Sunshine Systems: Enabling Privacy-Preserving Compliance with Freedom of Information Laws

Aarushi Goel
Rutgers University
aarushi.goel@rutgers.edu

Gabriel Kaptchuk
University of Maryland
kaptchuk@umd.edu

Yuange Li
Rutgers University
yl1407@rutgers.edu

Abstract

Freedom of information laws (informally known as Sunshine Laws), such as the US Federal Freedom of Information Act, provide legal pathways for members of the public to compel governments to produce internal documentation. These laws have become a powerful tool for government accountability, allowing journalists and activists to uncover malfeasance and advocate for change. However, because compliance with these laws requires government actors—the same entities who may be accused of malfeasance—to produce documentation, there is a risk that evidence of malfeasance may be suppressed rather than produced in response to a query.

In this work, we introduce a cryptographic framework called *Sunshine Systems*, designed to constrain government actors' ability to circumvent freedom of information laws. At a high level, Sunshine Systems enable agencies to prove, in zero knowledge, that they have produced the complete and correct set of documents responsive to a given request. We instantiate a Sunshine System that supports keyword queries, building on recent advances in lookup arguments. We implement and evaluate our approach on low-cost hardware, demonstrating its practicality even for small government agencies. Our evaluation shows that our implementation dramatically out-performs a PoneglyphDB [SIGMOD '25] baseline.

Contents

1 Introduction **3**
- 1.1 Our Contributions . . . 4

2 Technical Overview **5**

3 Related Work **8**

4 Preliminaries **10**
- 4.1 Vector Commitment . . . 10
- 4.2 Security Models and Zero-Knowledge Arguments . . . 11
- 4.3 Lookup Arguments . . . 13

5 Defining Sunshine Systems **14**
- 5.1 Model . . . 14
- 5.2 Online Query Phase . . . 15
- 5.3 Audit Phase . . . 17

6 Instantiating the Online Query Phase **18**
- 6.1 Committing to a Document Set . . . 19
- 6.2 Query Response and Verification . . . 20

7 Instantiating the Audit Phase **29**
- 7.1 Overview . . . 29
- 7.2 Audit Protocol . . . 32

8 Implementation and Evaluation **39**
- 8.1 Implementation . . . 39
- 8.2 Evaluation Results . . . 39

A Additional Details for Our Online Query Phase Protocol **49**
- A.1 Instantiating Vector Commitments . . . 49
- A.2 Instantiating $\Pi_{\mathsf{wf}}, \Pi_{\mathsf{cnt}}, \Pi_{\mathsf{sub}}, \Pi_{\mathsf{mem}}$ and $\Pi_{\cap}$ . . . 51
- A.3 Cost Analysis . . . 63

B Additional Details for Our Audit Phase Protocol **65**
- B.1 Instantiating Π_{state} and Π_{upd} . . . 65
- B.2 Cost Analysis . . . 69

1 Introduction

Government agencies produce a tremendous number of documents in the course of performing their everyday operations. These documents serve a dual purpose as institutional memory, facilitating continuity when employees rotate or administrations change, and as an audit log. While government auditors generally consume these logs, freedom of information laws (also called sunshine laws) provide a way for non-governmental actors—like journalists, activists, and other interested citizens—to compel governments to make these documents public. The most well-known example of these laws is the US Federal Freedom of Information Act, but many similar laws have been enacted, both in countries around the world [7] and at the state/regional level [61]. As a result, agencies of all sizes around the world are required to provide documents in response to the public's queries. These laws have become a powerful tool for government accountability, as they help uncover acts of malfeasance or systemic government failure [30].

The freedom of information ecosystem often fails in predictable ways. In addition to long processing delays and "normal" bureaucratic challenges, government agencies can omit records that *should* be included in response to a query. The National Freedom of Information Coalition (NFOIC) reviewed 511 open records requests from 2016–2017 in Kentucky and found that nearly 50% of cases exhibited a violation of the state's records laws in one way or another [60]. In the best case, this can be the result of a simple mistake, outdated document storage systems, or administrative incompetence. But, because freedom of information queries are a foundational tool for combating government malfeasance, documents may be omitted in order to actively suppress evidence of misdeeds or to ensure that information that government agencies may find embarrassing is not uncovered. As the NFOIC wrote: "at the press of a delete key... a public record can be destroyed or hidden, and with it, your right to the record can be stripped away" [60]. While it is impossible to know if an omission is intentional, omissions undermine freedom of information law's goals.

Concerns that documents may be omitted intentionally are, sadly, far from theoretical, and there are cases in which it is hard to imagine that omissions could be anything but intentional. For example, the Chester, New York police department denied the existence of records documenting misconduct on the part of its police chief, only to have the county district attorney's office later reveal documents it had received from the police department on the subject [70]. In Inglewood, California, a court determined that the police department had intentionally destroyed documentation right before a new sunshine law came into effect; the court then ordered that the police department proactively post *all* documents subject to the law for a period of three years—a ruling that clearly demonstrates that the court thought the department could not be trusted to respond truthfully to requests [11, 68, 69]. Similarly, a Martin County, Florida commissioner was accused of having "scrubbed information and altered public records," although the charges were later dropped because prosecutors were unable to bring enough evidence [60]. While smaller agencies are better situated to commit these offenses, more sophisticated agencies have been known to follow this pattern: the National Security Agency claimed that "no responsive documents" existed in response to a query for a well-documented video of a lecture by Rear Adm. Grace Hopper [50].[1] Shortly thereafter, the National Archives released the footage. In most documented cases, the public only learned that the initial denial was untrue because another agency had copies of the records in question; most of the time there is simply no way for the public to corroborate agency claims. As articulated by the NFOIC, "many violations [of transparency laws] occur outside the public's view" [60].

Towards more trustworthy freedom of information responses. Rectifying this situation

[1]The NSA clarified that the footage was stored on a medium it no longer had the technology to read, and claimed that this meant the agency did not need to search that footage in response to the request.

requires the development of both better administrative practices and better technologies. There are two largely orthogonal ways in which the data returned by freedom of information queries fails to capture reality: (1) the returned documents themselves contain errors relative to what really happened, either because an internal actor is trying to suppress information or because of an innocent oversight, and (2) the wrong documents are returned. Mitigating the first threat is almost entirely administrative, as it concerns the digitization of real-world events. The second threat, on the other hand, concerns the ways in which records are stored, searched, and processed—and, thus, is potentially ripe for technical intervention.

In this work, we study the problem of designing technical systems that make the responses to freedom of information queries more trustworthy. Although we cannot build a system that *forces* an agency to hand over documents, we observe that it is sufficient to make omission of relevant documents detectable; evidence that an agency failed to comply with the law can then be brought to court to compel action. Specifically, we build a system that makes the responses to freedom of information queries *verifiable* with respect to the state of some incrementally built database, without requiring that the agency release anything else about its documents. While this approach does not guarantee that all documents are correct—as the documents added to the database can be inconsistent with real-world events—this significantly constrains the ability of malicious actors.

Barriers to designing a performant system. Making freedom of information responses verifiable is conceptually straightforward: we can require that agencies commit to new documents as they are generated and then use zero-knowledge proofs to demonstrate that the returned documents are the correct ones (without revealing anything about the other documents). Importantly, the proofs must demonstrate that *all* relevant records have been returned; a poorly designed system might allow the prover to satisfy the relation with a subset of the "true" witness. Instantiating this template in a performant way, however, is challenging, particularly when one considers that any proposed solution must (a) scale to a large number of documents, (b) be runnable by agencies with small budgets (e.g., local police departments), and (c) interoperate with different document storage approaches.

1.1 Our Contributions

In this work we study the design constraints of Sunshine Systems, a class of cryptographic systems that are designed to make responses to freedom of information requests verifiable without revealing anything unnecessary about agencies' documents. We make the following contributions:

- **Defining Sunshine Systems.** We formalize Sunshine Systems. While these systems are closely related to existing systems like key transparency and verifiable SQL, we observe that there are subtle differences and existing systems cannot be directly applied to our setting.
- **Concrete instantiation for keyword queries.** There is no standard, formal query language for freedom of information requests. Thus, we initiate the study of these systems for a specific but natural, and widely used, query language: keyword search. Documents are associated with a set of keywords (e.g., chosen by a machine learning algorithm or manually curated). Each query consists of a set of keywords and the response contains all documents that contain *all* of these keywords. This is a rich query language that can easily be used to represent time ranges, document topics, and important events. Our instantiation relies on a structured reference string, and we prove its security in the algebraic group and random oracle models.
- **Implementation and evaluation.** Finally, we implement and evaluate our construction on the Enron email dataset [44]. Our results demonstrate that even on commodity hardware, producing query results requires at most several minutes, and preprocessing can be completed within several

hours—an infrequent task that can run overnight. We further demonstrate that our approach significantly outperforms naive solutions while providing stronger guarantees.

2 Technical Overview

Today, freedom of information requests rely almost entirely on the agency's word—the very agency whose misdeeds may be at stake. Requesters see only the response, with no insight into what was searched or withheld. Our primary concern is undetectable suppression of information. While we cannot compel an agency to disclose a document or verify that real-world events were faithfully recorded, searching a fixed body of records and returning matches is fundamentally *computation*, which can be made verifiable. Detectable omissions would allow requesters to seek court intervention, as with other forms of noncompliance today.

