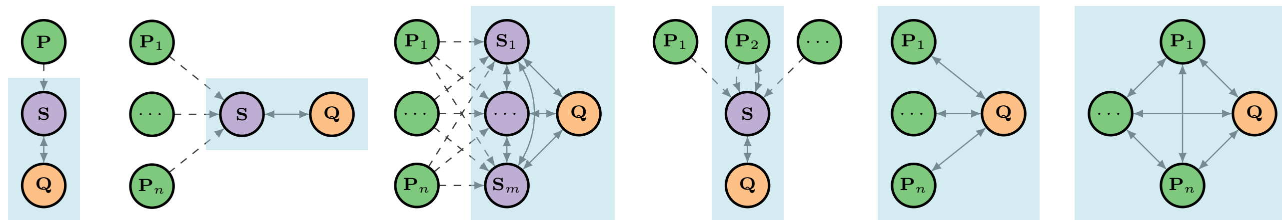
Data analysis is a core part of decision making in almost every facet of society. It is well understood that more data—specifically, relevant data from *different* sources—leads to more robust and beneficial insights [39], [41], [84]. This understanding explains the prevalence of collaborative, public policy studies where an *analyst* aggregates data from multiple *data owners* (e.g., non-profits and government organizations) in a larger dataset to enable more powerful analyses [7]. Some studies go further by *linking* data about the same person from different datasets by matching personal identifiable information (PII) for even more comprehensive analysis. This is commonly implemented via a centralized, *trusted* server that aggregates datasets and makes them available to analysts (e.g., [72]). However, PII and other sensitive information (e.g., healthcare diagnoses or visits) are subject to legal safeguards (e.g., HIPAA). Data owners must navigate institutional processes and legal approvals to make such analyses tenable, and these challenges only grow with the number of involved parties [85], [89].

§. Work conducted in part at Brown University.

One way to make these studies possible is to design systems that support analytics over encrypted datasets. Several existing systems towards this goal are designed as general-purpose systems (e.g., [12], [77], [87]). Other systems (e.g., [27], [83]) prioritize *application-centric* design, where a system is designed for the needs and challenges of a specific application setting. In this work, we take an application-centric approach to our system design. Our work was initiated by discussions with the Policy Lab at Brown University about balancing privacy concerns with the need for data-driven insights in policymaking [3]. They highlighted a specific public health initiative as a model of the constraints and functionalities to support when designing a system for privacy-preserving analytics used by public institutions.

The MA DPH initiative. Our work focuses on the Chapter 55 initiative conducted by the Massachusetts Department of Public Health (MA DPH), which produced 23 studies using data from multiple government agencies and public health institutions to better understand the scale of the opioid epidemic in Massachusetts, its underlying causes, and successful interventions for addressing it [72]. Throughout the paper, we provide a *running example* of how the MA DPH initiative informed our research and how our system can implement the multiple studies involved.

Deployment topology. As part of our application-centric approach, we first identified a set of design requirements. In doing so, we discovered a gap in the literature—namely, that the *deployment topology* and the design considerations required by our specific context had not been identified and appropiately addressed by existing systems. A deployment topology is a configuration of participating parties and the data flow between them. Figure 1 presents a taxonomy of commonly used deployment topologies. For example, (T0) is a simple topology where a data owner uploads data to a server and an analyst later queries the server. Many prior encrypted database systems (e.g., [10], [56], [59], [78]) follow this topology. At the other extreme, the synchronous compute topology (T5) requires all parties to synchronously interact throughout query execution. Other topologies found in prior work include: the analyst communicates synchronously with a group of servers that hold (outsourced) datasets previously uploaded by data owners (T2); the analyst communicates synchronously with one selected data owner (T3); and the analyst communicates synchronously with all the data owners (T4). With the exception of (T1) and (T2), all the multi-owner topologies require that data owners be online during query computation.



(T0) Single-writer. (T1) Centralized compute. (T2) Distributed compute. (T3) Selected owner. (T4) Iterative compute. (T5) Synchronous compute.

Figure 1. A taxonomy of deployment topologies. P nodes represent data owners who contribute data. S represent compute servers. Q represents an analyst. Highlighted areas and solid edges denote parts of the topology involved in synchronous computation with Q for queries. Increasing topology numbers roughly correspond to increasing synchronization requirements between the parties (e.g., (T1) has the least synchronization; (T5) has the most).

Topology of MA DPH. For the MA DPH initiative, participating data owners contributed 22 datasets with a diverse set of attributes about individuals in Massachusetts. Datasets were hand-delivered to the DPH on encrypted hard drives which were then semi-manually linked together in plaintext. Finally, records were de-identified and made available to approved analysts. This *centralized compute* topology (T1) is particularly important in the public policy context, due to a wide variety of reasons that we will examine in Section 2.

1.1. Our Contributions

We present Synq, a system which enables complex multi-dataset analyses over encrypted data, accounts for real-world constraints faced by public policy organizations, and guarantees data privacy even when all-but-one of the data owners collude with the server. While Synq was designed around the MA DPH context, we demonstrate the potential broader impact of our design by showing how Synq can also enable a privacy-preserving wage equity study [69] outside of the MA DPH initiative in Appendix C.

Design considerations. We use application-centric design to understand the usability and expressivity needs of multi-dataset analytics that Synq must satisfy. While we use the MA DPH setting as our primary example, we show that the application-centric approach surfaces concerns relevant for public policy studies at large that are not fully addressed by prior work. We define our design considerations and how they relate to prior work in Sections 2 and 3.

Query language. A significant challenge in designing a system for encrypted analytics is determining the appropriate level of query expressivity. In Section 5, we propose Synq-QL which supports common operations required by public policy studies. Synq-QL is based on a survey of the queries performed in the MA DPH initiative and supports filters, linking records across owners, and aggregations.

Protocol design. We propose a new protocol based on structured encryption (STE), somewhat homomorphic encryption (SHE), and oblivious pseudorandom functions (OPRFs). We provide a formal leakage analysis, and prove that our protocol reveals at most this leakage when the server and all-but-one of the data owners are corrupted by a semi-honest adversary. Any adversary who gains access to a copy of the encrypted data structures will only learn cumulative statistics (e.g. the total number of records in each dataset). Therefore, Synq provides better protection against external breaches compared to both plaintext systems and systems based on property-preserving encryption (PPE). Further, since Synq makes black-box use of STE schemes, we can leverage future advances (such as leakage suppression [47], [57]) to improve its security and efficiency.

Improved linking. Most prior work that supports record linking uses either a plaintext linking phase, a trusted third party (TTP), or deterministic encryption (DTE). Plaintext linking or the use of a TTP (such as in the MA DPH initiative) requires a significant trust assumption, and DTE reveals all the links in the data to the server at setup time. Synq uses a linking protocol that only reveals a subset of the links to the server at *query time*. We express this leakage precisely and show that it remains unchanged even when all-but-one of the data owners collude with the server.

Evaluation. We implement Synq and perform an empirical evaluation that shows its real-world feasibility.

2. Design Considerations

Inspired by discussions with the Policy Lab at Brown University and the emerging literature on secure multiparty computation usability (e.g., [66], [67], [69], [90]), we developed design requirements for Synq aimed at supporting the technical and usability needs demonstrated by previous real-world public policy studies—in particular, the MA DPH initiative and a public policy study about wage equity in Boston [69]. In both studies, analysts were able to obtain insight on public policy questions using data from a wide array of institutions (i.e. government, industry, academia, nonprofits, hospitals) while preserving the privacy of the original datasets. We analyzed the documented requirements of these studies, their priorities, and their challenges to come up with the design considerations for Synq in Table 1.

Usability. The usability of a technical system is crucial to its real-world applicability. Data owners may have non-technical backgrounds [11], may make mistakes during computation [69], and, most critically from a protocol perspective, may not be able to participate in synchronous computation [51], [85]. For this reason, we identify several considerations which focus on enabling asynchronous participation by the data owners. We acknowledge that these

TABLE 1. Synq's design considerations, grouped by their focus on *usability* or *expressivity*, and the specific public policy concerns that motivate them.

	Requirement	Description	Rationale
Usability	[D1] **Ephemeral Keys**	Data owners should not have to share a symmetric key, or maintain a public-private keypair in order to participate.	• *Practical logistics:* The Boston study notes that coordinating a time when all (or even a subset) of the owners are available simultaneously is infeasible [67], [69]. • *Key maintenance:* Owners do not have capability to maintain state (e.g., a long-lived key) before or after setup (also relevant to general multi-writer DBs [90]).
	[D2] **Async Setup**	Data owners should be able to upload their data without synchronous setup with other owners.	• *Practical logistics:* See above. • *Resource constraints:* Impossible for owners to remain online throughout the execution of the entire analysis process [67], [69]; involving any subset of owners as part of query computation requires unreasonable overhead [51], [69], [85].
	[D3] **Retry Setup**	Data owners should be able to retry setup easily and without synchronizing with the other data owners.	• *Correcting mistakes:* Asynchronous retries were necessary in the Boston study to correct mistakes; without this feature, every mistake would lead to a restart of the computation and erode trust in the system [67], [69].
	[D4] **Offline at Query**	Data owners should be able to go offline after setup and not be online during analyst queries.	• *Practical logistics:* See above. • *Resource constraints:* See above.
Expressivity	[D5] **Multiple Schemas**	Synq should support multiple schemas for input datasets.	• *Diverse data sources:* Participating MA DPH initiative data owners have their own schema (see Appendix A of [72]); not easily unified due to different contexts.
	[D6] **Multi-Col Linking**	Synq should have support for linking records across datasets using the values of multiple columns.	• *Linking individuals:* MA DPH initiative emphasizes importance of linking records across datasets based on PII with ways to adjust the linking granularity as a crucial mechanism for analyzing the activity of an individual across datasets [72].
	[D7] **Aggregate Functions**	Synq should support counts, sums, averages, and regressions.	• *Public policy studies:* All analyses in MA DPH initiative required an aggregate operation to be supported over the underlying datasets [72].

requirements are not traditionally considered "usability" concerns. However, prior work demonstrates that adherence to these requirements significantly impacts whether or not owners can feasibly participate in a policy study implementation [69], [85] and so we identify them as such.

Expressivity. Policy studies conducted over plaintext datasets can include very expressive queries. As part of our design process, we determined the appropriate expressivity for Synq by using the MA DPH report [72] to (1) construct an approximate representation of the schema used by each of the participating data owners; (2) list all the analysis questions mentioned in the report; and (3) identify the required operations that we would have to support to enable the study over encrypted datasets. Due to space restrictions, we defer this query survey to the full version of our paper.

Computational resources. We expect Synq's users to have varying resources and technical expertise. The MA DPH initiative included a diverse set of public agencies and the Boston study included 100+ institutions with disparities in technical support. Since institutions have varied resources, we ensure that *no specialized hardware* is required and that owners and analysts can participate asynchronously using *consumer-level* desktop computers or laptops.

3. Prior Work

We now examine prior work in multi-owner analytics over encrypted data with respect to our design considerations from Section 2. To our knowledge, no prior work in the literature addresses the combination of concerns in Table 1. However, some of these systems were designed for specific applications while others were designed as general-purpose solutions. Naturally, those that were designed to be application-centric, may have a more narrow (or different) set of design criteria to meet the needs of their specific use cases. Additionally, Table 2 demonstrates that many prior works address different topologies than ours. While they may use similar cryptographic primitives or have similar goals, they were designed for a different setting.

We emphasize that many of the works we include in Table 2 serve as important precedent for convincing various governmental entities (e.g., [39], [41], [85]) of the real-world feasibility of using cryptographic approaches for studies that affect public policy. Synq therefore synthesizes and builds upon lessons learned from prior work to create a system that addresses usability considerations important to deploying privacy-preserving analytics in this setting. Therefore, we highlight these prior works to demonstrate the gap that Synq fills rather than as criticism of prior work.

Table 2 demonstrates several trends in prior work:

- *Expressivity vs. usability.* Almost all prior work offers some degree of expressivity, but only Web-MPC [69] and H-SE [90] explicitly address usability concerns.
- *Usability and deployment topologies.* Our topology taxonomy reveals usability trends in prior work. As the topology number increases, the required synchronization also increases. Table 1 demonstrates how this leads to decreased usability based on our design considerations.
- *Lack of linking support.* Some prior work supports linking, and those that do use multi-party computation (MPC), shared PRF keys, or property-preserving encryption (PPE). However, as described below, these techniques incur usability or security downsides in our setting.

The only prior work that explicitly emphasizes usability in the policy setting is the Boston study [69]. Since this work motivates many of our usability considerations, it

TABLE 2. Survey of prior systems for multi-owner analytics over encrypted data with respect to Synq's design requirements, grouped by whether the system design was presented as application-centric (**ACD**) or not (**NACD**). Topologies are indicated using the taxonomy from Figure 1 with deviations indicated by ∗. For the *Design Considerations* from Table 1, ● indicates support, ◐ indicates partial support, and ○ indicates no or inconclusive support. We roughly group the works by technique and, within groups, order each work by increasing topology number.

					Usability				Expressivity		
	System	**Year**	**Protocol Overview**	**Topology**	**Ephem Keys**	**Async Setup**	**Retry Setup**	**Offline at Query**	**Multiple Schemas**	**Multi-Col Linking**	**Aggreg Func**
ACD					[D1]	[D2]	[D3]	[D4]	[D5]	[D6]	[D7]
STE	Synq	2023	Our system designed around the MA DPH initiative [72].	(T1)	●	●	●	●	●	●	●
	Gun Reg [58]	2021	Querier locates a single database managed by a county and generates STE tokens with the owner via MPC to retrieve records from it.	(T3)	○	●	●	◐	●	○	○
Secret Sharing	Boston Web-MPC [67], [69]	2018	Server aggregates data masked with 1-time-use additive secret shares (or Shamir secret shares) encrypted with analyst public key. Analyst retrieves & unmasks to sum data.	(T1)	●	●	●	●	○	○	◐
	CARRIER [35]	2020	Estimating coronary artery disease in Netherlands with additive-SS-based dot product from [86]. Supports sums and 1-column links.	(T5)	●	○	○	○	○	◐	◐
HE	PDCi2b2 [80]	2018	Owners upload AHE data to server. Server key-switches server-public-key data to analyst-public-key data to respond to queries. Only sums; filters are in plaintext; uses standardized schema.	(T1)	●	●	○	●	○	○	◐
	MedCo [81]	2019	Owners send data to subset of owners & non-owner compute nodes (∗) who answer sum queries. Uses key-switching from [80], SKE-to-DTE switching protocol; uses standardized schema.	(T2) ∗	◐	●	○	◐	○	○	◐
	Kaptchuk et al. [60]	2017	Owners globally publish FHE-encrypted data; analysts download 1 dataset, compute locally, and send result to owner to decrypt.	(T3)	○	●	◐	○	○	○	●
	FAMHE [45]	2021	Biomedical analyst sends query to owners, who iteratively perform local computation and exchange HE ciphertexts to compute.	(T4)	○	●	○	○	○	○	●
Circuit MPC	VaultDB [83]	2022	Owners send secret shares to subset of owners who answer MPC queries (∗); linking via shared PRF key & heuristic anonymization from [33]. Only counts; owners manually standardize schema.	(T2) ∗	◐	○	○	◐	◐	●	◐
	Estonia Tax Study [18]	2016	Links education/tax data with garbled circuits & additive SS from [17]. Uses plaintext process to standardize schema.	(T2)	●	○	○	○	◐	●	●
Enclave	Princess [27]	2017	Intel-SGX-based aggregations of genome data under a standardized format. SGX server acts as analyst (∗); all parties learn output.	(T4) ∗	●	○	○	○	○	○	●
Private Set Intersect	NPSAS Pilot [11]	2021	Protocol for linking education records over known student IDs using garbled circuits, programmable PRFs, and cuckoo hashing. Supports averages and 1-col links.	(T5)	●	○	○	○	●	◐	◐
	PSI-HU / PSI-CI [34]	2021	Analytics (supports two specialized metrics) and linking via cuckoo hashing with a shared PRF key. Assumes existence of PKI.	(T5)	○	○	○	○	○	●	◐
	PI-Sum [52]	2020	Uses OPRF-based oblivious transfer, Bloom filters, and AHE to compute aggregate sums for ad conversions. Supports 1-col links.	(T5)	●	○	○	○	○	○	◐
NACD											
PKSE	H-SE [90]	2022	General multi-writer system based on public-key searchable-encryption (PKSE) & identity-based encryption.	(T1)	○	●	○	●	○	○	○
HE	UnLynx [44]	2017	Uses HE, zero-knowledge-proofs, and verifiable shuffles to enforce confidentiality / unlinkability between providers and data for sums. Owners must respond at query time; no setup phase.	(T4)	○	○	○	○	○	○	◐
Circuit MPC	Jana [12], [51]	2018	Uses PPE for certain queries; for others, owners upload data shares to servers which collectively use MPC. Does not support regressions. Supports joins on plaintext or DTE values.	(T2)	●	●	○	●	●	●	◐
	Senate [77]	2021	Owners collaboratively run SQL queries using MPC query planning; supports malicious model. No regression support.	(T5)	●	○	○	○	●	●	◐
	Conclave [87]	2019	Broker orchestrates hybrid MPC-based query plans over owner-controlled databases; based on [17] and Obliv-C. No regressions.	(T5)	●	○	○	○	●	●	◐
	SMCQL [14]	2017	Broker orchestrates MPC-based query plans over owner-controlled databases in semi-honest model.	(T4)	●	○	○	○	●	●	●
Secret Sharing	Logistic regression [6]	2022	Owners upload inputs to two non-colluding servers which use function secret sharing to compute logistic regressions. No specific analyst party (∗).	(T2) ∗	●	○	○	○	○	○	◐

(unsurprisingly) satisfies them. However, it was designed for a particular use case which only needed one schema along with one-time use of contributed data. We design for similar usability considerations while expanding expressivity.