Design goals and threat model. We begin by informally describing the design goals shaped by this setting:

- *Detects omitted documents:* An agency should not be able to omit documents from the response to an information request without the requester *detecting* this omission.
- *Limits unrelated documents:* We also want to ensure that the agency cannot *add* unrelated documents to those returned. This property is in response to critical scholarship on transparency that institutions may disclose *too many* documents to consume investigators' resources [2]—an offline analog of a denial-of-service attack. We note that this does not prevent agencies from also returning other documents, but these documents' relevance to the query cannot be justified using our system.
- *Preserves document privacy:* Some documents produced by government agencies may be sensitive and cannot be immediately released to the public, but may later be subject to sunshine laws. The verification system must not reveal unnecessary information about documents, either as documents are created or when answering queries.
- *Interoperability:* Government agencies are not required to store documents in a particular format or using a particular technology. We therefore require a system interoperable with a variety of storage schemes.
- *Low concrete costs:* Local agencies may not have much computational power, nor the budget to rent cloud resources in response to information requests. As such, we need a system that can use existing, consumer-grade computers and fit naturally into the office workflow. Additionally, we will require proofs that are succinct in the number of documents so that lightweight verifiers can process the responses. We use "offline" preprocessing, which can run overnight when government offices are closed, resulting in concretely efficient online prover times.

We consider an adversary that is *semi-honest* while initially producing documents, but may later be malicious when maintaining those records or answering information requests. This captures a natural, real-world behavior: a government actor who initially thought there was nothing wrong with their actions, but later is afraid of embarrassment or punishment if those actions come to light; consider, for example, the California and Florida cases discussed in Section 1 in which officials tampered with government records long after they were created [11, 60, 68, 69]. Alternatively, an automated system, such as a mail server, may collect documents that government actors might try to suppress.

Problems with the strawman construction. A natural cryptographic approach would be via a hiding and binding *commitment* to a particular document set. The agency acts as *prover*, and gives the requester a zero-knowledge proof that the response contains *all and only* the matching documents, without revealing others. The requester acts as *verifier*. This *commit-and-prove* approach [14] provides a feasibility result, but faces difficulties:

– *Linear online prover complexity:* A straightforward proof iterates through all documents, showing that every omitted document is irrelevant to the query. This work is repeated for *each* query and becomes expensive as the set of documents grows.
– *Documents are added incrementally:* A malicious prover could commit to modified data if allowed to choose a fresh, unrelated commitment after receiving the query. Instead, commitments to the documents must be published before the query, and later commitments must preserve the earlier records. Simply including a separate commitment for every document in the statement would make reading that statement linear in the number of documents.

A template for efficient Sunshine Systems. Two families of techniques naturally address these difficulties:

1. *Improving online performance with lookup arguments:* Zero-knowledge lookup arguments allow a prover to convince a verifier that a (set of) committed value(s) are entries in a larger committed table. They are succinct and can use *preprocessing* to achieve online prover complexity sublinear in the table size. This allows the prover to reuse precomputation across queries and defer "expensive" computation to overnight downtime.

2. *Aggregating documents using append-only commitments:* A succinct vector commitment binds a sequence of documents. As documents are produced, the agency can prove that updates preserve existing entries while appending new ones, as in certificate and key transparency systems [1, 24, 36, 37, 45, 51]. A third-party auditor verifies updates, enabling query verifiers to rely on the latest audited commitment. Although recruiting auditors for key transparency has proved difficult, government oversight offices and civil society could serve this role for Sunshine Systems, provided verification is inexpensive.

Combining these two techniques provides a workable template. We represent documents by identifiers linked to *salted hashes* of the underlying documents, so the proofs reason over identifiers rather than the documents themselves. These identifiers are aggregated into a succinct commitment, and these updates are verified by an external auditor. At query time, a lookup argument certifies the returned identifiers. Section 5 formalizes this template, separating the online query phase from auditing. Combining these techniques, however, requires care.

The *first challenge* is that lookup arguments do not by themselves prevent the prover from omitting matching entries from a response. Completeness depends on the *query language*, so the commitment and proofs must bind the prover to the complete response for that language. The query language must support efficient processing: evaluating an arbitrary predicate separately for every document would still require a full scan. Looking ahead, we focus on the *keyword query* language, which provides a nice balance of expressiveness and low-cost, per-document preprocessing to overcome this challenge.

The *second challenge* is maintaining table-dependent lookup preprocessing across updates. Adding even one entry can require recomputing the entire preprocessing. Dutta et al. [21] provide "updatable" lookup arguments that reuse preprocessing across table updates and amortize the

recomputation cost. They combine these arguments with consistency checks to verify batches of table updates.

Instantiating the template for keyword queries. Keywords are an expressive query language that captures many natural queries that might be made in a Sunshine System (e.g., *"give me all police records associated with keywords* SEXUAL MISCONDUCT*,* CAPTAIN*, and* REPRIMAND*."*). Formally, each document has an identifier id_i and an associated set of keywords $\mathcal{W}_i$. Keywords can be assigned as the document is produced, either manually, using a pre-set dictionary, or using some machine-learning-powered system.

A possible approach. We support *conjunctions* of keywords, i.e., retrieving all documents that contain every keyword in the query. The natural starting point is an *index*, in which each keyword k is associated with the full list of relevant document identifiers $\mathsf{id}_1, \mathsf{id}_2, \ldots$.

One approach is to authenticate this index using zero-knowledge set accumulators [34], which provide succinct representations of sets and support membership proofs without revealing the remaining elements. We maintain one accumulator for each keyword's document identifier set. Each accumulator can be instantiated using a hiding commitment to a table of these identifiers and zero-knowledge lookup arguments for membership. A query can then be answered by returning the intersection of the corresponding identifier sets and proving that it is computed correctly. The construction of [34] supports such proofs while keeping the remaining elements private.

However, this construction relies on a trusted owner to maintain and update the committed sets. In our setting, the agency may be malicious, so we must verify that each set's updates preserve existing elements and only insert new ones, which is challenging and expensive with these accumulators. It might seem like instantiating each of these accumulators using the updatable lookup arguments of Dutta et al. [21] can solve these problems via their consistency checks for these updates. However, verifying updates to each keyword's document identifier set *independently* is still not sufficient. Consider an existing document with identifier id that is associated with k_1 but not k_2. Adding id to the set for k_2 is structurally valid, but retroactively edits the set of keywords associated with a document. This allows a malicious prover to inject non-relevant documents into a response, failing to meet our *limits unrelated documents* goal. Each appended document identifier must therefore be absent from the *entire* existing document set. We call this requirement *document permanence*. Moreover, if only the affected per-keyword commitments visibly change, those changes reveal which keywords were updated.

A straightforward adaptation checks document permanence against every keyword's set. With 1,000 keywords, each associated with 10,000 documents, this requires 1,000 non-membership checks per appended identifier. If implemented by scanning these sets, the prover checks 10 million keyword–document combinations for a single append. These checks must also be combined with proofs that the set updates are consistent with the same valid document append.

Our idea. We want to avoid repeatedly checking all keyword lists while answering queries without scanning the entire document set. We therefore maintain counts of the documents associated with each keyword to support query completeness, and a single set of all document identifiers to enforce document permanence. Namely, the prover incrementally adds pairs of the form (k, id) to a single (unordered) table, while keeping a running count of how many times each keyword k appears in the table. These counts form a vector E_{cnt}. The public commitment to the document set consists of hiding vector commitments to the pairs[2] and to E_{cnt}. The initial audit certifies that the pairs are valid and non-repeating and that the counts are consistent with the table. Subsequent audits

[2]Formally, the commitment consists of two vector commitments, the first of which holds the keywords and the second of which contains the document identifiers.

preserve these properties.

For each queried keyword, lookup arguments establish that the selected identifiers are associated with that keyword in the table. Proving that these identifiers are distinct and that their number equals the corresponding committed count prevents the prover from suppressing matching documents. These checks are performed in zero knowledge. We then prove that the public response is exactly the intersection of these complete identifier sets. The intermediate sets remain private; details of our approach can be found in Section 6.

It remains to ensure that updates to the table are well-formed. We use homomorphic vector commitments to combine old and new entries. For each update, the prover commits to new pairs at positions directly after the existing entries and proves that the old entries remain unchanged. When homomorphically combined with the old commitment, the result is the new table commitment. Similarly, the prover computes a commitment to counts of the new entries, proves that this vector is consistent with those entries, and then homomorphically adds it to the old count commitment.

To enforce document permanence, the audit also maintains a commitment to the set of all document identifiers, linked to those appearing in the table. For each update, the agency proves that the updated identifier set is the disjoint union of the previous set and the identifiers in the new batch. Thus, no identifier in the new batch already appears in the previous document set. These checks are linked to the same appended pairs and performed in zero knowledge, without revealing the appended identifiers or their associated keywords. Importantly, subsequent audits reuse the certification of the old pair table and prove the validity of the appended entries. Document permanence is checked against the global identifier set; we discuss the details of the audits in Section 7.

A remaining challenge is maintaining the preprocessing as documents are appended. We use the Base+Cache approach of Dutta et al. [21]. The prover maintains a preprocessed table containing most of the data, called the *Base*, while incrementally added entries form a smaller *Cache*. The prover can then generate proofs that reason over both the Base and Cache, but the proving cost increases as the Cache grows. To manage this cost, the prover periodically merges the Cache into the Base and recomputes the preprocessing, reducing subsequent query costs. We call this procedure *rebasing*. Notably, preprocessing is recomputed periodically rather than after every update.

A final thought. Our goal in this work is to scope this interesting social problem as a technical, cryptographic one and provide a set of formal definitions. Our construction combines preprocessing-based lookup arguments with complete keyword-query responses and privacy-preserving append-only audits that enforce document permanence. By carefully weaving together these techniques, our result serves as a feasibility result—and hopefully inspires further work to continue to explore more features of the freedom of information ecosystem.