3.1. Summary of Prior Techniques

General-purpose MPC-based. The seminal Estonia tax study [18] offers almost all the expressivity that we require except supporting multiple schemas. However, it required multiple restarts due to clients disconnecting during the computation. In general, while general-purpose MPC is a powerful tool, its synchronicity, performance, and lack of fault tolerance (e.g., client disconnects) present usability concerns in our setting.

MPC+PPE-based. Jana [51] is a system that supports multi-user analytics in one of two ways depending on the complexity of queries: by using PPE; or by using an outsourced MPC design (T2) (topology), which allows the data owners to remain offline during query computation. The protocol also necessitates the use of multiple compute servers which have to remain online during the duration of query. Jana uses PPE to support record linking and efficient queries (similar to MedCo [81], CryptDB [78], Seabed [8]). However, PPE is known to leak a significant amount of information even at setup time [73], so we avoid it in Synq. Jana uses outsourced MPC to handle more complex queries (similar to [77], [83], [87]). Of all the techniques used in prior work, outsourced MPC appears to be well-suited to our public policy setting because it addresses most of our usability considerations. However, if more than a threshold of the compute cluster is corrupted, the plaintext data is revealed to the adversary. On the other hand, in Synq, an adversary corrupting the server only learns small, well-defined leakage about the plaintext data and queries.

MPC+STE-based. The encrypted gun registry [58] uses both MPC and STE. Due to the use of STE, this system does not reveal any plaintext data to the server at setup time, and its leakage is well-defined. Similar to Synq, this system makes black-box use of STE schemes and can therefore leverage future advances in STE (e.g., [47], [57]). Although it does satisfy some of our usability requirements, the registry does not support aggregates or linking of records. The use of MPC also requires owners to be online during queries.

HE-based. Kaptchuk et. al. use fully homomorphic encryption (FHE) to support analytics on encrypted medical datasets [60]. Although their scheme supports regressions, a linear regression on 50 encrypted rows took approximately 9.5 hours. In comparison, Synq relies on SHE and takes less than 5 minutes to compute a linear regression on $100{,}000$ records. Other works use additively homomorphic encryption (AHE), such as PDCi2b2 [80] and MedCo [81]. Due to the use of AHE, these systems only support additive aggregates, and often rely on less secure mechanisms for filtering, such as PPE or even plaintext filtering.

Hardware-based. Princess [27] uses Intel SGX-based enclaves. While SGX allows for arbitrary query expressivity, the scheme does not generalize well to larger datasets due to SGX's memory limitations. Also, significant attacks have been discovered against SGX (e.g., [20], [30], [42], [68]).

Omitted work. For completeness, we note that the survey in Table 2 omits works on the following, which are orthogonal to our setting and do not match our requirements:

- *Federated machine learning* [19], [54], [88], [91], where parties collaboratively train (and potentially learn) a model without widely sharing their own dataset.
- *Private, repeated/streaming data aggregation* (e.g., Prio+ [5], Apple and Google's COVID-19 notification system [9], Flag [13], TimeCrypt, [21], Zeph [22], Prio [29], Indonesia tourism study [32], Elsa [82]), where many clients repeatedly contribute encrypted data to an aggregation server. MA DPH involves a much smaller number of data owners (in comparison to the number of clients usually considered in these works) who upload data once.
- *Single-writer encrypted databases* (e.g., Seabed [8], Cipherbase [10], ESPADA [46], OPX [59], Blind Seer [76], CryptDB [78]). Single-writer databases can technically be proxied for multi-writer use but proxying either requires a TTP or key sharing (which would violate [D1]). Further, simultaneously supporting setup retries [D3], multiple schemas [D5], and linking [D6] would be extremely difficult and require additional overhead.

4. Preliminaries

Tables. A data owner $\mathbf{P}_i$'s data is denoted as a table T_i. For readability, we assume that each data owner $\mathbf{P}_i$ has exactly one table T_i. X_i denotes the set of T_i's columns. A column can be numeric or non-numeric depending on the values it contains. We refer to a column using a column identifier, which may differ from the underlying column name in the dataset. Each data record $\mathbf{r}$ in a table has one value per column identifier x, denoted as $\mathbf{r}[x]$. $T_1 \bowtie T_2$ denotes a linked table, where each record $\mathbf{r}_1$ in T_1 is combined with a corresponding record $\mathbf{r}_2$ in T_2 if both records contain the same values for columns corresponding to some link condition. X denotes the set of all columns, X^{Filter} to denote the columns that the analyst can perform filters on, and X^{Num} to denote the numeric columns.

Dictionaries and multi-maps. A dictionary DX is a data structure that maps labels to values. Each unique label is mapped to one value, and a query for a label returns that corresponding value. A multi-map MM maps each label to a tuple containing multiple values. A query for a label then returns the label's entire tuple in the multi-map.

Public-Key Encryption. A public-key encryption scheme is a cryptographic primitive consisting of three polynomial–time algorithms $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, which together enable the encryption of a message m using a public key and the decryption of the corresponding ciphertext ct using a secret key. Gen takes a security parameter k as input and returns a key pair $(\mathsf{pk}, \mathsf{sk})$ where pk is the public key and sk is the secret key. Enc takes the public key pk and a message m as inputs and returns a ciphertext ct. Dec takes the secret key sk and a ciphertext ct as input and returns the underlying message m. Our protocol uses a public-key encryption scheme that is *CPA-secure*, which means an adversary cannot distinguish between the encryptions of two adversarially chosen plaintexts, even with access to the public key, except with negligible probability [61].

Somewhat Homomorphic Encryption (SHE). A SHE scheme $\mathsf{SHE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Sum}, \mathsf{Multiply})$ (e.g., [28]) supports addition and at least k multiplications over encrypted values, where k is the maximum number of datasets used to compute a regression. Sum is a polynomial-time algorithm that takes two SHE ciphertexts as input and returns a ciphertext corresponding to the sum of plaintext underlying the input ciphertexts. Similarly, $\mathsf{Multiply}$ is a polynomial-time algorithm that takes two SHE ciphertexts as input and returns a ciphertext corresponding to the product of the input ciphertexts. Our protocol uses a public-key SHE scheme that is CPA-secure.

Oblivious Pseudorandom Function (OPRF). An OPRF is a two-party protocol that involves a data owner with some input x and a server with a key k for some pseudorandom function F. The protocol is executed such that the data owner learns the result of $F_k(x)$ and the server learns nothing [43]. We describe our protocol in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid world, which functions like a real-world protocol execution except all parties have access to an ideal OPRF functionality $\mathcal{F}_{\mathsf{OPRF}}$—we elaborate on this in Section 6.

Structured Encryption (STE). Structured encryption is a cryptographic primitive that allows a data owner to encrypt a data structure for storage on an untrusted server and later query it using a key generated at setup time. We use STE schemes for both dictionaries and multi-maps in our protocol. In particular, we use *response-revealing* STE schemes, which reveal the query responses to the server. The security of STE schemes is formalized in a leakage-based model, where a leakage function is used to capture the information leaked about the data and queries to the adversary. Definitions 4.1 and 4.2 provide the syntax and security definition for static response-revealing STE schemes.

Definition 4.1 (Structured encryption [26]). A static, response-revealing structured encryption scheme $\Sigma_{\mathsf{DS}} = (\mathsf{Setup}, \mathsf{Token}, \mathsf{Query})$ for a data structure DS with query space $\mathbf{Q}$ consists of three PPT algorithms:

- $(K, \mathsf{EDS}) \leftarrow \mathsf{Setup}(1^k, \mathsf{DS})$: takes as input a security parameter 1^k, and a data structure DS. It outputs a secret key K and an encrypted structure EDS.
- $\mathsf{tk} \leftarrow \mathsf{Token}(K, q)$: is a possibly probabilistic algorithm that takes as input a secret key K and a query $q \in \mathbf{Q}$ and outputs a token tk.
- $r \leftarrow \mathsf{Query}(\mathsf{EDS}, \mathsf{tk})$: is a possibly probabilistic algorithm that takes as input an encrypted structure EDS and a token tk and outputs a response r.

We say $\Sigma_{\mathsf{DS}} = (\mathsf{Setup}, \mathsf{Token}, \mathsf{Query})$ is *correct* if, for all $k \in \mathbb{N}$, for all $\mathsf{poly}(k)$-size structures DS with query space $\mathbf{Q}$, for all $\mathsf{poly}(k)$-size sequences of queries $q_1, \ldots, q_s$ where $q_i \in \mathbf{Q}$, for all K and EDS output by $\mathsf{Setup}(1^k, \mathsf{DS})$, for all tk_i output by $\mathsf{Token}(K, q_i)$, $\mathsf{Query}(\mathsf{EDS}, \mathsf{tk}_i) = \mathsf{DS}[q_i]$ with all but negligible probability in k.

Definition 4.2 (Λ-security of STE [26], [31]). Let $\mathsf{STE} = (\mathsf{Setup}, \mathsf{Token}, \mathsf{Query})$ be a static response-revealing structured encryption scheme. Consider the following experiments where $\mathcal{C}$ is a stateful challenger, $\mathcal{A}$ is a stateful adversary, $\mathcal{S}$ is a stateful simulator, and $\Lambda = (\mathsf{patt}_{\mathsf{S}}, \mathsf{patt}_{\mathsf{Q}})$ is a leakage profile, and $z \in \{0,1\}^*$:

- $\mathbf{Real}_{\mathsf{STE},\mathcal{C},\mathcal{A}}(k)$: given z, the adversary $\mathcal{A}$ outputs a structure DS and receives EDS from the challenger, where $(K, \mathsf{EDS}) \leftarrow \mathsf{Setup}(1^k, \mathsf{DS})$. $\mathcal{A}$ then adaptively chooses a polynomial-size sequence of queries $(q_1, \ldots q_m)$. For all $1 \le i \le m$ the adversary receives tk_i where $\mathsf{tk}_i \leftarrow \mathsf{Token}(K, q_i)$. Finally, $\mathcal{A}$ outputs a bit b that is output by the experiment.
- $\mathbf{Ideal}_{\mathsf{STE},\mathcal{A},\mathcal{S}}(k)$: given z, the adversary $\mathcal{A}$ outputs a structure DS. Given $\mathsf{patt}_{\mathsf{S}}(\mathsf{DS})$, the simulator returns an encrypted structure EDS to $\mathcal{A}$. $\mathcal{A}$ then adaptively chooses a polynomial-size sequence of queries $(q_1, \ldots, q_m)$. For each $1 \le i \le m$, $\mathcal{S}$ is given $\mathsf{patt}_{\mathsf{Q}}(\mathsf{DS}, q_i)$ and r_i, where r_i is the response of the query q_i, and it returns a token tk_i to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a bit b that is output by the experiment.

We say STE is Λ-*secure* if there exists a PPT simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{A}$, and for all $z \in \{0,1\}^*$, $|\Pr[\mathbf{Real}_{\mathsf{STE},\mathcal{C},\mathcal{A}}(k) = 1] - \Pr[\mathbf{Ideal}_{\mathsf{STE},\mathcal{A},\mathcal{S}}(k) = 1]| \le \mathsf{negl}(k)$.

5. Query Language

Synq's *query language* (Synq-QL) allows an analyst to select data from any subset of the data owners, (optionally) apply column-based *filters*, link datasets by pre-specified sets of columns, and perform aggregate operations. A Synq-QL query consists of three operations in the following order.

1) **Filter.** Filters are expressed as a set of per-owner, conjunctive expressions of the form $(\mathbf{P}_i, x, \mathsf{value})$, where $\mathbf{P}_i$ is the data owner the filter applies to, x is a column identifier, and value is the value of the column.
2) **Link.** Synq-QL can optionally *link* data from multiple owners. The list of *linking conditions* $\mathsf{L} = [\perp, L_1, \ldots, L_m]$ is defined prior to protocol execution, where each L_i is a set of column identifiers. Analysts must choose a linking condition L_i from this list. Then two records are linked if they share the same values for the columns in L_i. When the linking condition is empty ($L_0 = \perp$), Synq-QL does not perform any linking. To enforce conjunctive filters, records that are not linked with any other records are ignored.
3) **Aggregate.** After filtering and linking, a set of aggregate functions are executed. Synq-QL aggregates are specified as trees composed of *base operators*.

   Definition 5.1 (Base Operators). The base operators are defined as follows:

   - $\mathsf{agg} \leftarrow (\mathsf{ColumnSum}, T, x)$: the *column sum* operator takes as input a column identifier x in table T and outputs $\Sigma_{\mathbf{r} \in T} \mathbf{r}[x]$.
   - $n \leftarrow (\mathsf{TableCount}, T)$: the *table count* operator outputs the number of entries in table T.

MA Ambulance Trip Record Information System (MATRIS) incident records ($\mathbf{P}_1$)

name	ssn	dob	diag	year
AA	1111	010199	overdose	2013
BB	2222	020201	overdose	2013
CC	3333	030305	overdose	2013
		...		

Prescription Drug Monitoring Program (PDMP) medication records ($\mathbf{P}_2$)

name	ssn	dob	med	cnt	pyear
AA	1111	010199	oxycodone	14	2013
CC	3333	030305	oxycodone	30	2013
DD	4444	121287	oxycodone	28	2012
		...			

"What is the # of patients who had a 2013 overdose and had oxycodone prescribed?"

Synq-QL: $([(\mathbf{P}_1, \mathsf{diag}, \texttt{"overdose"}),$
$(\mathbf{P}_1, \mathsf{year}, \texttt{2013}),$
$(\mathbf{P}_2, \mathsf{med}, \texttt{"oxycodone"}),$
$(\mathbf{P}_2, \mathsf{pyear}, \texttt{2013})],$
$1, [(\mathsf{TableCount}, T)])$

Figure 2. Running example for the relationship between opioid prescriptions and overdoses with the linking condition $L_1 = \{\mathsf{name}, \mathsf{ssn}, \mathsf{dob}\}$.

- $y \leftarrow (\mathsf{JoinMultiply}, T, x_1, x_2)$: the *joined multiplication* operator takes as input two column identifiers, x_1 and x_2 in table T, and computes a new column y where $\mathbf{r}[y] = \{(\mathbf{r}[x_1] \cdot \mathbf{r}[x_2]) : \mathbf{r} \in T\}$.

In Appendix A, we provide concrete examples of how to use Synq-QL's aggregation language to express Sum, Count, Average, Variance, linear Regression, and multiple Regression functions as required by [D7] from Section 2.

Running example. We present a representative, simplified workload from the MA DPH report [72] which examines the relationship between opioid prescriptions and opioid overdoses in 2013. Figure 2 provides two example datasets used in this analysis: MATRIS ($\mathbf{P}_1$), which contained records about ambulance trips, and PDMP ($\mathbf{P}_2$), which had records of restricted medication prescriptions across the state. Given the condition $L_1 = \{\mathsf{name}, \mathsf{ssn}, \mathsf{dob}\}$ and the schema of $\mathbf{P}_1$ and $\mathbf{P}_2$, the Synq-QL query in Figure 2 filters for all of $\mathbf{P}_1$'s records where $\mathsf{diag} = \texttt{"overdose"}$ and $\mathsf{year} = \texttt{2013}$ and for all of $\mathbf{P}_2$'s records where $\mathsf{med} = \texttt{"oxycodone"}$ and $\mathsf{pyear} = \texttt{2013}$, then counts the records in the linked dataset. Since all filters are treated as conjunctions, the resulting dataset only contains linked records that match all 4 filters. Then, $(\mathsf{TableCount}, T)$ outputs the number of records in the linked dataset.

6. Protocol

The Synq $= (\mathsf{Init}, \mathsf{Setup}, \mathsf{Query})$ protocol (described in Figures 4–5) supports the following parties:

- Server $\mathbf{S}$ that stores encrypted data structures and assists in executing queries.
- Data owners $\mathbf{P}_1, ..., \mathbf{P}_n$, where $n = poly(k)$ and k is the security parameter. Each $\mathbf{P}_i$ owns a table T_i and contributes data for analysis.
- Analyst $\mathbf{Q}$ that executes aggregate queries over the contributed data and receives the output of those queries.

Synq operates in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid model where all parties have access to $\mathcal{F}_{\mathsf{OPRF}}$ defined in Figure 3.

6.1. Linking

Recall Synq-QL supports linking over a chosen linking condition $L_i \in \mathsf{L}$. Records are linked if they share the same values for all the columns specified by L_i and then are treated as one logical record. L must be defined prior to protocol execution to allow for application-specific linking

$\mathcal{F}_{\mathsf{OPRF}} = (\mathsf{Init}, \mathsf{Eval})$ interacts with server $\mathbf{S}$ and party $\mathbf{P}$.

- Upon receiving (Init) from $\mathbf{S}$, the functionality initializes and stores an empty dictionary DX.
- Upon receiving (Eval, x) from $\mathbf{P}$, the functionality checks $\mathsf{DX}[x]$. If $\mathsf{DX}[x]$ is empty, the functionality samples $r \xleftarrow{\$} \{0,1\}^k$, sets $\mathsf{DX}[x] = r$, and returns r to $\mathbf{P}$. Else, the functionality returns $\mathsf{DX}[x]$ to $\mathbf{P}$. The functionality then sends a message d to $\mathbf{S}$.

Figure 3. $\mathcal{F}_{\mathsf{OPRF}}$: The OPRF functionality.

conditions. Synq's linking uses *link tags*, which are computed using a pseudorandom function (PRF) applied to all the values in the columns specified by the linking condition L_i. For example, the corresponding link tag for each record $\mathbf{r}$ under the linking condition L_1 from our running example in Figure 2 would be the output of the PRF for the input $\langle \mathbf{r}[\mathsf{name}] || \mathbf{r}[\mathsf{ssn}] || \mathbf{r}[\mathsf{dob}] \rangle$. To do this, each data owner must compute link tags using a PRF keyed on the same key. However, requiring data owners to utilize a shared PRF key would require either a public key infrastructure (violating [D1]) since the owners would have to maintain a secret key) or a synchronous key exchange (violating [D2]). We address all these concerns by using an OPRF, which guarantees that all data owners can generate PRF evaluations under the same key but learn only their own outputs. Our protocol is defined in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid world, where every data owner has access to an ideal OPRF functionality (Figure 3).

6.1.1. Instantiating the OPRF. Synq's formal security analysis assumes the $\mathcal{F}_{\mathsf{OPRF}}$ functionality is trusted. While the use of the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid model (and, more generally, analysis in the hybrid/ideal-world paradigm [23]) is a standard approach to modularly analyzing protocol security, the approach results in no formal OPRF "party" within the Synq protocol description. There are multiple ways to instantiate this OPRF functionality, such as using an MPC protocol between parties. We believe that the instantiation that best fits our design considerations is a separate *OPRF service*, where an additional party maintains an OPRF server that remains online throughout Setup. Each data owner computes linking tags on their own dataset by communicating with the OPRF server via a single-round protocol (e.g., [65], [79]). This specific instantiation works well with the MA DPH initiative [72], as two semi-trusted parties with a non-colluding assumption—the MA DPH and the Center for Health Information and Analysis (CHIA) [1]—were already used to heuristically secure the linking process. In the

Let $\mathbf{P}_1, \ldots, \mathbf{P}_n, \mathbf{Q}, \mathbf{S}$ be parties, $\Sigma_{\mathsf{DX}} = (\mathsf{Setup}, \mathsf{Token}, \mathsf{Query})$ be a response-revealing dictionary encryption scheme, let $\Sigma_{\mathsf{MM}} = (\mathsf{Setup}, \mathsf{Token}, \mathsf{Query})$ be a response-revealing multi-map encryption scheme, $\mathsf{SHE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Multiply})$ be a SHE scheme, $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme, and $\mathsf{L} = [\perp, L_1, \ldots, L_m]$ be a list of linking conditions. $\mathsf{Synq} = (\mathsf{Init}_{\mathbf{Q},\mathbf{S}}, \mathsf{Setup}_{\mathbf{P},\mathbf{S}}, \mathsf{Query}_{\mathbf{Q},\mathbf{S}})$ is then defined in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid model:

$\mathsf{Init}_{\mathbf{Q},\mathbf{S}}(1^k, \perp)$:

1) $\mathbf{Q}$ generates $(\mathsf{pk}_{\mathsf{num}}, \mathsf{sk}_{\mathsf{num}}) \leftarrow \mathsf{SHE.Gen}(1^k)$;
2) $\mathbf{Q}$ generates $(\mathsf{pk}_{\mathsf{key}}, \mathsf{sk}_{\mathsf{key}}) \leftarrow \mathsf{PKE.Gen}(1^k)$;
3) $\mathbf{S}$ sends (Init) to the ideal functionality $\mathcal{F}_{\mathsf{OPRF}}$;

$\mathsf{Setup}_{\mathbf{P},\mathbf{S}}(T, \perp)$:

1) $\mathbf{P}$ initializes multi-maps $\mathsf{MM}^{\mathsf{filter}}$, $\mathsf{MM}^{\mathsf{link}}$ & dictionary $\mathsf{DX}^{\mathsf{data}}$;
2) for each $\mathbf{r} \in T$, $\mathbf{P}$
   a) initializes dictionary $\mathsf{DX}_{\mathbf{r}}$;
   b) for all $x \in X^{\mathsf{Num}}$, sets $\mathsf{DX}_{\mathbf{r}}[x] := \mathsf{SHE.Enc}(\mathsf{pk}_{\mathsf{num}}, \mathbf{r}[x])$;
   c) samples an identifier $\mathsf{id}_{\mathbf{r}} \xleftarrow{\$} \{0,1\}^k$;
   d) sets $\mathsf{DX}^{\mathsf{data}}[\mathsf{id}_{\mathbf{r}}] := \mathsf{DX}_{\mathbf{r}}$;
3) $\mathbf{P}$ executes $(K^{\mathsf{data}}, \mathsf{EDX}^{\mathsf{data}}) \leftarrow \Sigma_{\mathsf{DX}}.\mathsf{Setup}(\mathsf{DX}^{\mathsf{data}})$;
4) for each $\mathbf{r} \in T$, $\mathbf{P}$,
   a) computes $\mathsf{tk}_{\mathbf{r}}^{\mathsf{data}} \leftarrow \Sigma_{\mathsf{DX}}.\mathsf{Token}(K^{\mathsf{data}}, \mathsf{id}_{\mathbf{r}})$;
   b) computes the linking tags for each linking condition $L_j \in \mathsf{L}$ using the functionality $\mathcal{F}_{\mathsf{OPRF}}$:
      i) sets $\mathsf{lid} \leftarrow (\mathbf{r}[x_1] || \ldots || \mathbf{r}[x_{|L_j|}])$ where $x \in L_j$;
      ii) sends $(\mathsf{Eval}, \mathsf{lid})$ to $\mathcal{F}_{\mathsf{OPRF}}$ and receives $\mathsf{ltg}_{\mathbf{r}}^j$;
   c) for all $x \in X^{\mathsf{Filter}}$,
      i) $\mathsf{MM}^{\mathsf{filter}}[\langle x \mid\mid \mathbf{r}[x] \rangle] \stackrel{+}{:=} \mathsf{tk}_{\mathbf{r}}^{\mathsf{data}}$;
      ii) for each $L_j \in \mathsf{L}$, $\mathsf{MM}^{\mathsf{link}}[\langle x \mid\mid \mathbf{r}[x] \mid\mid j \rangle] \stackrel{+}{:=} \mathsf{ltg}_{\mathbf{r}}^j$;
5) $\mathbf{P}$ sets up $(K^{\mathsf{filter}}, \mathsf{EMM}^{\mathsf{filter}}) \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Setup}(\mathsf{MM}^{\mathsf{filter}})$;
6) $\mathbf{P}$ sets up $(K^{\mathsf{link}}, \mathsf{EMM}^{\mathsf{link}}) \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Setup}(\mathsf{MM}^{\mathsf{link}})$;
7) $\mathbf{P}$ computes $\mathsf{ct}_K \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{key}}, K)$ where $K = (K^{\mathsf{filter}}, K^{\mathsf{link}}, K^{\mathsf{data}})$;
8) $\mathbf{P}$ sends $(\mathsf{EDS}, \mathsf{ct}_K)$ to $\mathbf{S}$ where $\mathsf{EDS} = (\mathsf{EMM}^{\mathsf{filter}}, \mathsf{EMM}^{\mathsf{link}}, \mathsf{EDX}^{\mathsf{data}})$;

Figure 4. The Synq protocol.

$\mathsf{Query}_{\mathbf{Q},\mathbf{S}}(q, \mathsf{EDS})$:

1) $\mathbf{Q}$ computes $K_i \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{key}}, \mathsf{ct}_{K_i})$ for each data owner, where $K_i = (K_i^{\mathsf{link}}, K_i^{\mathsf{filter}}, K_i^{\mathsf{data}})$;
2) $\mathbf{Q}$ parses $q = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$ and, for each $(\mathbf{P}_i, x, \mathsf{value}) \in \mathsf{filter}$,
   a) computes $\mathsf{ftk} \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_i^{\mathsf{filter}}, \langle x, \mathsf{value} \rangle)$;
   b) computes $\mathsf{ltk} \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_i^{\mathsf{link}}, \langle x, \mathsf{value}, \mathsf{link} \rangle)$;
   c) sets $\mathsf{filtertk} = \mathsf{filtertk} \cup (\mathbf{P}_i, \mathsf{ftk}, \mathsf{ltk})$;
3) $\mathbf{Q}$ sends $(\mathsf{filtertk}, \mathsf{aggregate})$ to $\mathbf{S}$;
4) $\mathbf{S}$ initializes a set tags and a multi-map MM;
5) for each $(\mathbf{P}_i, \mathsf{ftk}, \mathsf{ltk})$ in $\mathsf{filtertk}$,
   a) computes $\mathsf{ltgs} \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Query}(\mathsf{EMM}_i^{\mathsf{link}}, \mathsf{ltk})$;
   b) computes $\mathsf{tks} \leftarrow \Sigma_{\mathsf{MM}}.\mathsf{Query}(\mathsf{EMM}_i^{\mathsf{filter}}, \mathsf{ftk})$;
   c) for each $\mathsf{ltg}_{\mathbf{r}}$ in ltgs, and each corresponding $\mathsf{tk}_{\mathbf{r}}^{\mathsf{data}}$ in tks, $\mathbf{S}$ sets $\mathsf{MM}[\mathsf{ltg}_{\mathbf{r}}] \stackrel{+}{:=} (\mathbf{P}_i, \mathsf{tk}_{\mathbf{r}}^{\mathsf{data}})$;
6) $\mathbf{S}$ initializes a table T^{link} with columns $X^{\mathsf{link}} = X_i \cup X_j$;
7) for each label ltg in MM with $\mathsf{MM}[\mathsf{ltg}] = \langle (\mathbf{P}_i, \mathsf{tk}_{\mathbf{r}}^{\mathsf{data}}), (\mathbf{P}_j, \mathsf{tk}_{\mathbf{r}'}^{\mathsf{data}}) \rangle$ (i.e., where both filters were satisfied), $\mathbf{S}$ creates a linked record $\mathbf{r}^*$ in T^{link} as follows:
   a) $\mathbf{S}$ retrieves $\mathsf{DX}_{\mathbf{r}} \leftarrow \Sigma_{\mathsf{DX}}.\mathsf{Query}(\mathsf{DX}_i^{\mathsf{data}}, \mathsf{tk}_{\mathbf{r}}^{\mathsf{data}})$ and $\mathsf{DX}_{\mathbf{r}'} \leftarrow \Sigma_{\mathsf{DX}}.\mathsf{Query}(\mathsf{DX}_j^{\mathsf{data}}, \mathsf{tk}_{\mathbf{r}'}^{\mathsf{data}})$;
   b) for each $x \in X_i$, $\mathbf{S}$ sets $\mathbf{r}^*[x] = \mathsf{DX}_{\mathbf{r}}[x]$;
   c) for each $x \in X_j$, $\mathbf{S}$ sets $\mathbf{r}^*[x] = \mathsf{DX}_{\mathbf{r}'}[x]$;
8) for all $\mathsf{tree}_i \in \mathsf{aggregate}$, $\mathbf{S}$ post-order traverses each node $N \in \mathsf{tree}_i$ and computes the following:
   a) if $N \equiv (\mathsf{ColumnSum}, T^{\mathsf{link}}, x)$,
      i) set $\mathsf{res}_N = 0$;
      ii) for all $\mathbf{r} \in T^{\mathsf{link}}$, $\mathsf{res}_N \leftarrow \mathsf{SHE.Add}(\mathsf{res}_N, \mathbf{r}[x])$;
   b) if $N \equiv (\mathsf{TableCount}, T^{\mathsf{link}})$,
      i) set $\mathsf{res}_N = 0$;
      ii) for all $\mathbf{r} \in T^{\mathsf{link}}$, $\mathsf{res}_N \leftarrow \mathsf{res}_N + 1$;
   c) if $N \equiv (\mathsf{JoinMultiply}, T^{\mathsf{link}}, x_1, x_2)$, for all $\mathbf{r} \in T^{\mathsf{link}}$, $\mathbf{r}[(x_1||x_2)] \leftarrow \mathsf{SHE.Multiply}(\mathbf{r}[x_1], \mathbf{r}[x_2])$;
9) for all $\mathsf{tree}_i \in \mathsf{aggregate}$, $\mathbf{S}$ gets root node R, sets $\mathsf{res}_i \leftarrow \mathsf{res}_R$, and sends res_i to $\mathbf{Q}$;
10) for all res_i, $\mathbf{Q}$ computes $\mathsf{res}_i \leftarrow \mathsf{SHE.Dec}(\mathsf{sk}_{\mathsf{num}}, \mathsf{res}_i)$;
11) $\mathbf{Q}$ computes the final aggregate result;

Figure 5. The Synq protocol. For Query pseudocode simplicity, we assume that queries involve two data owners, $\mathbf{P}_i, \mathbf{P}_j$, though the protocol easily generalizes to more.

original study, CHIA processed each dataset individually and mapped the plaintext datasets between individuals and unique identifiers for each individual, scrubbed the datasets of personally identifiying information (with the exception of the unique identifier), and passed the datasets back to DPH. DPH then used the unique identifier to link records together. CHIA was only used as part of the linking process and did not participate in the analysis. A Synq-based deployment of the MA DPH initiative would leverage the already-existing trust assumptions of the MA DPH and CHIA. While MA DPH would maintain the server $\mathbf{S}$, CHIA would operate the OPRF service that data owners would communicate with to generate linking tags. Just as in the original study, CHIA only needs to remain online throughout Setup.

This two-party, non-colluding assumption has also been utilized in other practical public policy contexts and thus this specific instantiation of Synq may also work well for other studies (e.g., [12], [83]), and similar assumptions commonly appear in "outsourced MPC" work (e.g., those from Table 2 with topology (T2)) that rely on secret sharing of data between multiple non-colluding servers. In those works, the adversary learns all data in plaintext if more than a specified threshold of the compute servers are corrupted. Conversely, in our proposed OPRF instantiation, if both $\mathbf{S}$ and the OPRF service are corrupted, the adversary learns the PRF key and can then compute linking tags on arbitrary data—but the OPRF service never directly sees the plaintext of honest data owners and therefore an adversary who corrupts the OPRF server also cannot see this data in plaintext. An adversary with auxiliary information (e.g., possible inputs for individual identifiers) may directly evaluate the OPRF in a dictionary attack in an attempt to learn information about specific individuals of interest. However, in Section 7.2.3, we show that similar prior systems reveal significant information without any auxiliary information.

6.2. Description of Protocol

6.2.1. Initialization. In Init, the analyst $\mathbf{Q}$ generates PKE key pair $(\mathsf{pk}_{\mathsf{key}}, \mathsf{sk}_{\mathsf{key}})$. The public key $\mathsf{pk}_{\mathsf{key}}$ is later used by data owners to encrypt STE keys for the analyst. The analyst $\mathbf{Q}$ additionally generates $(\mathsf{pk}_{\mathsf{num}}, \mathsf{sk}_{\mathsf{num}})$ using SHE.