3 Related Work

Cryptography and governance. Several other works study the promise of using zero-knowledge as a tool for accountable governance. Goldwasser and Park [35] and Frankle et al. [29] study the use of zero-knowledge proofs as a way to monitor government processes that needed to remain secret, like national security related hearings. More recently, Bamberger et al. identified multiple uses for zero-knowledge within the legal profession, including during mergers and police investigations [6]. Green et al. [38] and Fetzer et al. [26] studied the use of zero-knowledge as a way to constrain the access law enforcement might be able to get to encrypted communications and provide aggregate insight into emerging enforcement trends. Fabrega et al. [25] look at cryptographic tools that would make voter registration verifiable, although their techniques do not consider zero-knowledge in

particular. Our work adds to this line of work by considering freedom of information laws, which is not directly considered by any prior work.

Preventing omissions. Several works have considered authenticated data structures that not only facilitate membership proofs, but also are designed to prevent a prover from omitting elements in their responses. For example, a line of work on "authenticated data structures" [72] considers adversarial data publishers who may want to suppress data provided by trusted data owners. Devanbu et al. [19] and Martel et al. [53] provide solutions to this problem based on Merkle trees. Micali et al. [56] introduce zero-knowledge sets, supporting membership and non-membership proofs while hiding information beyond the query answers. Chase et al. [16] study mercurial commitments with applications to zero-knowledge sets. Papamanthou et al. [63] provide succinct verification of operations on dynamic sets. Ghosh et al. [34] additionally hide the sets beyond the operation's result, including for intersection queries. However, updates in these constructions are performed by a trusted data owner. Several works in the database literature have also built systems that focus on ensuring database queries are *complete* with respect to the query, e.g., [47, 62].

Verifiable database queries. A line of work supports verifiable SQL queries [40, 48, 76–78], with some constructions also providing zero knowledge. Among these approaches, PoneglyphDB [40] provides non-interactive zero-knowledge proofs for SQL queries over a committed database. It uses PLONKish circuits for SQL operations and recursive proof composition to support complex queries. We use PoneglyphDB as a baseline for evaluating online proving cost of privacy-preserving query verification. For selection queries (corresponding to keyword queries in our work), its circuits process the input table, so the proving cost depends on the table size rather than only on the number of matching records. In contrast, our work leverages preprocessing to support keyword queries without walking through the whole database. Furthermore, PoneglyphDB does not provide a concrete approach for our updatability requirements. See Table 3 for a comparison of online proving and verification times. These systems differ in query support, privacy, and update requirements. CorrectDB [5] instead uses trusted execution environments (TEE) for query authentication. To our knowledge, our work is the first to consider this completeness property in combination with preprocessing-powered zero-knowledge proofs.

Lookup Arguments. Recent works have developed lookup arguments with sublinear online complexity and quasilinear preprocessing in the table size. Early constructions [22, 32, 41, 65, 74, 75] use vector commitments based on univariate polynomial commitments, while Setty et al. [67] extend this approach to multilinear polynomial commitments. Subsequent works support richer lookups, including segment lookup [18] and matrix lookup [20]. A key drawback is that table updates typically require recomputing the entire preprocessing. Dutta et al. [21] address this with *updatable lookup arguments*, where preprocessing can be amortized depending primarily on the modified entries. To our knowledge, completeness (discussed above) has not been studied for lookup arguments.

Key transparency. The problem we consider borrows the auditing model used in key transparency systems [1, 10, 15, 24, 36, 37, 42, 45, 46, 51, 52, 55, 73]. These systems are widely deployed and generally rely on Merkle trees, which are incompatible with state-of-the-art lookup arguments. These works also consider a different notion of "well-formed" updates and a different notion of response completeness.

Concurrent work. Namboothiry et al. [57] introduce *TruthTable*, a verifiable SQL engine for a static committed database. It proves correct and complete query results, but does not provide database privacy or verifiable updates.

Pappas et al. [64] introduce *Rogue*, an updatable matrix lookup argument that improves update costs of Dutta et al. [21] by avoiding periodic full-table reprocessing, and use it to construct *RogueDB*,

Framework	Query interface	Query completeness	Zero knowledge	Updatability	Document permanence
ZK sets [56]	Point queries	✓	✓	×	–
Authenticated sets [63]	Set operations	✓	×	✓†	–
ZK set algebra [34]	Set operations	✓	✓	✓†	–
CorrectDB [5]	SQL queries	✓	TEE*	✓	×
IntegriDB [78]	SQL queries	✓	×	✓†	×
vSQL/zk-vSQL [76, 77]	SQL queries	✓	×/✓	✓	×
RogueDB [64] (concurrent)	SQL queries	✓	×	✓	×
TruthTable [57] (concurrent)	SQL queries	✓	×	×	×
PoneglyphDB [40]	SQL queries	✓	✓	×	×
Sunshine System [Ours]	Conjunctive keyword queries	✓	✓	✓	✓

Table 1: Functional comparison of query systems. †: updates are performed by a trusted data owner. TEE*: TEE-based query authentication, rather than cryptographic zero-knowledge proofs. –: the property is not specified for the native interface.

a verifiable SQL system supporting authenticated indexes and database updates. However, Rogue does not achieve the zero-knowledge guarantee for private-table lookups required by our system. Its general database update functionality also does not satisfy our document permanence requirement. Table 1 summarizes the functional differences between these systems, while Table 3 compares their online proving and verification times.

4 Preliminaries

Notation. Throughout this work, we write λ to denote the security parameter and negl to denote a negligible function. We use $\mathbb{F} = \mathbb{F}_p$ to denote a prime field. "PPT" stands for probabilistic polynomial time, and "n.u." stands for non-uniform. For every nonnegative integer r, let $[r] := \{1, \ldots, r\}$, with $[0] = \emptyset$. Vector entries are indexed from one. Empty sums and products are interpreted as 0 and 1, respectively.

4.1 Vector Commitment

Definition 1 (Vector Commitment)**.** *A vector commitment (VC) scheme for vectors over $\mathbb{F}$ is a tuple of PPT algorithms $\mathsf{VC} = (\mathsf{VC.Setup}, \mathsf{VC.Commit}, \mathsf{VC.Open}, \mathsf{VC.Verify})$ defined as follows:*

- $\mathsf{pp}_{\mathsf{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$*: takes as input the security parameter λ and an upper bound d on the vector length, and outputs public parameters $\mathsf{pp}_{\mathsf{VC}}$.*
- $(\mathsf{Com}_V, \mathsf{op}_V) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, V)$*: takes as input $\mathsf{pp}_{\mathsf{VC}}$ and a vector $V \in \mathbb{F}^r$ of a public length $r \le d$, and outputs a commitment Com_V and the opening information op_V.*
- $(V|_{\mathcal{S}}, \pi_{\mathcal{S}}) \leftarrow \mathsf{VC.Open}(\mathsf{pp}_{\mathsf{VC}}, \mathsf{Com}_V, \mathcal{S}, \mathsf{op}_V)$*: takes as input $\mathsf{pp}_{\mathsf{VC}}$, a commitment Com_V to a vector of public length r, an index set $\mathcal{S} \subseteq [r]$, and its opening information op_V. It outputs $V|_{\mathcal{S}}$ and an opening proof $\pi_{\mathcal{S}}$.*

- $b \leftarrow \mathsf{VC.Verify}(\mathsf{pp_{VC}}, \mathsf{Com}_V, \mathcal{S}, V|_{\mathcal{S}}, \pi_{\mathcal{S}})$: *takes as input* $\mathsf{pp_{VC}}$, Com_V, *an index set* $\mathcal{S} \subseteq [r]$, *values* $V|_{\mathcal{S}}$, *and a proof* $\pi_{\mathcal{S}}$, *and outputs a bit* $b \in \{0,1\}$.

A pair (V, op_V) *is a* valid opening *of* Com_V *if recomputing* $\mathsf{VC.Commit}$ *on* V *using the randomness recorded in* op_V *reproduces* Com_V *and the claimed opening information.*

These algorithms satisfy the following properties:

- ***Completeness:*** *Let* $\mathsf{pp_{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. *For every public length* $r \le d$, *every* $V \in \mathbb{F}^r$ *and every* $\mathcal{S} \subseteq [r]$,

$$\Pr\left[\mathsf{VC.Verify}(\mathsf{pp_{VC}}, \mathsf{Com}_V, \mathcal{S}, V|_{\mathcal{S}}, \pi_{\mathcal{S}}) = 1 \middle| \begin{matrix}(\mathsf{Com}_V, \mathsf{op}_V) \leftarrow \mathsf{VC.Commit}(\mathsf{pp_{VC}}, V), \\ (V|_{\mathcal{S}}, \pi_{\mathcal{S}}) \leftarrow \mathsf{VC.Open}(\mathsf{pp_{VC}}, \mathsf{Com}_V, \mathcal{S}, \mathsf{op}_V)\end{matrix}\right] = 1.$$

- ***Subvector Binding:*** *Let* $\mathsf{pp_{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. *For every public length* $r \le d$ *and every n.u. PPT adversary* Adv, *which chooses the index set* $\mathcal{S} \subseteq [r]$ *and the subvectors* $V_0, V_1 \in \mathbb{F}^{|\mathcal{S}|}$ *below,*

$$\Pr\left[\begin{array}{l} V_0 \neq V_1 \\ \wedge \mathsf{VC.Verify}(\mathsf{pp_{VC}}, \mathsf{Com}^\star, \mathcal{S}, V_0, \pi_0) = 1 \\ \wedge \mathsf{VC.Verify}(\mathsf{pp_{VC}}, \mathsf{Com}^\star, \mathcal{S}, V_1, \pi_1) = 1 \end{array} \middle| \begin{array}{c} (\mathsf{Com}^\star, \mathcal{S}, V_0, \pi_0, V_1, \pi_1) \\ \leftarrow \mathsf{Adv}(\mathsf{pp_{VC}}) \end{array}\right] \le \mathsf{negl}(\lambda),$$