PDMP ($\mathbf{P}_2$) medication records

	name	ssn	dob	med	cnt	pyear
r1	AA	1111	010199	oxycodone	14	2013
r2	CC	3333	030305	oxycodone	30	2013
r3	DD	4444	121287	oxycodone	28	2012

Given $\mathsf{L} = \{L_1\}$ where $L_1 = \{\mathsf{name}, \mathsf{ssn}, \mathsf{dob}\}$, $X^{\mathsf{Filter}} = \{\mathsf{med}, \mathsf{pyear}\}$, and $X^{\mathsf{Num}} = \{\mathsf{cnt}\}$, $\mathbf{P}_2$ uses $\mathcal{F}_{\mathsf{OPRF}}$ to compute

$\mathsf{ltg}^1_{\mathbf{r}1} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{AA}||\mathtt{1111}|\mathtt{010199}\rangle)$,
$\mathsf{ltg}^1_{\mathbf{r}2} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{CC}||\mathtt{3333}|\mathtt{030305}\rangle)$, and
$\mathsf{ltg}^1_{\mathbf{r}3} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{DD}||\mathtt{4444}|\mathtt{121287}\rangle)$, and constructs:

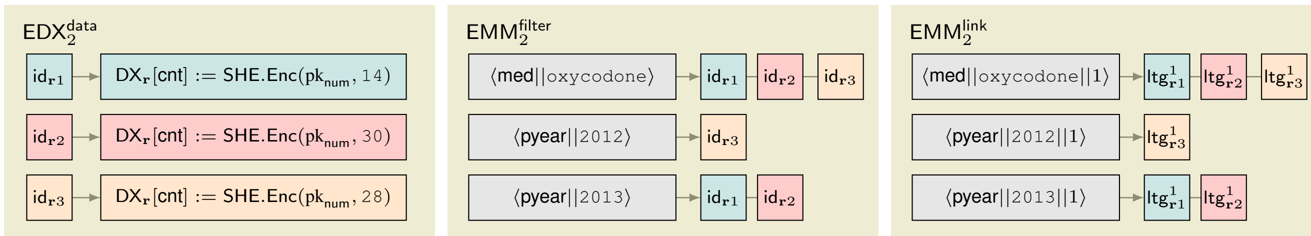


Figure 6. Running example for Setup using $\mathbf{P}_2$'s dataset from Figure 2.

MATRIS ($\mathbf{P}_1$) incident records

	name	ssn	dob	diag	year
r1	AA	1111	010199	overdose	2013
r2	BB	2222	020201	overdose	2013
r3	CC	3333	030305	overdose	2013

Given $\mathsf{L} = \{L_1\}$ where $L_1 = \{\mathsf{name}, \mathsf{ssn}, \mathsf{dob}\}$, $X^{\mathsf{Filter}} = \{\mathsf{diag}, \mathsf{year}\}$, and $X^{\mathsf{Num}} = \emptyset$, $\mathbf{P}_1$ uses $\mathcal{F}_{\mathsf{OPRF}}$ to compute

$\mathsf{ltg}^1_{\mathbf{r}1} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{AA}||\mathtt{1111}|\mathtt{010199}\rangle)$,
$\mathsf{ltg}^1_{\mathbf{r}2} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{BB}||\mathtt{2222}|\mathtt{020201}\rangle)$, and
$\mathsf{ltg}^1_{\mathbf{r}3} = \mathcal{F}_{\mathsf{OPRF}}(\mathsf{Eval}, \langle \mathtt{CC}||\mathtt{3333}|\mathtt{030305}\rangle)$, and constructs:

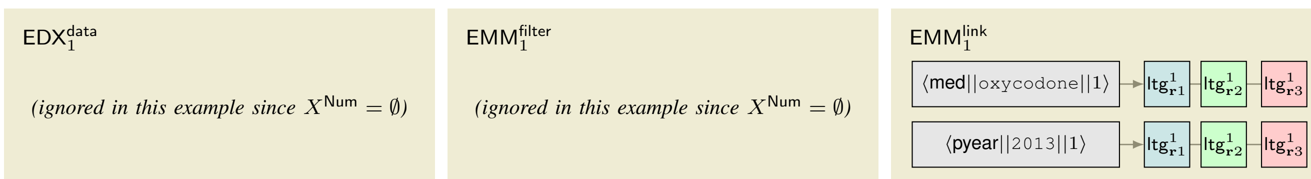


Figure 7. Running example for Setup using $\mathbf{P}_1$'s dataset from Figure 2.

$\mathsf{pk}_{\mathsf{num}}$ is later used by data owners to encrypt numeric data values within their datasets T_i, in order to support aggregate queries. Lastly, the server sends (Init) to initialize $\mathcal{F}_{\mathsf{OPRF}}$.

6.2.2. Setup. In Setup, each data owner $\mathbf{P}_i$ prepares their dataset T_i for analyses. Each $\mathbf{P}_i$ independently executes Setup with the server $\mathbf{S}$. During Setup, each $\mathbf{P}_i$ computes the link tags for T_i and initializes the following structures:

- $\mathsf{DX}^{\mathsf{data}}$ maps randomly generated record identifiers to SHE-encrypted records.
- $\mathsf{MM}^{\mathsf{link}}$ maps a column, value, and linking condition to the link tags for all the records with that column value.
- $\mathsf{MM}^{\mathsf{filter}}$ maps a column and value to encrypted tokens for all the records containing that column and value.

Encrypting data values. The data owner $\mathbf{P}_i$ encrypts their numeric data values using SHE under the analyst's public key ($\mathsf{pk}_{\mathsf{num}}$). For all numeric columns $x \in X^{\mathsf{Num}}$, $\mathbf{P}_i$ encrypts every value $\mathbf{r}[x]$ in T such that $\mathsf{ct} := \mathsf{SHE.Enc}(\mathsf{pk}_{\mathsf{num}}, \mathbf{r}[x])$. The encrypted records are stored in $\mathsf{DX}^{\mathsf{data}}$ under a randomly generated identifier $\mathsf{id}_{\mathbf{r}}$. $\mathbf{P}_i$ then encrypts $\mathsf{DX}^{\mathsf{data}}$, which contains all the SHE-encrypted records, to generate an encrypted dictionary $\mathsf{EDX}^{\mathsf{data}}$. $\mathbf{S}$ later uses these encrypted values to evaluate aggregate functions.

Generating link tags. After encrypting their data, $\mathbf{P}_i$ generates link tags for each linking condition $L_j \in \mathsf{L}$. For each record, $\mathbf{P}_i$ creates a link identifier $\mathsf{lid} := \mathbf{r}[x_1]||\ldots||\mathbf{r}[x_j]$, where $x_j \in L_j$, which is a concatenation of the values in columns specified in L_j. To obtain the link tag for each lid, $\mathbf{P}_i$ sends the message $(\mathsf{Eval}, \mathsf{lid})$ to $\mathcal{F}_{\mathsf{OPRF}}$ and receives $\mathsf{ltg}^j_{\mathbf{r}}$. Since $\mathsf{ltg}^j_{\mathbf{r}}$ is the result of an OPRF, it will be identical for records with the same lid. These linking tags later enable the untrusted server to link the corresponding records without learning the underlying values. For each column x in the set of filterable columns X^{Filter}, $\mathbf{P}_i$ adds $\mathsf{ltg}^j_{\mathbf{r}}$ to the tuple $\mathsf{MM}^{\mathsf{link}}[\langle x \mid\mid \mathbf{r}[x] \mid\mid j\rangle]$, where j is the index of L_j. $\mathsf{MM}^{\mathsf{link}}$ can then be queried based on a column, value, and linking condition to retrieve the linking tags of all records that contain that column and value.

Setting up data filters. Using K^{data}, $\mathbf{P}_i$ computes a token $\mathsf{tk}^{\mathsf{data}}_{\mathbf{r}} \leftarrow \Sigma_{\mathsf{DX}}.\mathsf{Token}(K^{\mathsf{data}}, \mathsf{id}_{\mathbf{r}})$ for each record $\mathbf{r}$ in T_i. These tokens are used to retrieve records from the encrypted dictionary $\mathsf{EDX}^{\mathsf{data}}$ and are inserted in $\mathsf{MM}^{\mathsf{filter}}[\langle x \mid\mid \mathbf{r}[x]\rangle]$ so that it can be queried based on a column identifier and its corresponding value. Together, $\mathsf{MM}^{\mathsf{filter}}$ and $\mathsf{MM}^{\mathsf{link}}$ support (1) the retrieval of records that satisfy a particular filter, and (2) the linking of the filtered records. $\mathbf{P}_i$ then encrypts $\mathsf{MM}^{\mathsf{filter}}$ and $\mathsf{MM}^{\mathsf{link}}$ and encrypts the generated STE keys under $\mathbf{Q}$'s public key. $\mathbf{P}_i$ then sends the encrypted database $\mathsf{EDB} = (\mathsf{EDX}^{\mathsf{data}}, \mathsf{EMM}^{\mathsf{filter}}, \mathsf{EMM}^{\mathsf{link}})$ and the encrypted keys to $\mathbf{S}$. Figures 6 and 7 show examples of the Setup structures computed from our running example in Figure 2.

Resubmission. Since each EDB_i is contributed and

Given q from Figure 2, $\mathbf{Q}$ constructs and sends $(\mathsf{filtertk}, [(\mathsf{TableCount}, T)])$ where:

$$\begin{aligned}
\mathsf{filtertk} = \{(&\mathbf{P}_1, \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_1^{\mathsf{filter}}, \langle \mathsf{diag}, \texttt{"overdose"} \rangle), && \text{from } \mathsf{EMM}_1^{\mathsf{filter}}\text{, retrieves } \emptyset \\
&\Sigma_{\mathsf{MM}}.\mathsf{Token}(K_1^{\mathsf{link}}, \langle \mathsf{diag}, \texttt{"overdose"}, 1 \rangle)) && \text{from } \mathsf{EMM}_1^{\mathsf{link}}\text{, retrieves } \{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
(&\mathbf{P}_1, \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_1^{\mathsf{filter}}, \langle \mathsf{year}, \texttt{2013} \rangle), && \text{from } \mathsf{EMM}_1^{\mathsf{filter}}\text{, retrieves } \emptyset \\
&\Sigma_{\mathsf{MM}}.\mathsf{Token}(K_1^{\mathsf{link}}, \langle \mathsf{year}, \texttt{2013}, 1 \rangle)) && \text{from } \mathsf{EMM}_1^{\mathsf{link}}\text{, retrieves } \{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
(&\mathbf{P}_2, \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_2^{\mathsf{filter}}, \langle \mathsf{med}, \texttt{"oxycodone"} \rangle), && \text{from } \mathsf{EMM}_2^{\mathsf{filter}}\text{, retrieves } \{\mathsf{id}_{\mathbf{r}1}, \mathsf{id}_{\mathbf{r}2}, \mathsf{id}_{\mathbf{r}3}\} \\
&\Sigma_{\mathsf{MM}}.\mathsf{Token}(K_2^{\mathsf{link}}, \langle \mathsf{med}, \texttt{"oxycodone"}, 1 \rangle)) && \text{from } \mathsf{EMM}_2^{\mathsf{link}}\text{, retrieves } \{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
(&\mathbf{P}_2, \Sigma_{\mathsf{MM}}.\mathsf{Token}(K_2^{\mathsf{filter}}, \langle \mathsf{pyear}, \texttt{2013} \rangle), && \text{from } \mathsf{EMM}_2^{\mathsf{filter}}\text{, retrieves } \{\mathsf{id}_{\mathbf{r}1}, \mathsf{id}_{\mathbf{r}2}\} \\
&\Sigma_{\mathsf{MM}}.\mathsf{Token}(K_2^{\mathsf{link}}, \langle \mathsf{pyear}, \texttt{2013}, 1 \rangle))\} && \text{from } \mathsf{EMM}_2^{\mathsf{link}}\text{, retrieves } \{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1\}
\end{aligned}$$

In (4) and (5), $\mathbf{S}$ takes the intersection of linking tags to determine the final record ids to include in T^{link}:

$$\begin{aligned}
&\{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
\cap &\{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
\cap &\{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1, \mathsf{ltg}_{\mathbf{r}3}^1\} \\
\cap &\{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1\} \\
= &\{\mathsf{ltg}_{\mathbf{r}1}^1, \mathsf{ltg}_{\mathbf{r}2}^1\},
\end{aligned}$$

$\mathbf{S}$ uses corresponding record ids $\{\mathsf{id}_{\mathbf{r}1}, \mathsf{id}_{\mathbf{r}2}\}$ to link data from $\mathsf{EDX}_1^{\mathsf{data}}$ and $\mathsf{EDX}_2^{\mathsf{data}}$ in T^{link}.

Figure 8. Running example for Query using the datasets and query from Figure 2 and the Setup-generated structures from Figure 6 and 7.

stored independently, a data owner $\mathbf{P}_i$ can rerun Setup if they need to resubmit their data prior to the query phase. In this case, $\mathbf{S}$ replaces any structures previously uploaded by $\mathbf{P}_i$ with the new ones. This allows data owners to correct mistakes without affecting other owners' contributions.

6.2.3. Query. In Query, the analyst $\mathbf{Q}$ and server $\mathbf{S}$ evaluate queries from the Synq-QL language from Section 5. $\mathbf{Q}$ uses its secret key $\mathsf{sk}_{\mathsf{key}}$ to decrypt each ct_{K_i} and receive $K_i = (K_i^{\mathsf{filter}}, K_i^{\mathsf{link}})$ for $\mathbf{P}_i$. For each filter $(\mathbf{P}_i, x, \mathsf{value})$, $\mathbf{Q}$ uses K_i^{filter} to compute a filter token ftk based on the filter's column x and value, and uses K_i^{link} to compute a link token ltk based on x, value, and linking index link.

Linking. $\mathbf{S}$ uses each filter token ftk to query $\mathsf{EMM}^{\mathsf{filter}}$ for tokens to query $\mathsf{EDX}^{\mathsf{data}}$ for encrypted records, and each link token ltk to query $\mathsf{EMM}^{\mathsf{link}}$ for link tags. $\mathbf{S}$ uses a multimap MM to map link tags to pairs $(\mathbf{P}_i, \mathsf{tk}_{\mathbf{r}}^{\mathsf{data}})$ so it can retrieve linked records from each $\mathsf{EDX}_i^{\mathsf{data}}$ using the tokens corresponding to every link tag. Using MM, $\mathbf{S}$ populates a new table T_{link} such that all data corresponding to the same link tag is treated as a consolidated record. Each row of T_{link} represents data from one linked record.

Aggregates. Once data has been linked, $\mathbf{S}$ uses T_{link} to evaluate Synq-QL aggregate trees composed of the base operators from Definition 5.1. Trees are evaluated as follows:

- $\mathsf{ColumnSum}(T, x)$: Given a table and column, $\mathbf{S}$ adds the corresponding SHE ciphertexts.
- $\mathsf{TableCount}(\mathsf{TableCount})$: For each record in T, $\mathbf{S}$ increments a counter, which is then returned.
- $\mathsf{JoinMultiply}(T, x_1, x_2)$: For each record in T, $\mathbf{S}$ performs multiplication using $\mathbf{r}[x_1]$ and $\mathbf{r}[x_2]$, which have been encrypted using SHE. The result is stored in a new column $\mathbf{r}[(x_1||x_2)]$. The updated table is returned.

$\mathbf{S}$ returns a res_i resulting from the evaluation of each tree. $\mathbf{Q}$ decrypts each res_i using its secret key $\mathsf{sk}_{\mathsf{num}}$ and performs any final computation based on their query workload. For example, using the instantiations in Appendix A, for an Sum or Count function, $\mathbf{Q}$ performs no additional computation, but for an Average, Variance, or Regression function, $\mathbf{Q}$ receives sub-aggregates which it uses to compute the final result. Figure 8 shows how Query (using the Synq-QL query from Figure 2) interacts with the structures from Figure 6.

7. Security

7.1. Definitions

Recall we formalize security in the hybrid/ideal-world paradigm [23], which requires that a protocol execution in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid world (which assumes existence of an ideal, trusted OPRF functionality) is indistinguishable from one in the ideal world. Figure 9 defines the ideal functionality for Synq. Both the hybrid-world and the ideal-world executions take place between an environment $\mathcal{Z}$ and an adversary. In the hybrid-world execution, we denote the adversary $\mathcal{A}$. In the ideal-world, we denote the adversary $\mathcal{S}$ to represent a simulator. Both executions include the parties $\mathbf{S}, \mathbf{P}_1, ..., \mathbf{P}_n$, and $\mathbf{Q}$ as defined in Section 6. We use the semi-honest model as Synq is designed for policy studies where parties work collectively to ensure study success.

Corruptions. We consider two kinds of corruptions: (1) the server $\mathbf{S}$ and up to $n-1$ data owners or (2) the analyst $\mathbf{Q}$. If $\mathbf{S}$ and up to $n-1$ data owners are corrupted, $\mathbf{S}$ only learns the leakage during protocol execution, and the owners learn nothing more than their own inputs. If $\mathbf{Q}$ is corrupted, it only learns results of its queries and is trusted not to collude with any other party. For completeness, we note that the ideal OPRF functionality, by definition, cannot be corrupted by the adversary. For example, if the OPRF functionality is instantiated using a separate OPRF service as in Section 6.1.1, our corruption model implies that the OPRF service is fully trusted and thus does not collude with the adversary.

Hybrid-world execution. In the hybrid-world, every party has access to $\mathcal{F}_{\mathsf{OPRF}}$. The environment $\mathcal{Z}$ takes as input a string $z \in \{0,1\}^*$ and chooses a set of parties I for $\mathcal{A}$ to corrupt, where I is selected according to *one* of the defined categories of corruptions. $\mathcal{Z}$ sends I to $\mathcal{A}$, which corrupts the parties in I and has access to their inputs and outputs, as well as their intermediate states during execution. $\mathcal{Z}$ chooses all the data owners' inputs. In the query phase, $\mathcal{Z}$ selects $m = poly(k)$ queries $q_1, \ldots, q_m$ for $\mathbf{Q}$. The parties then execute the protocols to setup and query Synq. At the end of the execution, $\mathcal{A}$ sends an arbitrary message to $\mathcal{Z}$, which outputs a bit b. We denote this bit $\mathbf{Hybrid}_{\mathcal{Z},\mathcal{A}}(k)$.

Ideal-world execution. In the ideal-world, all parties have access to the ideal functionality $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$ (Figure 9). The

$\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L} = (\mathsf{Init}, \mathsf{Setup}, \mathsf{Query})$ is parameterized with a leakage profile $\Lambda = (\mathcal{L}_\mathsf{I}, \mathcal{L}_\mathsf{S}, \mathcal{L}_\mathsf{Q})$ and linking conditions $\mathsf{L} = [\perp, L_1, \ldots, L_m]$. It interacts with parties $\{\mathbf{P}_1, \ldots, \mathbf{P}_n\}$, server $\mathbf{S}$, analyst $\mathbf{Q}$, and an ideal adversary $\mathcal{S}$ using the following operations:

- Upon receiving (Init) from $\mathbf{Q}$ and $\mathbf{S}$, the functionality sends $\mathcal{L}_\mathsf{I}(\perp)$ to $\mathcal{S}$.
- Upon receiving (Setup, T_i) from $\mathbf{P}_i$, store T_i. If the server $\mathbf{S}$ is corrupted, send $\mathcal{L}_\mathsf{S}(T_i)$ to the simulator $\mathcal{S}$. For each record in T_i,
  (1) Sample $\mathsf{rid} \leftarrow \{0,1\}^*$, and,
  (2) Add rid to T_i such that $\mathbf{r}[\text{“rid”}] = \mathsf{rid}$.
- Upon receiving $(\mathsf{Query}, q) = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate}))$,
  (1) Initialize tables $(T'_1, \ldots, T'_n)$ such that $T'_i = T_i$,
  (2) (Filtering) For each $(\mathbf{P}_i, x, \mathsf{value})$ in filter, for each $\mathbf{r} \in T'_i$, if $\mathbf{r}[x] \neq \mathsf{value}$, remove $\mathbf{r}$ from T'_i.
  (3) Initialize an empty multi-map MM.
  (4) (Linking) If link is not empty, for each T'_i, for each $\mathbf{r} \in T'_i$ and $x_i \in L_{\mathsf{link}}$, append $\mathbf{r}$ to $\mathsf{MM}[(\mathbf{r}[x_0], \ldots, \mathbf{r}[x_{|L_{\mathsf{link}}|}])]$.
  (5) (Aggregate) If link is empty, use $(T'_1, \ldots, T'_n)$ to compute aggregate and store the output in res. Otherwise, use MM to compute aggregate and store the output in res.
  (6) If $\mathbf{S}$ is corrupted, return $\mathcal{L}_\mathsf{Q}(T_1, \ldots, T_n)$ to $\mathcal{S}$ and res to $\mathbf{Q}$.
  (7) If $\mathbf{Q}$ is corrupted, return res to $\mathcal{S}$.

Figure 9. $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$: The Synq functionality.

environment $\mathcal{Z}$ takes as input a string $z \in \{0,1\}^*$ and chooses a set of parties I for the adversary to corrupt, where I is selected according to *one* of the defined categories of corruptions. $\mathcal{Z}$ sends I to the simulator $\mathcal{S}$ and $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$. $\mathcal{Z}$ then chooses the inputs for all the parties $\mathbf{P}_i$ and the queries for the analyst $\mathbf{Q}$. $\mathcal{S}$ receives the inputs for all the parties in I and interacts with $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$ on behalf of the corrupted parties.

- **Initialization.** $\mathbf{S}$ and $\mathbf{Q}$ send (Init) to $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$.
- **Setup.** Each honest $\mathbf{P}_i$ sends (Setup, T_i) to $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$.
- **Query.** $\mathbf{Q}$ receives queries $q_1, \ldots, q_m$ from $\mathcal{Z}$, where each query is of the form $q_j = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$. If $\mathbf{Q}$ is honest, it sends (Query, q_j) to $\mathcal{F}_{\mathsf{Synq}}^{\Lambda,L}$.

At the end of the execution, $\mathcal{S}$ sends an arbitrary message to $\mathcal{Z}$ which then outputs a bit b, which we denote $\mathbf{Ideal}^{\Lambda}_{\mathcal{Z},\mathcal{S}}(k)$.

Definition 7.1 (Λ-security of* Synq*). Synq is Λ-secure if for all PPT semi-honest adversaries $\mathcal{A}$, there exists a PPT ideal adversary $\mathcal{S}$ such that for all PPT standalone environments $\mathcal{Z}$, for all $z \in \{0,1\}^*$,

$$|\Pr[\mathbf{Hybrid}_{\mathcal{Z},\mathcal{A}}(k) = 1] - \Pr[\mathbf{Ideal}^{\Lambda}_{\mathcal{Z},\mathcal{S}}(k) = 1]| \leq \mathsf{negl}(k).$$

7.2. Formal Analysis

We now formally analyze the security of Synq, which uses a response-revealing multi-map encryption scheme Σ_{MM}, a response-revealing dictionary encryption scheme Σ_{DX}, a CPA-secure public-key encryption scheme PKE, and a CPA-secure somewhat homomorphic encryption scheme SHE. We prove the following properties in the $\mathcal{F}_{\mathsf{OPRF}}$-hybrid world with respect to the leakage function Λ_{Synq}:

- Synq is Λ_{Synq}-secure when $\mathbf{S}$ and up to $(n-1)$ data owners are corrupted, and;
- Synq only reveals query results to a semi-honest $\mathbf{Q}$.

7.2.1. Server and Data Owners.

We first analyze the black-box leakage of Synq when an adversary corrupts the server and up to $(n-1)$ data owners. A black-box leakage analysis is used to express the leakage of a system in terms of the leakage of its building blocks, which can be switched out in order to obtain better security and/or efficiency properties.

Black-box leakage analysis. Suppose the leakage profiles of the response-revealing encrypted dictionary scheme Σ_{DX} and the response-revealing encrypted multi-map scheme Σ_{MM} are

$$\Lambda_{\mathsf{DX}} = (\mathcal{L}^{\mathsf{DX}}_{\mathsf{S}}, \mathcal{L}^{\mathsf{DX}}_{\mathsf{Q}}) = (\mathsf{patt}^{\mathsf{DX}}_{\mathsf{S}}, \mathsf{patt}^{\mathsf{DX}}_{\mathsf{Q}}),$$
$$\Lambda_{\mathsf{MM}} = (\mathcal{L}^{\mathsf{MM}}_{\mathsf{S}}, \mathcal{L}^{\mathsf{MM}}_{\mathsf{Q}}) = (\mathsf{patt}^{\mathsf{MM}}_{\mathsf{S}}, \mathsf{patt}^{\mathsf{MM}}_{\mathsf{Q}}),$$

where Λ_{MM} is *content oblivious* [74], then Synq is Λ_{Synq}-secure where:

$$\Lambda_{\mathsf{Synq}} = (\mathcal{L}_\mathsf{I} = \perp, \mathcal{L}_\mathsf{S}, \mathcal{L}_\mathsf{Q}),$$
$$\mathcal{L}_\mathsf{S} = (\mathsf{patt}^{\mathsf{DX}}_{\mathsf{S}}(\mathsf{DX}^{\mathsf{data}}_i), \mathsf{patt}^{\mathsf{MM}}_{\mathsf{S}}(\mathsf{MM}^{\mathsf{filter}}_i), \mathsf{patt}^{\mathsf{MM}}_{\mathsf{S}}(\mathsf{MM}^{\mathsf{link}}_i)), \text{ for all } \mathbf{P}_i, \text{ and}$$
$$\mathcal{L}_\mathsf{Q} = (\mathsf{patt}^{\mathsf{DX}}_{\mathsf{Q}}(\mathsf{DX}^{\mathsf{data}}_j), \mathsf{patt}^{\mathsf{MM}}_{\mathsf{Q}}(\mathsf{MM}^{\mathsf{filter}}_j), \mathsf{patt}^{\mathsf{MM}}_{\mathsf{Q}}(\mathsf{MM}^{\mathsf{link}}_j), \mathsf{op}, \mathsf{G}^{\mathsf{link}}), \quad \text{for all queried } \mathbf{P}_j.$$

where $\mathsf{op} = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$ is the analyst's query, and $\mathsf{G}^{\mathsf{link}}$ is the *linking graph* of the queried records.

The linking graph $\mathsf{G}^{\mathsf{link}} = (V, E)$ represents the leakage revealed to the server during a query. The resulting $\mathsf{G}^{\mathsf{link}}$ of $q = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$ contains one vertex $\mathsf{v}_\mathbf{r}$ for each record $\mathbf{r}$ that satisfies at least one of the filters in q. Each vertex has 2 attributes: (1) rid, which is a unique identifier for the record $\mathbf{r}$, and (2) filterlist, which identifies the filter(s) in q that output the record $\mathbf{r}$. We note that the list filterlist need not contain the exact filter that output the record, but for convenience we will refer to a filter as $(\mathbf{P}_i, x, \mathsf{value})$. Each edge $e = (\mathsf{v}_\mathbf{r}, \mathsf{v}_{\mathbf{r}'})$ denotes a link between $\mathbf{r}$ and $\mathbf{r}'$, i.e., $\mathbf{r}[x] = \mathbf{r}'[x]$, for all $x \in L_{\mathsf{link}}$, where L_{link} is q's linking condition. If link is empty, $\mathsf{G}^{\mathsf{link}}$ contains no edges. Our simulator is stateful, and maintains a global linking graph G^*, or the union of all the individual query linking graphs. To compute the union of two linking graphs, any vertices sharing the same rid attribute are combined to form one new vertex. The new vertex has the same rid and filterlist set to the union of all the individual filterlist attributes—forming a list of all the filters that output the same record. Then, Synq is secure as stated in the following theorem.

Theorem 7.2. If SHE and PKE are CPA-secure, Σ_{DX} is Λ_{DX}-secure, Σ_{MM} is Λ_{MM}-secure, then Synq is Λ_{Synq}-secure.

The proof sketch of Theorem 7.2 is in Appendix B.

Concrete leakage analysis. Since the leakage profile of Synq depends on the leakage profiles of Σ_{DX} and Σ_{MM}, we now instantiate Σ_{DX} and Σ_{MM} with a standard version of the Π_{bas} scheme [25] to demonstrate the concrete leakage of a potential implementation. (We emphasize that other choices of Σ_{DX} and Σ_{MM} are possible.) Using Π_{bas}, the leakage patterns Λ_{DX}, Λ_{MM} are as follows:

$$\Lambda_{\mathsf{DX}} = (\mathcal{L}^{\mathsf{DX}}_{\mathsf{S}}, \mathcal{L}^{\mathsf{DX}}_{\mathsf{Q}}) = (\mathsf{size}, \mathsf{qeq}),$$

$$\Lambda_{\mathsf{MM}} = (\mathcal{L}_{\mathsf{S}}^{\mathsf{MM}}, \mathcal{L}_{\mathsf{Q}}^{\mathsf{MM}}) = (\mathsf{size}, (\mathsf{qeq}, \mathsf{vol})),$$

where size outputs the total number of values in the data structure, qeq is the query equality pattern (reveals query repetitions), and vol is the volume pattern (reveals the number of results corresponding to a query). Note that Λ_{MM} is content oblivious. Then, Synq's concrete leakage profile is:

- (Init) No information is leaked during initialization.
- (Setup) During setup, the total number of values N for each dictionary and multi-map is leaked. Concretely, the following information is leaked for each owner $\mathbf{P}_i$:
  - N_i, the total number of records in the dataset T_i,
  - $|X_i^{\mathsf{Filter}}|$, the number of filterable columns in T_i.
- (Query) During a query, qeq and vol for all the queried data structures is leaked. Concretely, given the query $q = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$ the leakage is as follows:
  - op, which represents the list of filters, the linking condition, and the aggregations that the query contains,
  - for each filter $(\mathbf{P}_i, x, \mathsf{value})$, $N_i^{x=\mathsf{value}}$, the number of records for data owner $\mathbf{P}_i$ that satisfy the filter,
  - for each record that satisfies at least one filter, the following is leaked: (1) the previous query history of the record, and (2) all *links* to the record for the linking condition L_{link}, where links exist between a pair of records if they have the same values for the columns in the linking condition L_{link}.

Figure 10 provides a visual representation of the concrete leakage profile's output against the running example query from Figure 2. op reveals a per-data owner list of filters, the linking condition, and the aggregate operation of a query. $\mathsf{G}^{\mathsf{link}}$ captures the information revealed to the server during the computation of the filters, linking and aggregates. For every record matched by any filter, this includes the matching filter and the links resulting from the linking condition.

7.2.2. Analyst. Synq guarantees that a semi-honest analyst $\mathbf{Q}$ cannot learn more information than the results of its queries. Intuitively, this holds because the simulator $\mathcal{S}$ receives the (corrupted) $\mathbf{Q}$'s queries from the environment $\mathcal{Z}$, and forwards them to the ideal functionality to receive the results. $\mathcal{S}$ then sends the results (encrypted with $\mathbf{Q}$'s public key) to $\mathbf{Q}$. Then, $\mathbf{Q}$'s view in both worlds are identical. We defer the security proof to the full version of our paper. We note that, for some datasets, $\mathbf{Q}$ can potentially learn fine-grained information about the plaintext datasets even when only using aggregate queries. This is an orthogonal data privacy concern present in any system that supports analytics over encrypted data. This concern could be addressed using system-level mitigations such as query auditing tools or rate limits (see [89] for more detailed limitation procedures). Data-level techniques such as differential privacy [36] can also be used with Synq without changing the protocol or existing security guarantees, though we note that data privacy with respect to analysts is not always a requirement even in real-world deployments of privacy-preserving systems (e.g., the Boston study computed exact aggregates [67]). In the MA DPH setting, analysts are trusted and can execute any queries of their choice over the input datasets.

7.2.3. Implications of Leakage. The leakage of a system expresses the information revealed to an adversary as a function of the plaintext data and queries. Any efficient encrypted system, even systems that are based on FHE or MPC, will reveal some leakage about the underlying data and queries. This leakage might be small (such as the size of the plaintext) or large (such as DTE's leakage which reveals all the correlations present in the plaintext). Making the leakage function explicit allows us to express trade-offs between functionality and security. However, a leakage function only describes the leakage—it does not describe if the leakage can be exploited effectively or if the leakage is appropriate for the application setting. In this section, we will describe prior work on exploiting leakage and explain our rationale for why Synq's leakage might present a reasonable trade-off between functionality and efficiency in the public policy setting.

Leakage attacks. Leakage attacks, where the adversary exploits the leakage profile of an STE scheme in an attempt to reconstruct information about the data or the queries, have received significant attention in the literature. Islam et al. [53] were the first to investigate leakage attacks. More recent works (e.g., [16], [24], [48]–[50], [62]–[64], [70], [71], [75]) have relaxed assumptions and targeted other profiles. While these attacks have clear theoretical interest, [55] shows their practical impact varies and an attack's assumptions must be carefully analyzed in the context of a particular system.

Comparison to prior work. Any system supporting aggregate queries over multiple datasets has to choose a trade-off between functionality and security. From Table 2, the Boston wage equity study [67], Jana [51], and i2b2 [80] most closely satisfy our usability and expressivity considerations. Each of these works makes a different trade-off. The Boston study uses a fixed schema and protocol to enable computation of an average, revealing only the sizes of the underlying tables. However, their system can only be used for a single aggregate. Conversely, supporting multiple schemas and repeated complex aggregates inevitably leads to greater leakage. In particular, the linking of records will always leak some information to the linking server—short of using FHE and a prohibitive amount of computation. Jana uses either outsourced MPC or PPE, depending on the query. Outsourced MPC leverages interactive computation to reduce leakage to any individual compute server. However, the sizes of the plaintext are still leaked, and if the adversary corrupts more than a threshold of servers, it learns Jana's plaintext data immediately. In contrast, even if the non-collusion assumption in Synq (between the server and the OPRF service, see 6.1.1) is violated, the adversary never directly learns the plaintext data in the absence of auxiliary information. The PPE-based approach allows Jana to perform more efficient queries, but also leaks correlations in the data to any adversary with access to the server. Further, Jana only supports linking using plaintext data or DTE, which reveals all the links to the server at setup time. i2b2 only supports secure sums, and performs filtering over plaintext data. In

$$\mathcal{L}_{\mathsf{I}} = \perp$$

$$\mathcal{L}_{\mathsf{S}} = \left\{ \begin{array}{l} \mathbf{P}_1 : \text{3 records, 2 filterable columns} \\ \mathbf{P}_2 : \text{3 records, 2 filterable columns} \end{array} \right\}$$

$$\mathcal{L}_{\mathsf{Q}} = \big(\mathsf{op}(q), \mathsf{G}^{\mathsf{link}}\big), \text{ where}$$

$$\mathsf{op}(q) = \Big(\big[(\mathbf{P}_1 : f_1, f_2), (\mathbf{P}_2 : f_3, f_4)\big], L_1, \big[(\mathsf{TableCount}, T)\big]\Big)$$

and $\mathsf{G}^{\mathsf{link}}$ is shown on the right

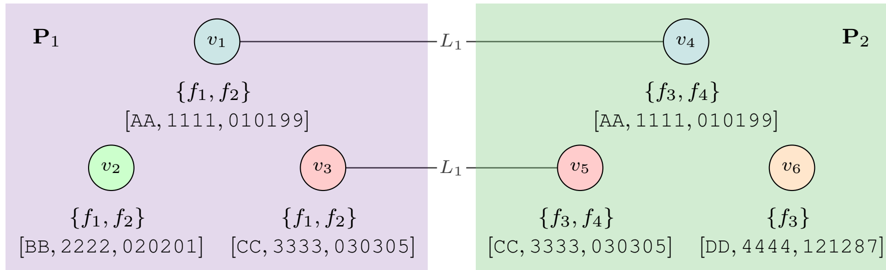


Figure 10. Running example for the concrete leakage generated by the setup process illustrated in Figure 6 and Figure 7 and the query q from Figure 2. The `[data records]` shown in $\mathsf{G}^{\mathsf{link}}$ are for illustrative purposes and are not actually included in the leakage.