- ***Hiding:*** *Let* $\mathsf{pp_{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. *For every supported public length* $r \le d$, *every* $\mathcal{S} \subseteq [r]$ *and* $V_0, V_1 \in \mathbb{F}^r$ *satisfying* $V_0|_{\mathcal{S}} = V_1|_{\mathcal{S}}$, *define, for* $b \in \{0,1\}$, $(\mathsf{Com}_b, \mathsf{op}_b) \leftarrow \mathsf{VC.Commit}(\mathsf{pp_{VC}}, V_b)$ *and* $(V_b|_{\mathcal{S}}, \pi^b_{\mathcal{S}}) \leftarrow \mathsf{VC.Open}(\mathsf{pp_{VC}}, \mathsf{Com}_b, \mathcal{S}, \mathsf{op}_b)$. *Then*

$$\{(\mathsf{pp_{VC}}, \mathsf{Com}_0, \mathcal{S}, V_0|_{\mathcal{S}}, \pi^0_{\mathcal{S}})\} \approx_c \{(\mathsf{pp_{VC}}, \mathsf{Com}_1, \mathcal{S}, V_1|_{\mathcal{S}}, \pi^1_{\mathcal{S}})\}.$$

  The challenge vectors and index set may be chosen efficiently after the public parameters are known, subject to the same constraints; the choosing algorithm's retained state is included in both views.

- ***Linear Homomorphism:*** *Let* $\mathsf{pp_{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. *For every public length* $r \le d$, $V_0, V_1 \in \mathbb{F}^r$ *and scalars* $\alpha, \beta \in \mathbb{F}$, *let* $(\mathsf{Com}_b, \mathsf{op}_b) \leftarrow \mathsf{VC.Commit}(\mathsf{pp_{VC}}, V_b)$ *for* $b \in \{0,1\}$. *There are efficient algorithms deriving* $\widehat{\mathsf{Com}} := \alpha\mathsf{Com}_0 + \beta\mathsf{Com}_1$ *and opening information* $\widehat{\mathsf{op}}$ *such that* $(\alpha V_0 + \beta V_1, \widehat{\mathsf{op}})$ *is a valid opening of* $\widehat{\mathsf{Com}}$.

4.2 Security Models and Zero-Knowledge Arguments

Let $\mathcal{X}$ and $\mathcal{W}$ be an instance space and a witness space, respectively, and let $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{W}$ be an NP relation. The language induced by $\mathcal{R}$ is $\mathcal{L}_{\mathcal{R}} := \{x \in \mathcal{X} \mid \exists w \in \mathcal{W} : (x, w) \in \mathcal{R}\}$. We call $x \in \mathcal{X}$ an *instance*, or equivalently a *statement*, and call $w \in \mathcal{W}$ a *witness* for x if $(x, w) \in \mathcal{R}$.

Arguments and arguments of knowledge. An interactive argument for $\mathcal{R}$ is a protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. Completeness requires $\mathcal{V}$ to accept when the prover follows the protocol using a valid witness. Computational soundness requires that no PPT prover make $\mathcal{V}$ accept a statement outside $\mathcal{L}_{\mathcal{R}}$, except with negligible probability.

An interactive argument is an *argument of knowledge* if it additionally satisfies *knowledge soundness*: whenever a PPT prover makes the verifier accept, then there exists an efficient extractor that outputs a witness for the accepted statement, except with negligible probability.

All interactive arguments used in our construction are *public coin*: every verifier message consists of public random challenges. We apply the Fiat–Shamir transform to obtain non-interactive arguments in the programmable ROM. We provide formal definitions in Definition 2.

Structured reference string. A structured reference string (SRS) consists of public elements with algebraic structure determined by hidden setup randomness.

In the simulated experiment, the simulator may generate a public SRS together with a simulation trapdoor that is not given to the adversary. Indistinguishability of the honest and simulated setup distributions is included in the zero-knowledge experiment below.

Programmable random oracle model. In the random oracle model (ROM) [8], all parties have oracle access to a uniformly sampled function $\mathcal{H} : \{0,1\}^* \to \mathcal{Y}$ for a finite range $\mathcal{Y}$. The oracle is evaluated by lazy sampling: its answer to a fresh query is sampled uniformly from $\mathcal{Y}$, recorded, and returned consistently on every repeated query.

In the programmable ROM [58], the simulator may assign an output $\mathcal{H}(s) := a$ at an input s that has not previously been answered. Once an answer has been returned, it is never changed. Honest parties and the adversary receive only ordinary oracle access.

We apply the Fiat–Shamir transform [27] to the public-coin interactive arguments used by our construction. Each verifier challenge is replaced by the output of a random oracle $\mathcal{H}$. This produces a non-interactive argument whose security is analyzed in the programmable ROM.

When a concrete instantiation uses the ROM, the algorithms have access to a common random oracle. This access is implicit in the notation below; programming is available only to the simulators in the simulation experiment.

Algebraic group model. We analyze knowledge soundness of the group-based arguments in the algebraic group model (AGM) [31]. An adversary is *algebraic* if, whenever it outputs a group element, it also outputs coefficients expressing that element as a linear combination of the group elements it has previously received. In a bilinear setting, this requirement is applied separately in each group. For AGM-based instantiations, the extractor also receives the algebraic representations associated with the recorded execution, including those of the statement commitments. We leave this input implicit; it is not part of the proof.

Definition 2 (Zero-Knowledge Argument of Knowledge)**.** *Let $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{W}$ be an* NP *relation. A zero-knowledge argument of knowledge for $\mathcal{R}$ is a tuple of PPT algorithms $\Pi = (\Pi.\mathsf{Setup}, \Pi.\mathsf{Prove}, \Pi.\mathsf{Verify})$ defined as follows:*

- $\mathsf{pp}_\Pi \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$*: takes as input the security parameter λ and outputs public parameters pp_Π.*
- $\pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{pp}_\Pi, x, w)$*: takes as input pp_Π, a statement x, and a witness w such that $(x, w) \in \mathcal{R}$, and outputs a proof π.*
- $b \leftarrow \Pi.\mathsf{Verify}(\mathsf{pp}_\Pi, x, \pi)$*: takes as input pp_Π, a statement x, and a proof π, and outputs a bit $b \in \{0,1\}$.*

The argument satisfies the following properties.

- ***Completeness:*** *For every $(x, w) \in \mathcal{R}$,*

$$\Pr\left[\Pi.\mathsf{Verify}(\mathsf{pp}_\Pi, x, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp}_\Pi \leftarrow \Pi.\mathsf{Setup}(1^\lambda), \\ \pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{pp}_\Pi, x, w) \end{array}\right] = 1.$$

- ***Knowledge Soundness:*** *For every n.u. stateful PPT adversary* Adv*, there exists a PPT extractor* Ext_Π *such that,*

$$\Pr\left[\begin{array}{l|l} \Pi.\mathsf{Verify}(\mathsf{pp}_\Pi, x, \pi) = 1 & \mathsf{pp}_\Pi \leftarrow \Pi.\mathsf{Setup}(1^\lambda), \\ \wedge (x,w) \notin \mathcal{R} & (x,\pi) \leftarrow \mathsf{Adv}(\mathsf{pp}_\Pi), \\ & w \leftarrow \mathsf{Ext}_\Pi(\mathsf{pp}_\Pi, x, \pi) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

- ***Zero Knowledge:*** *There exist stateful PPT simulators* Sim_1 *and* Sim_2 *defined as follows:*
  - $(\widetilde{\mathsf{pp}}_\Pi, \tau_\Pi) \leftarrow \mathsf{Sim}_1(1^\lambda)$*, where* $\widetilde{\mathsf{pp}}_\Pi$ *are the simulated public parameters and* τ_Π *contains the simulation trapdoor.*
  - $\widetilde{\pi} \leftarrow \mathsf{Sim}_2(\widetilde{\mathsf{pp}}_\Pi, \tau_\Pi, x)$*, where* Sim_2 *generates a simulated proof for the statement* x *without receiving a witness.*

  For every n.u. stateful PPT adversary Adv *that, on receiving the public parameters of the corresponding experiment, outputs a valid statement–witness pair* $(x,w) \in \mathcal{R}$*,*

$$\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \mathsf{pp}_\Pi \leftarrow \Pi.\mathsf{Setup}(1^\lambda), \\ (x,w) \leftarrow \mathsf{Adv}(\mathsf{pp}_\Pi), \\ \pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{pp}_\Pi, x, w), \\ b \leftarrow \mathsf{Adv}(\mathsf{pp}_\Pi, x, \pi) \end{array}\right] - \Pr\left[\widetilde{b} = 1 \;\middle|\; \begin{array}{l} (\widetilde{\mathsf{pp}}_\Pi, \tau_\Pi) \leftarrow \mathsf{Sim}_1(1^\lambda), \\ (x,w) \leftarrow \mathsf{Adv}(\widetilde{\mathsf{pp}}_\Pi), \\ \widetilde{\pi} \leftarrow \mathsf{Sim}_2(\widetilde{\mathsf{pp}}_\Pi, \tau_\Pi, x), \\ \widetilde{b} \leftarrow \mathsf{Adv}(\widetilde{\mathsf{pp}}_\Pi, x, \widetilde{\pi}) \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

- ***Succinctness:*** *The size of the output* π *of* $\Pi.\mathsf{Prove}$ *is* $\mathsf{poly}(\lambda, |x|)$*. The runtime of* $\Pi.\mathsf{Verify}$ *is* $\mathsf{poly}(\lambda, |x|, \log|w|)$*.*

Bounded adaptive zero knowledge. For repeated use, we consider the same experiment with one setup and a publicly bounded number of adaptive proof queries. On each valid input $(x,w) \in \mathcal{R}$, the real oracle returns $\Pi.\mathsf{Prove}(\mathsf{pp}_\Pi, x, w)$, while the simulated oracle returns $\mathsf{Sim}_2(\widetilde{\mathsf{pp}}_\Pi, \tau_\Pi, x)$. The simulator retains its internal state across calls, and the complete views must remain computationally indistinguishable.