Synq, we use SHE to support aggregates, and an OPRF to support linking. Our use of pseudorandom tags and revealing a subset of them at query time is a reasonable compromise between full link leakage and prohibitive computation.

Our rationale. We now explain why the design of Synq and the context in which it is used allows for security better than or equivalent to existing solutions. In the context of the Boston study, the requirement was to compute a one-time average, given a standardized schema. Synq could be used to carry out the same one-time average computation. Both the Boston study and Synq reveal the sizes of the underlying tables and intermediate sums to the analyst, and since linking is not required, Synq does not have any additional leakage. However, Synq would also allow for re-computation of these averages with different filters, whereas the Boston study would need to perform setup again for any further computation. When compared to the trusted server solution used for MA DPH, Synq only reveals a small, well-defined leakage to a semi-honest server. If the server $\mathbf{S}$ and up to $n-1$ data owners are corrupted, the server learns the leakage Λ_{Synq} during the protocol execution and the data owners learn nothing more than their own inputs.

Impact of leakage. First, we consider what can be learned from the leakage profile Λ_{Synq} alone. Λ_{Synq} is a function of columns in the pre-specified linking conditions $\mathsf{L} = [\perp, L_1, \ldots, L_m]$ and in X^{Filter}. Specifically, $\mathsf{G}^{\mathsf{link}}$ is a function of columns in $\mathsf{L} = [\perp, L_1, \ldots, L_m]$ and the queries, and $\mathsf{patt}_{\mathsf{Q}}^{\mathsf{DX}}(\mathsf{DX}_j^{\mathsf{data}})$, $\mathsf{patt}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}_j^{\mathsf{filter}})$, $\mathsf{patt}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}_j^{\mathsf{link}})$ is a function of X^{Filter} and the queries. Given this observation as well as the CPA-security of the SHE scheme, we know that data contained in columns that are not in L or X^{Filter} cannot be reconstructed. In the context of public policy studies, inference attacks (i.e., attacks that require distributional knowledge of data and/or queries) could also be a concern since the data could come from known distributions (e.g., demographic data from certain geographical regions). We stress, however, that because Synq makes use of standard STE primitives, these attacks could (potentially) only be executed by a *persistent* adversary that has access to the server throughout the query execution. On the other hand, PPE-based solutions are prone to inference attacks even by a *single-snapshot* adversary that only sees the encrypted data once and does not have access to queries. Finally, Synq also reveals the linking condition and the aggregate query to the adversary. Since the linking conditions are public and, in many studies, the aggregate query is ultimately revealed due to study publication, we believe both components are acceptable to leak given Synq's public policy context.

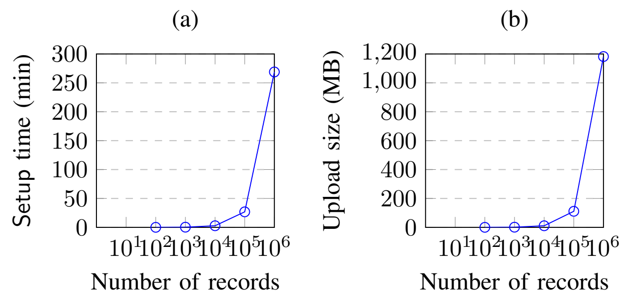


Figure 11. (a) Setup time for an individual data owner. (b) Total size of EDB and keys of individual owners in Setup.

8. Empirical Evaluation

In this section, we describe our prototype implementation and evaluation of Synq. The source code (which is written in Python 3.10) can be found at `https://github.com/encryptedsystems/synq`.

Cryptographic primitives. The Synq protocol makes black-box use of several cryptographic primitives. We instantiate SHE with the implementation of the CKKS scheme for approximate somewhat homomorphic encryption [28] from Tenseal 0.3.1 [15]. We instantiate Σ_{DX} and Σ_{MM} with response-revealing variants of the `SimpleEDX` and `PiBaseEMM` schemes from the Arca 0.1 package [37], [38]. Both schemes are implementations of the Π_{bas} scheme by Cash et al. [25] for dictionaries and multi-maps respectively. For symmetric encryption and PRFs within Σ_{DX} and Σ_{MM}, we use AES-256 and SHA-512 respectively from Arca's provided cryptographic primitives. For PKE, we similarly use Arca's primitives for RSA-2048. Finally, we instantiate the $\mathcal{F}_{\mathsf{OPRF}}$ functionality using the `oprf` 3.0.0 package [65].

Environment. We conducted our experiments locally on a 2021 MacBook Pro (macOS Monterey 12.3.1, M1 Max chip, 64 GB of memory). The main server and OPRF server ran persistently as background processes. Clients interacted with these servers via localhost gRPC connections. We measured the time of each component by running the experiment 5 times and reporting the average time over 5 runs.

Aggregate	Token size (bytes)	Query time (s)
Sum	81	5.62
Average	81	5.69
Count	66	4.31
Variance	81	139
Regression (1-col)	106	282
Regression (2-col)	116	645

TABLE 3. Average execution time and increase in query token size for aggregate functions based on the instantiations in Appendix A, assuming two filters and a linked table with 10^5 records from two data owners.

Datasets. Obtaining real-world datasets with PII for linking is challenging due to privacy concerns, licensing costs, and is likely inadvisable due to the privacy risks and consent issues that would be involved. We generated several healthcare-themed example datasets using the `faker` package [40]. We used `faker` to generate a universe of 1 million "people", each with up to 15 randomly generated attributes (such as first name, data of birth, social security number, etc.). Then, we randomly assigned between 7 and 15 columns to each of the datasets. From there, we saved between 25% and 100% of the people in each of the datasets. This allowed us to capture substantial links between records in datasets (as observed in the MA DPH report) while also allowing us to demonstrate that we could perform operations over columns that did not appear in multiple datasets.

For Setup, we assigned columns in Synq based on the semantic meaning of the data we generated. This resulted in experiments with 1 linking condition (consisting of 5 columns), 6 numerical columns (X^{Num}), and 7 filterable columns (X^{Filter}). For our Query experiments, we tweaked our data generation such that the linked table consistently included 10^5 records so that differences in the number of linked records would not impact our observations.

Summary of results. To summarize our evaluation:

- Our Setup algorithm is the most computationally expensive part of the protocol. Figure 11(a) shows that, at 1 million records, Setup takes just under 4.5 hours of compute time. About 87% of Setup time (Figure 11) was spent in step (2) which computes several SHE-encrypted values and populates $\mathsf{DX}^{\mathsf{data}}$ (prior to encryption).
- The encrypted structures result in a 7.61× size increase over the size of the plaintext CSV (at 1 million records, the plaintext is 155MB and the encrypted Synq structures are 1182MB, as shown in Figure 11(b)).
- The query bandwidth is small. Each aggregate function adds at most 116 bytes to the token (Table 3). Each filter adds between 19 and 22 bytes to the query token size. As expected, the query size scales linearly according to the number of filters and aggregate functions in the query, and does so relatively consistently.
- There is a performance gap between functions that do not involve SHE multiplications (Sum, Average, Count) and those that do (Variance, Regression). As shown in Table 3, while Sum, Average, and Count queries took at most 6 seconds to complete, Variance and Regression queries took at least 2 minutes (and at most 11 minutes) to complete. For those latter queries, approximately 81% of the Query time is spent in the JoinMultiply handler.

We did not perform system-level optimization of our prototype. For instance, the clients are single-threaded but the Π_{bas} scheme (used for Σ_{MM} and Σ_{DX}) and other parts of Setup are parallelizable. To focus on Synq's design and methodology, we defer system optimizations to future work.

Full version. Our paper's full version describes additional functionality extensions to Synq and details on the query survey conducted on the MA DPH report.

Acknowledgements

The authors would like to thank Brown's Data Science Institute [2] for providing initial funding for this project, Kevin Wilson and David Yokum from Brown's Policy Lab [3] for the discussions that led to this project, and Zheguang Zhao for their help in the initial stages of the project. Zachary Espiritu was supported in part by the Randy Pausch Undergraduate Summer Research Award at Brown University. Marilyn George would like to thank Nishanth Chandran, Divya Gupta, and her other collaborators at Microsoft Research India for early discussions about analytics on encrypted data. Lucy Qin was supported by the Graduate Research Fellowship Program funded by the National Science Foundation, and would like to thank her previous team at Boston University and the Boston Women's Workforce Council for experiences and insights in deploying real-world cryptographic tools that have impacted this work.

References

[1] "Center for health information and analysis." [Online]. Available: https://www.chiamass.gov/

[2] "Data science institute at brown." [Online]. Available: https://dsi.brown.edu/

[3] "The policy lab at brown university." [Online]. Available: https://thepolicylab.brown.edu/

[4] "Boston women's workforce council," 2023. [Online]. Available: https://thebwwc.org/

[5] S. Addanki, K. Garbe, E. Jaffe, R. Ostrovsky, and A. Polychroniadou, "Prio+: Privacy preserving aggregate statistics via boolean shares," in *Security and Cryptography for Networks*, C. Galdi and S. Jarecki, Eds. Cham: Springer International Publishing, 2022, pp. 516–539.

[6] A. Agarwal, S. Peceny, M. Raykova, P. Schoppmann, and K. Seth, "Communication-efficient secure logistic regression," Cryptology ePrint Archive, Paper 2022/866, 2022.

[7] E. H. Allen, H. Samuel-Jakubos, and T. A. Waidmann, "Data sharing in cross-sector collaborations," The Urban Institute, Tech. Rep., Jul. 2021. [Online]. Available: https://www.urban.org/research/publication/data-sharing-cross-sector-collaborations

[8] Antonis Papadimitriou, Nishanth Chandran, Ranjita Bhagwan, Ramachandran Ramjee, Andreas Haeberlen, Harmeet Singh, Abhishek Modi, and Saikrishna Badrinarayanan, "Big Data Analytics over Encrypted Datasets with Seabed," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16. Savannah, GA: USENIX Association, Nov. 2016.

[9] Apple and Google, "Exposure notification privacy-preserving analytics (enpa) white paper," Tech. Rep., 4 2021. [Online]. Available: https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ENPA_White_Paper.pdf

[10] A. Arasu, S. Blanas, K. Eguro, M. Joglekar, R. Kaushik, D. Kossmann, R. Ramamurthy, P. Upadhyaya, and R. Venkatesan, "Secure database-as-a-service with cipherbase," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1033–1036.

[11] D. Archer, A. O'Hara, R. Issa, and S. Straus, "Sharing sensitive department of education data across organizational boundaries using secure multiparty computation," Galois, Inc. and Georgetown University, Tech. Rep., 5 2021. [Online]. Available: https://github.com/Ra1issa/ra1issa-website/blob/main/NCES_Demo_Paper_technical.pdf

[12] D. W. Archer, D. Bogdanov, Y. Lindell, L. Kamm, K. Nielsen, J. I. Pagter, N. P. Smart, and R. N. Wright, "From Keys to Databases—Real-World Applications of Secure Multi-Party Computation," *The Computer Journal*, vol. 61, no. 12, pp. 1749–1771, 09 2018.

[13] L. Bangalore, M. H. F. Sereshgi, C. Hazay, and M. Venkitasubramaniam, "Flag: A framework for lightweight robust secure aggregation," in *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 14–28.

[14] J. Bater, G. Elliott, C. Eggen, S. Goel, A. Kho, and J. Rogers, "Smcql: Secure querying for federated databases," *Proc. VLDB Endow.*, vol. 10, no. 6, p. 673–684, feb 2017.

[15] A. Benaissa, B. Retiat, B. Cebere, and A. E. Belfedhal, "Tenseal: A library for encrypted tensor operations using homomorphic encryption," 2021.

[16] L. Blackstone, S. Kamara, and T. Moataz, "Revisiting Leakage Abuse Attacks," in *27th Annual Network and Distributed System Security Symposium (NDSS 2020)*. San Diego, California, USA: The Internet Society, 2020.

[17] D. Bogdanov, "Sharemind: programmable secure computations with practical applications," Ph.D. dissertation, University of Tartu, 2013.

[18] D. Bogdanov, L. Kamm, B. Kubo, R. Rebane, V. Sokk, and R. Talviste, "Students and taxes: a privacy-preserving study using secure computation," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 117–135, 2016.

[19] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1175–1191.

[20] P. Borrello, A. Kogler, M. Schwarzl, M. Lipp, D. Gruss, and M. Schwarz, "ÆPIC leak: Architecturally leaking uninitialized data from the microarchitecture," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 3917–3934.

[21] L. Burkhalter, A. Hithnawi, A. Viand, H. Shafagh, and S. Ratnasamy, "TimeCrypt: Encrypted data stream processing at scale with cryptographic access control," in *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. Santa Clara, CA: USENIX Association, Feb. 2020, pp. 835–850.

[22] L. Burkhalter, N. Küchler, A. Viand, H. Shafagh, and A. Hithnawi, "Zeph: Cryptographic enforcement of end-to-end data privacy," in *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*. USENIX Association, Jul. 2021, pp. 387–404.

[23] R. Canetti, "Security and Composition of Multiparty Cryptographic Protocols," *Journal of Cryptology*, vol. 13, no. 1, Jan. 2000.

[24] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-Abuse Attacks Against Searchable Encryption," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. Association for Computing Machinery, 2015.

[25] D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Dynamic searchable encryption in very-large databases: Data structures and implementation," in *21st Annual Network and Distributed System Security Symposium, NDSS 2014*. San Diego, California, USA: The Internet Society, 2 2014.

[26] M. Chase and S. Kamara, "Structured encryption and controlled disclosure," in *Advances in Cryptology - ASIACRYPT '10*, ser. Lecture Notes in Computer Science, vol. 6477. Springer, 2010, pp. 577–594.

[27] F. Chen, S. Wang, X. Jiang, S. Ding, Y. Lu, J. Kim, S. C. Sahinalp, C. Shimizu, J. C. Burns, V. J. Wright, E. Png, M. L. Hibberd, D. D. Lloyd, H. Yang, A. Telenti, C. S. Bloss, D. Fox, K. Lauter, and L. Ohno-Machado, "PRINCESS: Privacy-protecting rare disease international network collaboration via encryption through software guard extensions," *Bioinformatics*, pp. 871–878, Mar. 2017.

[28] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology – ASIACRYPT 2017*, T. Takagi and T. Peyrin, Eds. Cham: Springer International Publishing, 2017, pp. 409–437.

[29] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. Boston, MA: USENIX Association, Mar. 2017, pp. 259–282.

[30] J. Cui, J. Z. Yu, S. Shinde, P. Saxena, and Z. Cai, "Smashex: Smashing sgx enclaves using exceptions," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21, 2021, p. 779–793.

[31] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *ACM Conference on Computer and Communications Security (CCS '06)*. ACM, 2006, pp. 79–88.

[32] Cybernetica, "Mobile phone data meets Sharemind HI: tourism statistics innovation in Indonesia," Cybernetica, Tech. Rep., 6 2019. [Online]. Available: https://sharemind.cyber.ee/mobile-phone-data-meets-sharemind-hi/

[33] Datavant, "Overview of datavant's de-identification and linking technology for structured data," Datavant, Tech. Rep., 8 2018. [Online]. Available: https://datavant.com/wp-content/uploads/dlm_uploads/2018/09/WhitePaper_-De-Identifying-and-Linking-Structured-Data.pdf

[34] X. Dong, D. A. Randolph, C. Weng, A. N. Kho, J. M. Rogers, and X. Wang, "Developing High Performance Secure Multi-Party Computation Protocols in Healthcare: A Case Study of Patient Risk Stratification," *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science*, vol. 2021, 2021.

[35] Dutch Research Council, "Coronary ARtery disease: Risk estimations and Interventions for prevention and EaRly detection – a Personal Health Train project," Dutch Research Council, Tech. Rep., 2020.

[36] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*, M. Agrawal, D. Du, Z. Duan, and A. Li, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19.

[37] Z. Espiritu, "Arca," GitHub, 2022. [Online]. Available: https://github.com/cloudsecuritygroup/arca

[38] Z. Espiritu, E. A. Markatou, and R. Tamassia, "Time- and space-efficient aggregate range queries over encrypted databases," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, no. 4, 2022.

[39] European Commission and Joint Research Centre, D. Ramírez, L. Díaz, S. Rahimian, J. García, B. Peña, Y. Al-Khazraji, A. Alarcón, P. Fuente, J. Soler Garrido, and A. Kotsev, *Technological enablers for privacy preserving data sharing and analysis – A comparative study*. Publications Office of the European Union, 2023.

[40] D. Faraglia and Other Contributors, "Faker." [Online]. Available: https://github.com/joke2k/faker

[41] Fast-Track Action Committee on Advancing Privacy-Preserving Data Sharing and Analytics Networking and Information Technology Research and Development Subcommittee, "National strategy to advance privacy-preserving data sharing and analytics," National Science and Technology Council, Tech. Rep., 3 2023.

[42] S. Fei, Z. Yan, W. Ding, and H. Xie, "Security vulnerabilities of sgx and countermeasures: A survey," *ACM Comput. Surv.*, vol. 54, no. 6, jul 2021.

[43] M. J. Freedman, Y. Ishai, B. Pinkas, and O. Reingold, "Keyword Search and Oblivious Pseudorandom Functions," in *Theory of Cryptography*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and J. Kilian, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, vol. 3378.

[44] D. Froelicher, P. Egger, J. S. Sousa, J. L. Raisaro, Z. Huang, C. Mouchet, B. Ford, and J.-P. Hubaux, "Unlynx: A decentralized system for privacy-conscious data sharing," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 4, pp. 232–250, 2017.

[45] D. Froelicher, J. R. Troncoso-Pastoriza, J. L. Raisaro, M. A. Cuendet, J. S. Sousa, H. Cho, B. Berger, J. Fellay, and J.-P. Hubaux, "Truly privacy-preserving federated analytics for precision medicine with multiparty homomorphic encryption," *Nature Communications*, vol. 12, no. 1, p. 5910, 10 2021.

[46] B. Fuller, D. Mitchell, R. Cunningham, U. Blumenthal, P. Cable, A. Hamlin, L. Milechin, M. Rabe, N. Schear, R. Shay, M. Varia, and S. Yakoubov, "SPAR pilot evaluation," MIT Lincoln Laboratory, Lexington, MA, USA, Tech. Rep., 11 2015.

[47] M. George, S. Kamara, and T. Moataz, "Structured encryption and dynamic leakage suppression," in *Advances in Cryptology – EUROCRYPT 2021*, A. Canteaut and F.-X. Standaert, Eds. Cham: Springer International Publishing, 2021, pp. 370–396.

[48] R. Groot Roessink, A. Peter, and F. Hahn, "Experimental review of the ikk query recovery attack: Assumptions, recovery rate and improvements," in *Applied Cryptography and Network Security*, K. Sako and N. O. Tippenhauer, Eds. Cham: Springer International Publishing, 2021, pp. 155–183.

[49] P. Grubbs, M.-S. Lacharite, B. Minaud, and K. G. Paterson, "Pump up the volume: Practical database reconstruction from volume leakage on range queries," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018.

[50] P. Grubbs, M.-S. Lacharité, B. Minaud, and K. G. Paterson, "Learning to reconstruct: Statistical learning theory and encrypted database attacks," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1067–1083.

[51] N. R. Hart, D. W. Archer, and E. Dalton, "Privacy-preserved data sharing for evidence-based policy decisions: A demonstration project using human services administrative records for evidence-building activities," Bipartisan Policy Center, Tech. Rep., March 2019.

[52] M. Ion, B. Kreuter, A. E. Nergiz, S. Patel, S. Saxena, K. Seth, M. Raykova, D. Shanahan, and M. Yung, "On Deploying Secure Computing: Private Intersection-Sum-with-Cardinality," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. Genoa, Italy: IEEE, Sep. 2020.

[53] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access Pattern Disclosure on Searchable Encryption: Ramification, Attack and Mitigation," in *19th Annual Network and Distributed System Security Symposium (NDSS 2012)*. San Diego, California, USA: The Internet Society, 2012.

[54] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konevcný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," 2019. [Online]. Available: https://arxiv.org/abs/1912.04977

[55] S. Kamara, A. Kati, T. Moataz, T. Schneider, A. Treiber, and M. Yonli, "Sok: Cryptanalysis of encrypted search with leaker – a framework for leakage attack evaluation on real-world data," in *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, 2022, pp. 90–108.

[56] S. Kamara and T. Moataz, "SQL on Structurally-Encrypted Databases," in *Advances in Cryptology – ASIACRYPT 2018*. Cham: Springer International Publishing, 2018, vol. 11272.

[57] S. Kamara, T. Moataz, and O. Ohrimenko, "Structured encryption and leakage suppression," in *Advances in Cryptology – CRYPTO 2018*, H. Shacham and A. Boldyreva, Eds. Cham: Springer International Publishing, 2018, pp. 339–370.

[58] S. Kamara, T. Moataz, A. Park, and L. Qin, "A decentralized and encrypted national gun registry," in *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*. IEEE, 2021, pp. 1520–1537.

[59] S. Kamara, T. Moataz, S. Zdonik, and Z. Zhao, "An optimal relational database encryption scheme," Cryptology ePrint Archive, Paper 2020/274, 2020.

[60] G. Kaptchuk, M. Green, and A. Rubin, "Outsourcing Medical Dataset Analysis: A Possible Solution," in *Financial Cryptography and Data Security*. Springer International Publishing, 2017, vol. 10322.

[61] J. Katz and Y. Lindell, *Introduction to modern cryptography*, 3rd ed. Boca Raton, FL: CRC Press, 2020.

[62] G. Kellaris, G. Kollios, K. Nissim, and A. O'Neill, "Generic attacks on secure outsourced databases," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1329–1340.

[63] E. M. Kornaropoulos, C. Papamanthou, and R. Tamassia, "The state of the uniform: Attacks on encrypted databases beyond the uniform query distribution," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 1223–1240.

[64] ——, "Response-hiding encrypted ranges: Revisiting security via parametrized leakage-abuse attacks," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021.

[65] A. Lapets, "oprf," Nth Party, Ltd., Apr 2022. [Online]. Available: https://pypi.org/project/oprf/

[66] A. Lapets, K. Dak Albab, R. Issa, L. Qin, M. Varia, A. Bestavros, and F. Jansen, "Role-based ecosystem for the design, development, and deployment of secure multi-party data analytics applications," in *2019 IEEE Cybersecurity Development (SecDev)*, 2019, pp. 129–140.

[67] A. Lapets, F. Jansen, K. D. Albab, R. Issa, L. Qin, M. Varia, and A. Bestavros, "Accessible Privacy-Preserving Web-Based Data Analysis for Assessing and Addressing Economic Inequalities," in *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*. ACM, Jun. 2018.

[68] M. Lipp, A. Kogler, D. Oswald, M. Schwarz, C. Easdon, C. Canella, and D. Gruss, "Platypus: Software-based power side-channel attacks on x86," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 355–371.

[69] Lucy Qin, Andrei Lapets, Frederick Jansen, Peter Flockhart, Kinan Dak Albab, Ira Globus-Harris, Shannon Roberts, and Mayank Varia, "From Usability to Secure Computing and Back Again," in *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*. Santa Clara, CA: USENIX Association, Aug. 2019.

[70] E. A. Markatou, F. Falzon, Z. Espiritu, and R. Tamassia, "Attacks on encrypted response-hiding range search schemes in multiple dimensions," *Proceedings on Privacy Enhancing Technologies*, vol. 2023, no. 4, pp. 204–223, 2023.

[71] E. A. Markatou and R. Tamassia, "Full database reconstruction with access and search pattern leakage," in *Information Security*, Z. Lin, C. Papamanthou, and M. Polychronakis, Eds. Cham: Springer International Publishing, 2019, pp. 25–43.

[72] Massachusetts Department of Public Health, "An Assessment of Fatal and Nonfatal Opioid Overdoses in Massachusetts (2011-2015)," Massachusetts Department of Public Health, Tech. Rep., Aug. 2017. [Online]. Available: https://www.mass.gov/files/documents/2017/08/31/legislative-report-chapter-55-aug-2017.pdf

[73] M. Naveed, S. Kamara, and C. V. Wright, "Inference Attacks on Property-Preserving Encrypted Databases," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. Denver, Colorado, USA: Association for Computing Machinery, 2015, pp. 644–655.

[74] R. Ng, A. Hoover, D. Cash, and E. Ee, "Structured encryption for indirect addressing," Cryptology ePrint Archive, Paper 2023/1146, 2023.

[75] S. Oya and F. Kerschbaum, "Hiding the Access Pattern is Not Enough: Exploiting Search Pattern Leakage in Searchable Encryption," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021.

[76] V. Pappas, F. Krell, B. Vo, V. Kolesnikov, T. Malkin, S. Choi, W. George, A. Keromytis, and S. Bellovin, "Blind seer: A scalable private dbms," in *2014 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, may 2014, pp. 359–374.

[77] R. Poddar, S. Kalra, A. Yanai, R. Deng, R. A. Popa, and J. M. Hellerstein, "Senate: A Maliciously-Secure MPC platform for collaborative analytics," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 2129–2146.

[78] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: Protecting confidentiality with encrypted query processing," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, ser. SOSP '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 85–100.

[79] L. Qin, F. Jansen, K. Bab, T. Braun, and ignoramous, "multiparty/oprf," https://github.com/multiparty/oprf/releases/tag/v2.0.0, 2013.

[80] J. L. Raisaro, J. G. Klann, K. B. Wagholikar, H. Estiri, J.-P. Hubaux, and S. N. Murphy, "Feasibility of homomorphic encryption for sharing I2B2 aggregate-level data in the cloud," *AMIA Summits Transl. Sci. Proc.*, vol. 2017, pp. 176–185, May 2018.

[81] J. L. Raisaro, J. R. Troncoso-Pastoriza, M. Misbach, J. S. Sousa, S. Pradervand, E. Missiaglia, O. Michielin, B. Ford, and J.-P. Hubaux, "Medco: Enabling secure and privacy-preserving exploration of distributed clinical and genomic data," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 16, no. 4, p. 1328–1341, jul 2019.

[82] M. Rathee, C. Shen, S. Wagh, and R. Popa, "Elsa: Secure aggregation for federated learning with malicious actors," in *2023 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2023, pp. 1961–1979.

[83] J. Rogers, E. Adetoro, J. Bater, T. Canter, D. Fu, A. Hamilton, A. Hassan, A. Martinez, E. Michalski, V. Mitrovic, F. Rachman, R. Shah, M. Sterling, K. VanDoren, T. L. Walunas, X. Wang, and A. Kho, "Vaultdb: A real-world pilot of secure multi-party computation within a clinical research network," 2022.

[84] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, and S. Bakas, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific Reports*, vol. 10, no. 1, p. 12598, Jul 2020.

[85] United Nations Committee of Experts on Big Data and Data Science for Official Statistics, "United Nations Guide on Privacy-Enhancing Technologies for Official Statistics," United Nations, New York, USA, Tech. Rep., 2023. [Online]. Available: https://unstats.un.org/bigdata/task-teams/privacy/guide/

[86] F. van Daalen, L. Ippel, A. Dekker, and I. Bermejo, "Privacy preserving nn-party scalar product protocol," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 4, pp. 1060–1066, 2023.

[87] N. Volgushev, M. Schwarzkopf, B. Getchell, M. Varia, A. Lapets, and A. Bestavros, "Conclave: Secure multi-party computation on big data," in *Proceedings of the Fourteenth EuroSys Conference 2019*, ser. EuroSys '19. New York, NY, USA: Association for Computing Machinery, 2019.

[88] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, "Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1342–1397, 2021.

[89] J. M. Walsh, M. Varia, A. Cohen, A. Sellars, and A. Bestavros, "Multi-regulation computing: Examining the legal and policy questions that arise from secure multiparty computation," in *Proceedings of the 2022 Symposium on Computer Science and Law*, ser. CSLAW '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 53–65.

[90] J. Wang and S. S. M. Chow, "Omnes pro uno: Practical Multi-Writer encrypted database," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, aug 2022, pp. 2371–2388.

[91] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, Jan 2019.

Appendix A. Examples of Aggregate Composition

Our scheme supports Sum, Count, $\mathsf{Average}$, $\mathsf{Variance}$, linear $\mathsf{Regression}$, and multiple $\mathsf{Regression}$, where:

- (Sum, x) outputs the sum of values in x.
- (Count) outputs the total number of records in the linked table after filtering and linking.
- $(\mathsf{Average}, x)$ outputs the average of values in x.
- $(\mathsf{Variance}, x)$ outputs the variance of values in x.
- $(\mathsf{Regression}, y, x)$ outputs a linear regression where y is the dependent variable and x is an independent variable.
- $(\mathsf{Regression}, y, x_1, x_2)$ outputs a multiple or binary logistic regression where y is the dependent variable and x_1, x_2 are independent variables.

Each aggregate function is instantiated using the base operators (Definition 5.1) as follows:

Sum. $(\mathsf{ColumnSum}, T, x)$.

Count. $(\mathsf{TableCount}, T)$.

Average. $\mathsf{avg}(x) = (\mathsf{ColumnSum}, T, x)/(\mathsf{TableCount}, T)$.

Variance. The variance of column x of some table T is defined as follows, where n is the number of values in x:

$$\mathsf{var}(x) = \frac{1}{n}\sum_{\mathbf{r}\in T}(\mathbf{r}[x] - \mathsf{avg}(x))^2 = \frac{1}{n}\left(\sum_{\mathbf{r}\in T}\mathbf{r}[x]^2\right) - \mathsf{avg}(x)^2$$

The server computes n using $\mathsf{TableCount}$, $\mathsf{avg}(x)$ using $\mathsf{Average}$, and $\sum_{\mathbf{r}\in T}\mathbf{r}[x]^2 \leftarrow (\mathsf{ColumnSum}, T, (\mathsf{JoinMultiply}, T, x, x))$. Finally, given the values n, $\sum \mathbf{r}[x]^2$, and $\bar{x}$, the analyst computes the variance of the column x.

Linear Regression. A linear regression for an independent column x and dependent column y is defined by the formula: $y = b_1 x + b_0$. Let both the columns x and y belong to some table T. If both x and y contain n values, the coefficients b_0 and b_1 are defined as follows:

$$b_1 = \frac{n \sum_{\mathbf{r}\in T} \mathbf{r}[x] \cdot \mathbf{r}[y] - \left(\sum_{\mathbf{r}\in T} \mathbf{r}[x]\right)\left(\sum_{\mathbf{r}\in T} \mathbf{r}[y]\right)}{n \sum_{\mathbf{r}\in T} \mathbf{r}[x]^2 - \left(\sum_{\mathbf{r}\in T} \mathbf{r}[x]\right)^2}$$

$$b_0 = \frac{\sum_{\mathbf{r}\in T} \mathbf{r}[y] - b_1 \left(\sum_{\mathbf{r}\in T} \mathbf{r}[x]\right)}{n}$$

The server computes the sums $\sum \mathbf{r}[x], \sum \mathbf{r}[y]$ using $\mathsf{ColumnSum}$, n using $\mathsf{TableCount}$, and $\sum \mathbf{r}[x]^2$ using $\mathsf{JoinMultiply}$, as shown in the description for computing the variance. The server computes $\sum \mathbf{r}[x] \cdot \mathbf{r}[y]$ as follows:

$$\sum_{\mathbf{r}\in T} \mathbf{r}[x] \cdot \mathbf{r}[y] \leftarrow (\mathsf{ColumnSum}, T, (\mathsf{JoinMultiply}, T, x, y)).$$

If the columns x and y belong to different tables, say T_1 and T_2, the table T is replaced by the linked table $T_1 \bowtie T_2$ for all the aggregate computations. Finally, given the values of $\sum \mathbf{r}[x], \sum \mathbf{r}[y], \sum \mathbf{r}[x]^2, \sum \mathbf{r}[x] \cdot \mathbf{r}[y]$, and n, the analyst computes the regression coefficients b_0 and b_1.

Multivariate Regression. A multivariate regression for independent columns $x_1, ..., x_p$ and dependent column y is defined by $y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p$, such that, $b^T = [b_0, b_1, \ldots, b_{p-1}] = \left(X^T X\right)^{-1} X^T Y$, where X is the matrix such that the i^{th} column contains the values of column x_i and Y is the vector contains the values of column y. Let the column x_i belong to table T_i, and the column y belong to table T. For simplicity, assume that all the columns contain n values, and that all the records are linked. In the following, we overload notation and write $\sum_T x_1$ to denote $\sum_{\mathbf{r}\in T} \mathbf{r}[x_1]$.