Model-specific guarantees. In an SRS-based instantiation, the public parameters contain the SRS. In the programmable ROM, the algorithms have ordinary access to a common random oracle, and the simulators may program it at unanswered inputs. We say that knowledge soundness holds against algebraic adversaries if the knowledge-soundness requirement holds for every n.u. stateful PPT algebraic adversary Adv.

4.3 Lookup Arguments

Lookup arguments allow a prover to show that every entry of a committed vector $\mathbf{v}$ appears in a preprocessed, and possibly committed, table $\mathbf{t}$.

Definition 3 (Lookup Argument, adopted from [67])**.** *Given commitments* $\mathsf{Com}_\mathbf{v}$ *and* $\mathsf{Com}_\mathbf{t}$ *to* $\mathbf{v} \in \mathbb{F}^m$ *and* $\mathbf{t} \in \mathbb{F}^N$*, respectively, a lookup argument proves that the prover knows an opening* $\mathbf{v} = (v_1, \ldots, v_m)$ *of* $\mathsf{Com}_\mathbf{v}$ *such that, for every* $i \in [m]$*, there exists* $j \in [N]$ *for which* $v_i = t_j$*.*

Logarithmic Derivative Argument (LogUp). To certify multiset consistency, we use the Logarithmic Derivative Argument, commonly referred to as LogUp [41]. We treat LogUp as a black-box subprotocol $\Pi_{\mathsf{LogUp}} = (\Pi_{\mathsf{LogUp}}.\mathsf{Prove}, \Pi_{\mathsf{LogUp}}.\mathsf{Verify})$ for proving rational identities over committed polynomials.

Let $\mathbf{v} \in \mathbb{F}^m$ be a query vector, let $\mathbf{t} \in \mathbb{F}^N$ be a table, and let $\mathsf{mult} \in \mathbb{F}^N$ be a multiplicity vector, where each query entry is assigned to one matching table position and $\mathsf{mult}[j]$ counts the entries assigned to position j.

For a verifier challenge $\beta \xleftarrow{\$} \mathbb{F}$, multiset consistency is characterized by the identity

$$\sum_{i=1}^{m} \frac{1}{\beta - \mathbf{v}[i]} = \sum_{j=1}^{N} \frac{\mathsf{mult}[j]}{\beta - \mathbf{t}[j]}.$$

Thus, LogUp provides a succinct way to certify multiset consistency under commitments by proving the above rational identity.

cq-based Lookup Arguments. The cq lookup argument [23] uses the logarithmic-derivative identity above and reduces prover work through table-dependent preprocessing and cached quotients.

Let $\mathbb{V} \subset \mathbb{F}$ be a multiplicative subgroup of order N, and let $m(X)$, $T(X)$, and $A(X)$ denote the Lagrange interpolation polynomials of $(\mathsf{mult}[i])_{i=1}^N$, $\mathbf{t}$, and $(\mathsf{mult}[i]/(\beta - \mathbf{t}[i]))_{i=1}^N$, respectively. The core consistency constraint enforced by cq is

$$A(X)\big(\beta - T(X)\big) - m(X) = Q_A(X) Z_{\mathbb{V}}(X),$$

where $Z_{\mathbb{V}}$ is the vanishing polynomial of $\mathbb{V}$.

To accelerate quotient computation, cq performs a one-time table-dependent preprocessing phase. For each Lagrange basis polynomial $L_i(X)$ over $\mathbb{V}$, define $Q_i(X)$ by

$$L_i(X)T(X) = \mathbf{t}[i]L_i(X) + Q_i(X)Z_{\mathbb{V}}(X),$$

and cache the corresponding polynomial commitments.

We use zkcq+ [12] to obtain zero knowledge for both the private table and the lookup values.

To support updates efficiently, we use the Base+Cache framework of Dutta et al. [21]. A preprocessed base table is retained as the *Base*, while recent updates form the *Cache*. Lookup proofs reuse the Base preprocessing and incorporate the Cache through an additional sparse correction, avoiding a full recomputation of the table-dependent preprocessing after every update.

5 Defining Sunshine Systems

We now formally define Sunshine Systems that operate in two phases: an *online query phase* and an *audit phase*.

5.1 Model

For a cryptographic formalization of a Sunshine System, we must first specify how documents are represented and stored, as well as how queries are parsed.

Documents and Keywords. We assume that each document is assigned a unique identifier. Let $\mathcal{ID}$ denote the finite set of possible document identifiers. We further assume that each document is associated with a set of keywords that capture its relevant content. Let $\mathcal{K}$ denote the finite set of possible keywords. We will represent each document as a pair $(\mathsf{id}, \mathcal{W})$, where $\mathsf{id} \in \mathcal{ID}$ is its identifier and $\mathcal{W} \subseteq \mathcal{K}$ is the set of keywords associated with the document. We assume that the sets $\mathcal{ID}$ and $\mathcal{K}$ are publicly known.

Definition 4 (Well-Formed Document Set)**.** *We say that a set of documents $\mathcal{D}$ is* well-formed *if it is of the form*

$$\mathcal{D} = \{(\mathsf{id}_1, \mathcal{W}_1), \ldots, (\mathsf{id}_M, \mathcal{W}_M)\},$$

where each document identifier $\mathsf{id}_i \in \mathcal{ID}$ and each keyword set $\mathcal{W}_i \subseteq \mathcal{K}$. In particular, since each $\mathcal{W}_i$ is a set, it contains no duplicate keywords.[3] *Moreover, all document identifiers are distinct; that is, for each $i \neq j$, $\mathsf{id}_i \neq \mathsf{id}_j$.*

Queries. We will represent each query as a set of keywords and assume that $c \in \mathbb{N}$ is a public upper bound on the number of keywords that may appear in a query. That is,

$$\mathcal{Q} = \{q_1, \ldots, q_\ell\} \subseteq \mathcal{K}, \qquad 1 \leq \ell \leq c,$$

where $q_u \neq q_v$ for each distinct $u, v \in [\ell]$. Given a set of documents $\mathcal{D}$, the correct response to a query $\mathcal{Q}$ is the set of identifiers corresponding to all documents whose keyword sets contain every keyword appearing in $\mathcal{Q}$. Formally,

$$\mathsf{Res}_{\mathcal{D}}(\mathcal{Q}) := \Big\{ \mathsf{id}_i \;\Big|\; \{q_1, \ldots, q_\ell\} \subseteq \mathcal{W}_i \wedge (\mathsf{id}_i, \mathcal{W}_i) \in \mathcal{D} \Big\}.$$

Raw Document Files $\leftrightarrow$ Identifiers. We assume that each raw document file is hashed using a salted collision-resistant hash function, with each full hash associated with a distinct identifier in $\mathcal{ID}$. We assume that this association is fixed when the document is produced, authenticated together with the database commitment, and preserved across updates. This implicitly assumes that there exists a canonical, digital representation of the documents in question, e.g., a PDF scan of a physical form or a digitally native representation. When completing a query response, the agency returns the corresponding raw documents and salts, allowing the verifier to check their hashes against the identifier–hash associations. In our formal treatment of the system, we abstract this process away, allowing the remainder of the paper—including our formal definition of Sunshine Systems and our construction—to focus only on the identifier-and-keyword representation of documents.

5.2 Online Query Phase

At a high level, a basic Sunshine System operates as follows. First, the agency commits to an initial set of documents. This commitment must be both hiding and binding. Subsequently, query issuers may submit queries to the agency and receive both a response and a verifiable proof certifying that the response was generated correctly with respect to the committed document set. This constitutes the *online query phase* in Sunshine Systems.

Definition 5 (Sunshine System: Online Query Phase)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, $\mathcal{K}$ be a set of keywords and $\mathcal{ID}$ be a set of identifiers. We fix a public polynomially bounded query limit q_{max}, included in the public parameters* pp*, for responses under the same commitment. The online query phase in a Sunshine System comprises a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Response}, \mathsf{Verify})$ *defined as follows:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$*: This algorithm takes the security parameter 1^λ as input and outputs public parameters* pp*.*
- $\mathsf{Commit}(\mathsf{pp}, \mathcal{D}) \to (\mathsf{com}, \mathsf{op})$*: This algorithm takes the public parameters* pp*, and a set of documents $\mathcal{D}$ as input and outputs a commitment* com *along with its opening* op*.*

[3] We additionally require $\mathcal{W}_i \neq \emptyset$ for every document. A raw document to which no keyword is assigned does not appear in the document set.