For example, in the case of two independent variables, the coefficients b_0, b_1, b_2 are computed as follows:

$$b_0 = \frac{\sum_{\mathbf{r}\in T} y - b_1 \left(\sum_{\mathbf{r}\in T} x_1\right) - b_2 \left(\sum_{\mathbf{r}\in T} x_2\right)}{n}$$

$$b_1 = \frac{\sum_{T_2} x_2^2 \cdot \sum_{T_1 \bowtie T} x_1 y - \sum_{T_1 \bowtie T_2} x_1 x_2 \cdot \sum_{T_2 \bowtie T} x_2 y}{\sum_{T_1} x_1^2 \cdot \sum_{T_2} x_2^2 - \left(\sum_{T_1 \bowtie T_2} x_1 x_2\right)^2}$$

$$b_2 = \frac{\sum_{T_1} x_1^2 \cdot \sum_{T_2 \bowtie T} x_2 y - \sum_{T_1 \bowtie T_2} x_1 x_2 \cdot \sum_{T_1 \bowtie T} x_1 y}{\sum_{T_1} x_1^2 \cdot \sum_{T_2} x_2^2 - \left(\sum_{T_1 \bowtie T_2} x_1 x_2\right)^2}$$

The computation of a multivariate regression then requires the sums $\sum \mathbf{r}[x_i]$, $\sum \mathbf{r}[x_i] \cdot \mathbf{r}[x_j]$, $\sum \mathbf{r}[x_i] \cdot \mathbf{r}[y]$, $\sum \mathbf{r}[x_i]^2$ and n, which are computed as in the case of linear regressions. Similarly, the server uses $\mathsf{JoinMultiply}$ and $\mathsf{ColumnSum}$ to compute these values. Given the aggregates, the analyst can compute the coefficients of the multivariate regression.

Binary Logistic Regression. A binary logistic regression for independent variables $x_1, \ldots, x_p$ and dependent variable y is described by the following formula $y = \frac{e^{b_0 + b_1 x_1 + \cdots + b_p x_p}}{1 + e^{b_0 + b_1 x_1 + \cdots + b_p x_p}}$. The coefficients b_i are derived using the same formula as for linear and multivariate regression, and therefore, binary logistic regressions are not explicitly labeled as an aggregate function but encompassed by $\mathsf{Regression}$. After the intermediate sums are computed, similar to the linear and multivariate regressions, the analyst computes the coefficients b_i of the binary logistic regression.

Appendix B. Proof Sketch of Theorem 7.2

Proof. Let $\mathbf{Sim}_{\mathsf{DX}}$ be the simulator that exists by the Λ_{DX}-security of Σ_{DX} and $\mathbf{Sim}_{\mathsf{MM}}$ be the simulator that exists by the Λ_{MM}-security of Σ_{MM}. Then we describe the simulator $\mathcal{S}$ that simulates $\mathcal{A}$, in the context of the corruption of the server $\mathbf{S}$ and some subset of up to $n-1$ data owners.

- (simulating Init) $\mathcal{S}$ generates the keys
  - $(\mathsf{pk}_{\mathsf{num}}, \mathsf{sk}_{\mathsf{num}}) \leftarrow \mathsf{SHE.Gen}(1^k)$,
  - $(\mathsf{pk}_{\mathsf{key}}, \mathsf{sk}_{\mathsf{key}}) \leftarrow \mathsf{PKE.Gen}(1^k)$,

  and initializes a global linking graph G^*.
- (simulating Setup) For each honest data owner, $\mathcal{S}$ receives the setup leakage. $\mathcal{S}$ computes the following:
  - $\mathsf{EDX}^{\mathsf{data}} \leftarrow \mathbf{Sim}_{\mathsf{DX}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{DX}}(\mathsf{DX}^{\mathsf{data}}))$,
  - $\mathsf{EMM}^{\mathsf{link}} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{link}}))$,
  - $\mathsf{EMM}^{\mathsf{filter}} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{filter}}))$.

  $\mathcal{S}$ sends $(\mathsf{EDS}, \mathsf{ct}_K)$ to the server $\mathbf{S}$ where $\mathsf{ct}_K = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{key}}, 0^k)$ and $\mathsf{EDS} = (\mathsf{EMM}^{\mathsf{filter}}, \mathsf{EDX}^{\mathsf{data}}, \mathsf{EDX}^{\mathsf{link}})$. For each corrupt data owner $\mathbf{P}_i$, $\mathcal{S}$ has access to T_i, and $\mathcal{S}$ simulates the functionality $\mathcal{F}_{\mathsf{OPRF}}$ as described in Figure 3.
- (simulating Query) For each query q, $\mathcal{S}$ receives the leakage $\mathsf{op} = (\mathsf{filter}, \mathsf{link}, \mathsf{aggregate})$, the linking graph $\mathsf{G}^{\mathsf{link}} = (V, E)$, and the respective query leakage of the encrypted structures. Then, $\mathcal{S}$ computes the tokens:
  - (data tokens) for each (encrypted) record $\mathbf{r}$ that is output by some filter $(\mathbf{P}_i, x, \mathsf{value})$:
    1) $\mathcal{S}$ initializes a dictionary $\mathsf{DX}_{\mathbf{r}}$,
    2) for every numeric column $x \in X^{\mathsf{Num}}$, it sets $\mathsf{DX}_{\mathbf{r}}[x] = \mathsf{SHE.Enc}(\mathsf{pk}_{\mathsf{num}}, 0^k)$,
    3) since Σ_{DX} is response-revealing, $\mathcal{S}$ computes the token using the response $\mathsf{DX}_{\mathbf{r}}$ and the query leakage: $\mathsf{tk}_{\mathbf{r}}^{\mathsf{data}} \leftarrow \mathbf{Sim}_{\mathsf{DX}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{DX}}(\mathsf{DX}_i^{\mathsf{data}}, q), \mathsf{DX}_{\mathbf{r}})$,
  - (link tags) $\mathcal{S}$ sets the link tags for vertices in $\mathsf{G}^{\mathsf{link}}$:
    1) for all vertices such that $\mathsf{v}_{\mathbf{r}}.\mathsf{rid}$ already exists in G^* with a tag for linking condition $L[\mathsf{link}]$, $\mathcal{S}$ adds the previous $\mathsf{ltg}_{\mathbf{r}}^{\mathsf{link}}$ to $\mathsf{v}_{\mathbf{r}}$,
    2) for any vertices $\mathsf{v}'_{\mathbf{r}}$ neighboring some $\mathsf{v}_{\mathbf{r}}$ with an existing link tag, $\mathcal{S}$ sets the same tag $\mathsf{ltg}_{\mathbf{r}}^{\mathsf{link}}$ as $\mathsf{v}_{\mathbf{r}}$,
    3) for any remaining vertices, $\mathcal{S}$ samples a new $\mathsf{ltg}_{\mathbf{r}}^{\mathsf{link}} \leftarrow \{0,1\}^*$,
  - (query tokens) For each $(\mathbf{P}_i, x, \mathsf{value})$ in filter,
    1) for each vertex $\mathsf{v}_{\mathbf{r}} \in V$ such that $\mathsf{v}_{\mathbf{r}}.\mathsf{filterlist}$ contains $(\mathbf{P}_i, x, \mathsf{value})$, $\mathcal{S}$ adds $\mathsf{tk}_{\mathbf{r}}^{\mathsf{data}}$ to the tuple tks, and adds $\mathsf{ltg}_{\mathbf{r}}^{\mathsf{link}}$ to the tuple ltgs,

2) since Σ_{MM} is response-revealing, $\mathcal{S}$ uses the response tks and the query leakage to compute $\mathsf{ftk} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}_i^{\mathsf{filter}}, q), \mathsf{tks})$,
3) $\mathcal{S}$ uses the response ltgs and the query leakage to compute $\mathsf{ltk} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}_i^{\mathsf{link}}, q), \mathsf{ltgs})$,
4) $\mathcal{S}$ sets $\mathsf{filtertk} = \mathsf{filtertk} \cup (\mathbf{P}_i, \mathsf{ftk}, \mathsf{ltk})$

– $\mathcal{S}$ sets $\mathsf{G}^* = \mathsf{G}^* \cup \mathsf{G}^{\mathsf{link}}$,
– Finally, $\mathcal{S}$ sends $(\mathsf{filtertk}, \mathsf{aggregate})$ to $\mathbf{S}$.

We show through the following sequence of games that $\mathcal{A}$'s view in the $\mathbf{Ideal}^{\Lambda}_{\mathcal{Z},\mathcal{S}}(k)$ experiment is indistinguishable from its view in a $\mathbf{Hybrid}_{\mathcal{Z},\mathcal{A}}(k)$ experiment.

- Game_0: is an execution of a $\mathbf{Hybrid}(k)$ experiment.
- Game_1: is the same as Game_0 except that the $\mathcal{F}_{\mathsf{OPRF}}$ functionality is simulated. The adversary $\mathcal{A}$'s view does not change as a result of this.
- Game_2: is the same as Game_1 except that ct_K is replaced with $\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{key}}, 0^k)$. The security of PKE guarantees that we can replace these without affecting $\mathcal{A}$'s view.
- Game_3: is the same as Game_2 except that the dictionaries $\mathsf{DX}_{\mathbf{r}}$ are replaced with dictionaries containing SHE encryptions of 0. The security of SHE guarantees that this will not affect the adversary's view.
- Game_4: is the same as Game_3 except that each $\mathsf{EDX}^{\mathsf{data}}$ is replaced with a simulated encrypted dictionary:
  – In Setup, $\mathsf{EDX}^{\mathsf{data}} \leftarrow \mathbf{Sim}_{\mathsf{DX}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{DX}}(\mathsf{DX}^{\mathsf{data}}))$, and
  – In Query, $\mathsf{tk}^{\mathsf{data}}$ is replaced with $\mathsf{tk}^{\mathsf{data}} \leftarrow \mathbf{Sim}_{\mathsf{DX}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{DX}}(\mathsf{DX}^{\mathsf{data}}, q), \mathsf{DX}_{\mathbf{r}})$

  The Λ_{DX} security of Σ_{DX} guarantees that the simulated dictionary $\mathsf{EDX}^{\mathsf{data}}$ and tokens are computationally indistinguishable from a real dictionary and tokens.
- Game_5: is the same as Game_4 except that each $\mathsf{EMM}^{\mathsf{link}}$ is replaced with a simulated encrypted multi-map as follows:
  – In Setup, $\mathsf{EMM}^{\mathsf{link}} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{link}}))$, and
  – In Query, ltk is replaced with $\mathsf{ltk} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{link}}, q), \mathsf{ltgs})$, where ltgs contains random linking tags that are simulated using the linking graph as described in the (link tags) step of the proof. Since G^* describes all the previous links visible to the adversary, the simulator can sample random tags such that the adversary's view is unchanged.

  The Λ_{MM} security of Σ_{MM} guarantees that the simulated $\mathsf{EMM}^{\mathsf{link}}$ and all simulated tokens are computationally indistinguishable from a real encrypted multi-map.
- Game_6: is the same as Game_5 except that each $\mathsf{EMM}^{\mathsf{filter}}$ is simulated as follows:
  – In Setup, $\mathsf{EMM}^{\mathsf{filter}} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{S}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{filter}}))$, and
  – In Query, ftk is replaced with $\mathsf{ftk} \leftarrow \mathbf{Sim}_{\mathsf{MM}}(\mathcal{L}_{\mathsf{Q}}^{\mathsf{MM}}(\mathsf{MM}^{\mathsf{filter}}, q), \mathsf{tks})$ where tks is a tuple of $\mathsf{tk}^{\mathsf{data}}$ tokens.

  The Λ_{MM} security of Σ_{MM} guarantees that the simulated $\mathsf{EMM}^{\mathsf{filter}}$ and all simulated tokens are computationally indistinguishable from a real encrypted multi-map.

Finally, we note that Game_6 is equivalent to $\mathbf{Ideal}^{\Lambda}_{\mathcal{Z},\mathcal{S}}(k)$, and hence Synq is Λ_{Synq}-secure for this corruption setting. We defer the expanded proof to the full version of our paper.

Appendix C. Extended Application: Boston Wage Equity

The Boston wage equity study [69] was an initiative led by the Boston Women's Workforce Council [4]. It used data contributed by over 100 different employers to analyze differences in wage across gender, race, and job roles in a privacy-preserving manner using MPC. Specifically, in their initial design, they used a variant of additive secret sharing in which random masks were added to each data owners' values such that $\mathsf{masked\_val} = \mathsf{mask} + \mathsf{value}$. These masked values were then sent to the server and the random masks were sent to an analyst. The server aggregated each submitted masked value $\mathsf{masked\_val}_i$ from data owner $\mathbf{P}_i$: $\mathsf{masked\_sum} = \mathsf{masked\_val}_0 + \ldots + \mathsf{masked\_val}_n$. The analyst performed the same aggregation using the masks: $\mathsf{mask\_sum} = \mathsf{mask}_0 + \ldots + \mathsf{mask}_n$. Finally, the server sent over the masked_sum to the analyst, and the analyst subtracted mask_sum from the masked_sum to compute the total sum, which they would divide by n to arrive at the average. The privacy of individual data values was preserved as long as the server and the analyst did not collude.

Using Synq, the analyst would make one query for each average that needed to be computed across gender, race, and job position. For example, to compute the average salary for Asian female CEOs, the analyst would formulate a query that filters each data owners' data on these values. In this use case, linking is skipped ($\mathsf{link} = 0$) and filtering is directly followed by the computation of the average salary. The full query can be expressed as follows in Synq-QL:

$$\begin{aligned}&([(\mathbf{P}_1, \mathsf{role}, \texttt{"CEO"}), (\mathbf{P}_1, \mathsf{race}, \texttt{"Asian"}),\\ &(\mathbf{P}_1, \mathsf{gender}, \texttt{"F"}), \ldots,\\ &(\mathbf{P}_n, \mathsf{role}, \texttt{"CEO"}), (\mathbf{P}_n, \mathsf{race}, \texttt{"Asian"}),\\ &(\mathbf{P}_n, \mathsf{gender}, \texttt{"F"}), 0, [(\mathsf{Average}, \mathsf{salary})])\end{aligned}$$

In the original MPC-based protocol, the untrusted server receives masked values, which can effectively be viewed as ciphertexts. The server then sums these values together. Through receiving the masked values, the server in the Boston wage equity study learns the size of each data owners' data. The server additionally learns the number of data owners, which it also forwards to the analyst, and the specific operation to be performed (computing an average). If the wage equity study had used Synq instead, the server would learn the size of the data owners' data through its storage of ciphertexts. It would also learn the total number of data owners and the aggregate operation (computing an average). Similar to the original protocol, the analyst would learn intermediate aggregated sums. In summary, since in this setting there is no linking, Synq does not leak anything more than the protocol used in the Boston study. Additionally, Synq would also support re-computation of these averages with different filters, whereas the original protocol would need to recompute all the masks and all the secret shares for any further computation.

Appendix D.
Meta-Review

D.1. Summary

This paper presents Synq, an encrypted database system tailored for analyses related to certain public policy applications. It supports multiple data sources, and presents a query language for expressing different computations over these data. The language admits filtering and linking of different records, and can be used to specify aggregation functions over the result. Synq has been specifically designed to fit the needs of public-policy case studies, and does not require data owners to be online during computation.

D.2. Scientific Contributions

- Creates a New Tool to Enable Future Science
- Provides a Valuable Step Forward in an Established Field

D.3. Reasons for Acceptance

1) The paper analyses data analytics needs for certain public policy applications, and identifies critical challenges in this space that existing work does not address. Most notably: the need for data owners to be offline during computation and the ability to link records using multiple fields.
2) The paper presents a solution to the identified problem in the form of a new cryptographic scheme that uses existing primitives in a novel way. This scheme is supported by a new query language for expressing aggregate queries over datasets.
3) The paper defines the security of the Synq protocol using an ideal functionality, which expresses the leakage, and proves that the scheme is secure with respect to this ideal functionality.

D.4. Noteworthy Concerns

Reviewers had differing opinions on the value of the Taxonomy in the introduction of the paper.

Appendix E.
Response to the Meta-Review

We thank the reviewers for their useful feedback, all of which helped shape the final version of this paper.

The paper includes the topology taxonomy to capture a broader point about historical trends in systems that enable analytics over encrypted data. Specifically, the paper maps each of the prior works to the topologies in Table 2 to show that increased synchronization requirements in prior systems lead to decreased adherence to the design considerations in Table 1 (all of which the paper identifies as important for real-world deployment in the public policy context). This relationship between synchronization and usability is perhaps intuitive, but, to our knowledge, has never been observed formally in prior work. The paper presents the topology taxomony to formally make this insight; it then leverages this observation to make a compelling case for Synq's design considerations. More broadly, we believe the taxonomy can serve as a useful guideline for future works which are concerned about similar design requirements.