– $\mathsf{Response}(\mathsf{pp}, \mathsf{com}, \mathsf{op}, \mathcal{D}, \mathcal{Q}) \to (\mathcal{J}, \pi)$: *This algorithm takes the public parameters* pp*, a query* $\mathcal{Q} \subseteq \mathcal{K}$*, a commitment* com *along with the corresponding set of documents* $\mathcal{D}$ *and opening* op *as input and outputs a response* $\mathcal{J} \subseteq \mathcal{ID}$ *and a proof* π*.*

– $\mathsf{Verify}(\mathsf{pp}, \mathsf{com}, \mathcal{Q}, \mathcal{J}, \pi) \to b$*: This algorithm takes the public parameters* pp*, a commitment* com*, a query* $\mathcal{Q} \subseteq \mathcal{K}$*, and the corresponding response* $\mathcal{J} \subseteq \mathcal{ID}$ *and proof* π *as input and outputs a bit* $b \in \{0, 1\}$*.*

These algorithms must satisfy the following properties:

1. ***Completeness:*** *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. For any well-formed document set (see Definition 4)* $\mathcal{D}$*, and any query* $\mathcal{Q} \subseteq \mathcal{K}$ *with* $1 \leq |\mathcal{Q}| \leq c$*, it holds that*

$$\Pr\left[\mathsf{Verify}(\mathsf{pp}, \mathsf{com}, \mathcal{Q}, \mathcal{J}, \pi) = 1 \;\middle|\; \begin{array}{l}(\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}) \\ (\mathcal{J}, \pi) \leftarrow \mathsf{Response}(\mathsf{pp}, \mathsf{com}, \mathsf{op}, \mathcal{D}, \mathcal{Q})\end{array}\right] = 1.$$

2. ***Knowledge Soundness:*** *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. For every n.u. stateful PPT adversary* $\mathsf{Adv} = (\mathsf{Adv}_1, \mathsf{Adv}_2)$*, there exists a PPT extractor* Ext*, such that for any query* $\mathcal{Q} \subseteq \mathcal{K}$*, if* $\mathsf{com}^* \leftarrow \mathsf{Adv}_1(\mathsf{pp})$ *and* $(\mathcal{J}^*, \pi^*) \leftarrow \mathsf{Adv}_2(\mathsf{pp}, \mathcal{Q})$*, then it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{Verify}(\mathsf{pp}, \mathsf{com}^*, \mathcal{Q}, \mathcal{J}^*, \pi^*) = 1 \\ \wedge \big((\mathsf{Res}_{\mathcal{D}}(\mathcal{Q}) \neq \mathcal{J}^*) \\ \quad \vee (\mathcal{D} \textit{ is not well-formed}) \\ \quad \vee (\mathsf{op}^* \textit{ is not a valid opening of } \mathsf{com}^* \textit{ to } \mathcal{D})\big)\end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $(\mathcal{D}, \mathsf{op}^*) \leftarrow \mathsf{Ext}(\mathsf{pp}, \mathsf{com}^*, \mathcal{Q}, \mathcal{J}^*, \pi^*)$*.*

3. ***Hiding Commitment:*** *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. For any two distinct well-formed document sets* $\mathcal{D}_0$ *and* $\mathcal{D}_1$ *such that the total number of keywords, summed across all documents in the set, is the same for both* $\mathcal{D}_0$ *and* $\mathcal{D}_1$*,* [4] *it holds that*

$$\{(\mathsf{pp}, \mathsf{com}_0) \mid (\mathsf{com}_0, \mathsf{op}_0) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}_0)\} \approx_c \{(\mathsf{pp}, \mathsf{com}_1) \mid (\mathsf{com}_1, \mathsf{op}_1) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}_1)\}.$$

4. ***Zero Knowledge:*** *There exist stateful PPT simulators* $\mathsf{Sim}_{\mathsf{online},1}$ *and* $\mathsf{Sim}_{\mathsf{online},2}$ *such that, for any well-formed document set* $\mathcal{D}$ *and every n.u. PPT adversary* Adv*, it holds that*

$$\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \\ (\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}), \\ b \leftarrow \mathsf{Adv}^{\mathcal{O}_{\mathsf{real}}}(\mathsf{pp}, \mathsf{com})\end{array}\right]\right.$$
$$\left. - \Pr\left[\widetilde{b} = 1 \;\middle|\; \begin{array}{l}(\widetilde{\mathsf{pp}}, \tau_{\mathsf{online}}) \leftarrow \mathsf{Sim}_{\mathsf{online},1}(1^\lambda), \\ (\widetilde{\mathsf{com}}, \cdot) \leftarrow \mathsf{Commit}(\widetilde{\mathsf{pp}}, \mathcal{D}), \\ \widetilde{b} \leftarrow \mathsf{Adv}^{\mathcal{O}_{\mathsf{sim}}}(\widetilde{\mathsf{pp}}, \widetilde{\mathsf{com}})\end{array}\right]\right| \leq \mathsf{negl}(\lambda),$$

where Adv *may make at most* q_{max} *adaptive queries* $\mathcal{Q} \subseteq \mathcal{K}$ *with* $1 \leq |\mathcal{Q}| \leq c$*, including repeated queries. The oracles use the values sampled in their respective experiment as follows:*

– $\mathcal{O}_{\mathsf{real}}(\mathcal{Q})$ *returns* $(\mathcal{J}, \pi_{\mathsf{online}}) \leftarrow \mathsf{Response}(\mathsf{pp}, \mathsf{com}, \mathsf{op}, \mathcal{D}, \mathcal{Q})$*.*

– $\mathcal{O}_{\mathsf{sim}}(\mathcal{Q})$ *obtains* $\mathcal{J} := \mathsf{Res}_{\mathcal{D}}(\mathcal{Q})$*, computes* $\widetilde{\pi_{\mathsf{online}}} \leftarrow \mathsf{Sim}_{\mathsf{online},2}(\widetilde{\mathsf{pp}}, \tau_{\mathsf{online}}, \widetilde{\mathsf{com}}, \mathcal{Q}, \mathcal{J})$*, and returns* $(\mathcal{J}, \widetilde{\pi_{\mathsf{online}}})$*.*

[4] This somewhat cumbersome formulation is necessitated by our constructions in Sections 6 and 7. Ideally, one could ask for a stronger hiding notion. However, in our constructions, the commitment reveals the total number of keywords across all documents in the committed set. While this makes the definition slightly less clean, we do not expect this aggregate size information to reveal any privacy-sensitive information, and hence we adopt this weaker notion of hiding.

The same commitment is used throughout each experiment. The oracles retain their execution records between calls, including any reused auxiliary commitments and proofs. The simulator $\mathsf{Sim}_{\mathsf{online},2}$ *retains its internal records and receives neither* $\mathcal{D}$ *nor its commitment opening.*

5. ***Succinctness:*** *The size of the output* π *of* Response *is* $\mathsf{poly}(\lambda, |\mathcal{Q}|)$*. The runtime of verification is* $\mathsf{poly}(\lambda, |\mathcal{Q}|, |\mathcal{J}|, \log |\mathcal{D}|)$*.*

5.3 Audit Phase

A Sunshine System is not static. As new documents are created, agencies must update their existing commitments to reflect the expanded set of documents. While an agency could simply compute a fresh commitment to the updated set of documents, doing so creates an opportunity for a dishonest agency to modify or remove previously committed documents. To mitigate this risk, we incorporate a *privacy-preserving* audit phase into the Sunshine System. This phase enables auditors to verify that a new commitment corresponds to the previously committed document set together with a collection of newly added documents. As discussed earlier, auditors may be the same entities as the query issuers or could be separate. The schedules for commitment publication and Audit verification are deployment choices; the protocol does not impose a maximum delay for either.

Definition 6 (Sunshine System: Audit Phase)**.** *Let* $\lambda \in \mathbb{N}$ *be the security parameter, let* $\mathcal{K}$ *be a set of keywords, and let* $\mathcal{ID}$ *be a set of identifiers. Let* Setup *and* Commit *be as defined in Definition 5. The Audit Phase in a Sunshine System comprises two additional PPT algorithms* (Update, Audit) *defined as follows:*

- $\mathsf{Update}(\mathsf{pp}, \mathcal{D}, \mathsf{com}, \mathsf{op}, \Delta, \mathsf{com}^+, \mathsf{op}^+) \to \pi_{\mathsf{audit}}$*: This algorithm takes the public parameters* pp*, an initial set of documents* $\mathcal{D}$ *together with its commitment* com *and opening* op*, a document set* Δ *to be appended to* $\mathcal{D}$*, and a commitment* com^+ *and opening* op^+ *to the combined set* $(\mathcal{D} \cup \Delta)$ *as input, and outputs an Audit proof* π_{audit}*.*[5]
- $\mathsf{Audit}(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}}) \to b$*: This algorithm takes the public parameters* pp*, initial and new commitments* $\mathsf{com}, \mathsf{com}^+$ *and an Audit proof* π_{audit} *as input, and outputs a bit* $b \in \{0, 1\}$*.*

These algorithms must satisfy the following properties:

1. ***Completeness:*** *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. For any well-formed document sets* $\mathcal{D}$ *and* Δ *such that* $\mathcal{D} \cup \Delta$ *is well-formed and* $\nexists\, \mathsf{id}, \mathcal{W}, \mathcal{W}' : (\mathsf{id}, \mathcal{W}) \in \mathcal{D} \wedge (\mathsf{id}, \mathcal{W}') \in \Delta$*, let* $(\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D})$ *and* $(\mathsf{com}^+, \mathsf{op}^+) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D} \cup \Delta)$*.*[6] *Then it holds that*

$$\Pr\left[\mathsf{Audit}(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}}) = 1 \;\middle|\; \pi_{\mathsf{audit}} \leftarrow \mathsf{Update}(\mathsf{pp}, \mathcal{D}, \mathsf{com}, \mathsf{op}, \Delta, \mathsf{com}^+, \mathsf{op}^+)\right] = 1.$$

2. ***Knowledge Soundness:*** *Let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*. For every n.u. stateful PPT adversary* Adv*,*

[5] We use the term *Audit proof* for the complete transcript produced by Update; this transcript may contain auxiliary commitments and several component proofs rather than one standalone zero-knowledge proof.

[6] We leave the private choice of representation implicit in the notation $\mathsf{Commit}(\mathsf{pp}, \mathcal{D})$. In the honest Audit experiments, the new representation retains the old representation's non-padding entries in place and appends the entries induced by Δ, using fresh independent commitment randomness.

there exists a PPT extractor Ext*, such that if* $(\mathsf{com}^*, \mathsf{com}^{+*}, \pi^*_{\mathsf{audit}}) \leftarrow \mathsf{Adv}(\mathsf{pp})$*, then it holds that*

$$\Pr\left[\begin{array}{l} \mathsf{Audit}(\mathsf{pp}, \mathsf{com}^*, \mathsf{com}^{+*}, \pi^*_{\mathsf{audit}}) = 1 \wedge \\ \big((\mathcal{D} \textit{ is not well-formed}) \vee \\ (\Delta \textit{ is not well-formed}) \vee \\ (\mathcal{D} \cup \Delta \textit{ is not well-formed}) \vee \\ (\exists \mathsf{id}, \mathcal{W}, \mathcal{W}' : (\mathsf{id}, \mathcal{W}) \in \mathcal{D} \wedge (\mathsf{id}, \mathcal{W}') \in \Delta) \vee \\ (\mathsf{op}^* \textit{ is not a valid opening of } \mathsf{com}^* \textit{ to } \mathcal{D}) \vee \\ (\mathsf{op}^{+*} \textit{ is not a valid opening of } \mathsf{com}^{+*} \textit{ to } \mathcal{D} \cup \Delta)\big) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $(\mathcal{D}, \Delta, \mathsf{op}^*, \mathsf{op}^{+*}) \leftarrow \mathsf{Ext}(\mathsf{pp}, \mathsf{com}^*, \mathsf{com}^{+*}, \pi^*_{\mathsf{audit}})$.

3. ***Zero Knowledge:*** *There exist PPT simulators* $\mathsf{Sim}_{\mathsf{Audit},1}$ *and* $\mathsf{Sim}_{\mathsf{Audit},2}$ *such that, for any well-formed document sets* $\mathcal{D}, \Delta$ *for which* $\mathcal{D} \cup \Delta$ *is well-formed and* $\nexists\, \mathsf{id}, \mathcal{W}, \mathcal{W}' : (\mathsf{id}, \mathcal{W}) \in \mathcal{D} \wedge (\mathsf{id}, \mathcal{W}') \in \Delta$*, and every n.u. PPT adversary* Adv*, it holds that*

$$\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \\ (\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}), \\ (\mathsf{com}^+, \mathsf{op}^+) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D} \cup \Delta), \\ \pi_{\mathsf{audit}} \leftarrow \mathsf{Update}(\mathsf{pp}, \mathcal{D}, \mathsf{com}, \mathsf{op}, \Delta, \mathsf{com}^+, \mathsf{op}^+), \\ b \leftarrow \mathsf{Adv}(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}}) \end{array}\right]\right.$$
$$\left. - \Pr\left[\widetilde{b} = 1 \;\middle|\; \begin{array}{l} (\widetilde{\mathsf{pp}}, \tau_{\mathsf{Audit}}) \leftarrow \mathsf{Sim}_{\mathsf{Audit},1}(1^\lambda), \\ (\widetilde{\mathsf{com}}, \cdot) \leftarrow \mathsf{Commit}(\widetilde{\mathsf{pp}}, \mathcal{D}), \\ (\widetilde{\mathsf{com}}^+, \cdot) \leftarrow \mathsf{Commit}(\widetilde{\mathsf{pp}}, \mathcal{D} \cup \Delta), \\ \widetilde{\pi_{\mathsf{audit}}} \leftarrow \mathsf{Sim}_{\mathsf{Audit},2}(\widetilde{\mathsf{pp}}, \tau_{\mathsf{Audit}}, \widetilde{\mathsf{com}}, \widetilde{\mathsf{com}}^+), \\ \widetilde{b} \leftarrow \mathsf{Adv}(\widetilde{\mathsf{pp}}, \widetilde{\mathsf{com}}, \widetilde{\mathsf{com}}^+, \widetilde{\pi_{\mathsf{audit}}}) \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

Remark 1 (Incremental efficiency)**.** *Generating the first* π_{audit} *may require processing the entire committed document set. For each subsequent accepted transition, however, the Audit cost may depend on the newly appended document set and on fixed public capacity parameters, but should remain independent of the number of previously accepted updates. We provide concrete complexity bounds for our construction in Section B.2.*

Remark 2 (Standalone vs. sequential audit)**.** *Definition 6 describes a standalone Audit invocation, in which* π_{audit} *contains all information necessary to prove that* com *opens to a well-formed document set and that* com^+ *is obtained by a valid append. In a sequential execution, the auditor retains the accepted Audit record for* com*; the next invocation proves only the update from* com *to* com^+*. Our construction makes this state reuse explicit in Section 7.*

Remark 3 (Queries after an audit)**.** *Each honest update uses fresh independent commitment randomness. The Audit phase preserves hiding of the updated commitment, which can then be used for further Online queries under the zero-knowledge guarantee of Definition 5, up to the public query bound* q_{max}*. See Section B.1 for the simulation of an honest Audit followed by adaptive queries.*

6 Instantiating the Online Query Phase

We now present an efficient realization of the online query phase (see Definition 5). We describe the document set representation and commitment. Section 6.2 defines the relations and their composition. Concrete instantiations and their cost analysis are given in Sections A.2 and A.3.

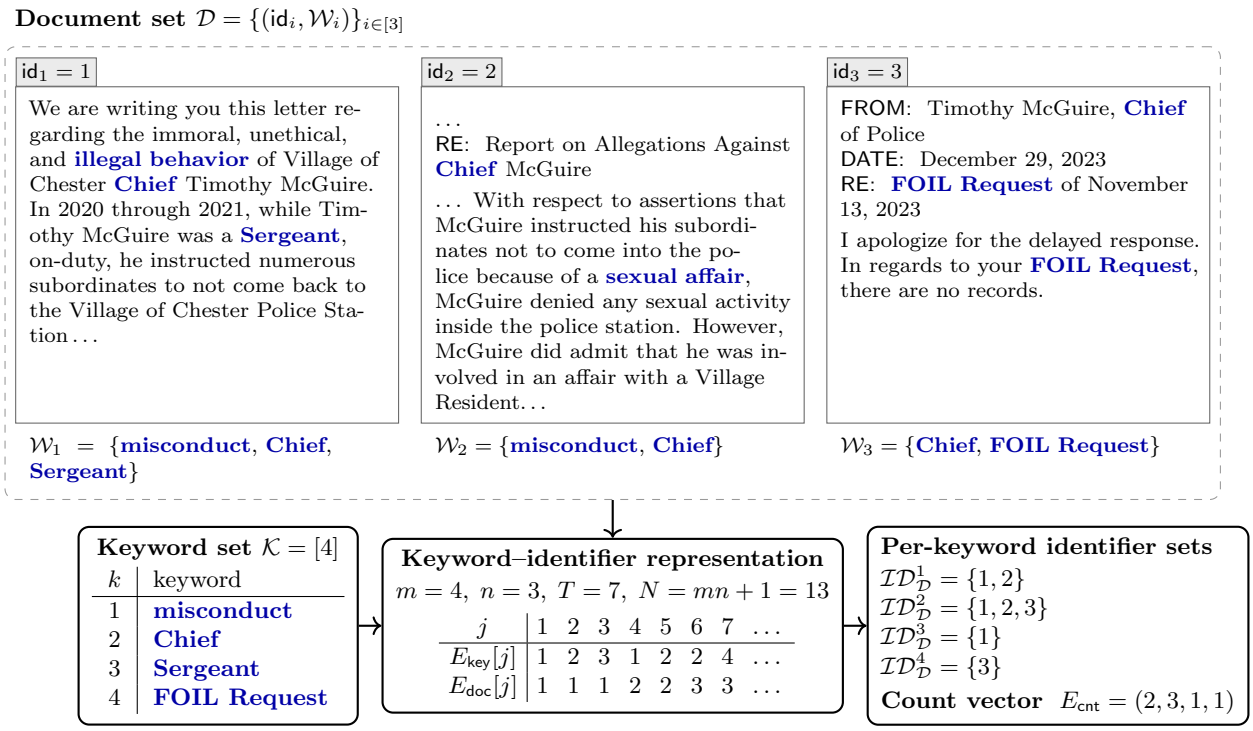


Figure 1: Example of document-set representation, built from records of the Village of Chester, NY police department investigation discussed in Section 1 [70]. The first two documents contain verbatim excerpts from relevant documents [3, 59], while the third is reconstructed from *Sussman v. Village of Chester* [71]. Each document is assigned an id_i and a set of keywords $\mathcal{W}_i$ (highlighted), potentially just extracting exact words from text or by creating fuzzy buckets (e.g., grouping *illegal behavior* and *sexual affair* under *misconduct*). A query returns every identifier whose keyword set contains all queried keywords, e.g., $\mathsf{Res}_{\mathcal{D}}(\{1, 2\}) = \mathcal{ID}^1_{\mathcal{D}} \cap \mathcal{ID}^2_{\mathcal{D}} = \{1, 2\}$.

6.1 Committing to a Document Set

Using integer labels for the sets in Section 5.1, let the set of document identifiers be $\mathcal{ID} = [n]$ and the keyword set be $\mathcal{K} = [m]$. We encode vectors over a prime field $\mathbb{F}$ and reserve the field element 0 for padding, so $0 \notin \mathcal{ID} \cup \mathcal{K}$.

Representation. Recall from Section 5.1 that each document is represented as a pair $(\mathsf{id}, \mathcal{W})$, where $\mathsf{id} \in \mathcal{ID}$ is its identifier and $\mathcal{W} \subseteq \mathcal{K}$ is its set of keywords. For our protocol, we represent $(\mathsf{id}, \mathcal{W})$ by the set of keyword–identifier pairs $\{(k, \mathsf{id}) : k \in \mathcal{W}\}$. Thus, $\mathcal{D}$ can equivalently be viewed as a collection of keyword–identifier pairs. We store this representation as follows:

- *Keyword and Document Vectors.* We store all keyword–identifier pairs in vectors $E_{\mathsf{key}}, E_{\mathsf{doc}} \in \mathbb{F}^N$ of fixed public capacity N. The entries $(E_{\mathsf{key}}[j], E_{\mathsf{doc}}[j])$ represent keyword–identifier pairs. Since a well-formed document set contains at most mn distinct keyword–identifier pairs, we choose a capacity $N \geq mn + 1$.[7] If $\mathcal{D}$ contains T such pairs, positions $1, \ldots, T$ contain exactly these pairs, while positions $T + 1, \ldots, N$ are padded with $(0, 0)$. Thus $T < N$, and the final row is always $(0, 0)$.[8] The public value T reveals only the total number of keyword–identifier pairs.

[7] An implementation may use a smaller public capacity when its supported document sets contain fewer than N keyword–identifier pairs.

[8] The role of this reserved row in the membership argument is explained in Section A.2.5.

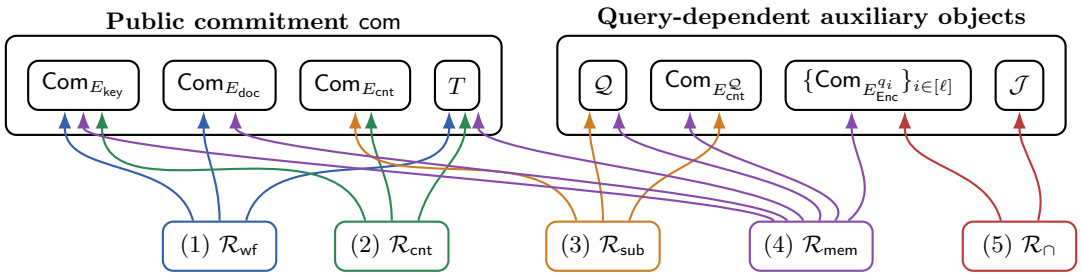


Figure 2: Overview of the five relations underlying the online query protocol. The query-independent relations $\mathcal{R}_{\mathsf{wf}}$ and $\mathcal{R}_{\mathsf{cnt}}$ certify the well-formedness of the committed document-set representation and the consistency of its count vector. The query-dependent relations $\mathcal{R}_{\mathsf{sub}}$, $\mathcal{R}_{\mathsf{mem}}$, and $\mathcal{R}_{\cap}$ certify the query-specific counts, the per-keyword identifier sets, and their final intersection.

- *Count Vector.* The vector $E_{\mathsf{cnt}} \in \mathbb{F}^m$ records, for each keyword $k \in \mathcal{K}$, the number of documents in $\mathcal{D}$ containing k. Equivalently, $E_{\mathsf{cnt}}[k] := \big|\{j \in [T] : E_{\mathsf{key}}[j] = k\}\big|$.

See Figure 1 for an example.

Commitment. After constructing E_{key}, E_{doc}, and E_{cnt}, the commit algorithm commits to each vector using a hiding and subvector-binding vector commitment scheme (see Definition 1 and Section A.1). The resulting public commitment is $\mathsf{com} = \big(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T\big)$, which binds the prover to the above representation of $\mathcal{D}$.

6.2 Query Response and Verification

Recall from Section 5.1 that, on a query $\mathcal{Q} \subseteq \mathcal{K}$, the agency returns a candidate response $\mathcal{J} \subseteq \mathcal{ID}$ and a proof that $\mathcal{J} = \mathsf{Res}_{\mathcal{D}}(\mathcal{Q})$, where $\mathcal{D}$ is a well-formed document set represented by $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T)$ and bound by com. We establish this statement through five simpler relations, defined in Section 6.2.1. In Section 6.2.2, we compose zero-knowledgearguments for these relations into an online query protocol satisfying Definition 5.

6.2.1 Relations

We define the five relations below, beginning with the two that depend only on the committed document set. Their dependencies are illustrated in Figure 2. All relations use the public vector lengths and evaluation domains; validity of every witness opening is part of the corresponding relation.

Query-independent relations. The following two relations certify the committed document-set representation and its keyword counts, independently of the query.

The "well formed" relation $\mathcal{R}_{\mathsf{wf}}$. The relation $\mathcal{R}_{\mathsf{wf}}$ checks that the committed pair vectors encode a well-formed document set (see Definition 4):

– **Statement:** $x_{\mathsf{wf}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$.

– **Witness:** $w_{\mathsf{wf}} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}})$.

– **Relation:** We define $\mathcal{R}_{\mathsf{wf}}(x_{\mathsf{wf}}, w_{\mathsf{wf}}) = 1$ if:

1. *Valid commitments:* $(E_{\mathsf{key}}, \mathsf{op}_{E_{\mathsf{key}}})$ and $(E_{\mathsf{doc}}, \mathsf{op}_{E_{\mathsf{doc}}})$ are valid openings of $\mathsf{Com}_{E_{\mathsf{key}}}$ and $\mathsf{Com}_{E_{\mathsf{doc}}}$, respectively.

2. *Valid nonzero entries:* For every $j \in [T]$, $E_{\mathsf{key}}[j] \in \mathcal{K}$ and $E_{\mathsf{doc}}[j] \in \mathcal{ID}$.
3. *Zero padding:* For every $j \in \{T+1, \ldots, N\}$, $E_{\mathsf{key}}[j] = E_{\mathsf{doc}}[j] = 0$.
4. *No duplicate keyword–identifier pairs:* For every distinct $j, j' \in [T]$, $E_{\mathsf{doc}}[j] = E_{\mathsf{doc}}[j'] \implies E_{\mathsf{key}}[j] \neq E_{\mathsf{key}}[j']$.

The "count" relation $\mathcal{R}_{\mathsf{cnt}}$. The relation $\mathcal{R}_{\mathsf{cnt}}$ then links the committed counts to the first T entries of E_{key}:

– **Statement:** $x_{\mathsf{cnt}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$.

– **Witness:** $w_{\mathsf{cnt}} := (E_{\mathsf{key}}, E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{cnt}}})$.

– **Relation:** We define $\mathcal{R}_{\mathsf{cnt}}(x_{\mathsf{cnt}}, w_{\mathsf{cnt}}) = 1$ if:

1. *Valid commitments:* $(E_{\mathsf{key}}, \mathsf{op}_{E_{\mathsf{key}}})$ and $(E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{cnt}}})$ are valid openings of $\mathsf{Com}_{E_{\mathsf{key}}}$ and $\mathsf{Com}_{E_{\mathsf{cnt}}}$, respectively.
2. *Consistent counts:* For every $k \in \mathcal{K}$, $E_{\mathsf{cnt}}[k] = \big|\{j \in [T] : E_{\mathsf{key}}[j] = k\}\big|$.

Query-dependent auxiliary commitments. The previous relations concern only the committed document set. To certify a response $\mathcal{J}$ to a query $\mathcal{Q}$, we introduce the following query-dependent commitments. For each $k \in \mathcal{K}$, let $\mathcal{ID}^k_{\mathcal{D}} := \{\mathsf{id} \in \mathcal{ID} \mid \exists j \in [T] : E_{\mathsf{key}}[j] = k \wedge E_{\mathsf{doc}}[j] = \mathsf{id}\}$ denote the set of document identifiers associated with k. Write $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$ in the canonical order inherited from $\mathcal{K} = [m]$. The prover constructs the query-specific count vector $E^{\mathcal{Q}}_{\mathsf{cnt}} := (E_{\mathsf{cnt}}[q_1], \ldots, E_{\mathsf{cnt}}[q_\ell])$, which records the number of documents associated with each queried keyword.

We fix a public capacity $h \geq n+1$ for each query-specific identifier vector. Since at most n documents can contain any keyword, this capacity accommodates every identifier set while ensuring at least one padding position.[9] Because h is fixed independently of $\mathcal{D}$ and $\mathcal{Q}$, the vector lengths do not reveal the number of documents associated with each keyword. For each $i \in [\ell]$, let $c_i := E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$ and define $E^{q_i}_{\mathsf{Enc}} := (\mathsf{id}_{i,1}, \ldots, \mathsf{id}_{i,c_i}, 0, \ldots, 0) \in \mathbb{F}^h$, where the first c_i entries are intended to contain the distinct identifiers associated with q_i. Let $\mathcal{E}^{q_i}_{\mathsf{doc}} := \{E^{q_i}_{\mathsf{Enc}}[j] \mid j \in [h] \wedge E^{q_i}_{\mathsf{Enc}}[j] \neq 0\}$ denote the set represented by these entries. The prover commits to $E^{\mathcal{Q}}_{\mathsf{cnt}}$ and each $E^{q_i}_{\mathsf{Enc}}$ and sets $\mathsf{Com}_{\mathsf{aux}} := \big(\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}\}_{i \in [\ell]}\big)$.

Query-dependent relations. The following three relations use these auxiliary commitments to certify the query-specific counts, the corresponding identifier sets, and their intersection.

The "subset" relation $\mathcal{R}_{\mathsf{sub}}$. Using these auxiliary commitments, $\mathcal{R}_{\mathsf{sub}}$ checks that $E^{\mathcal{Q}}_{\mathsf{cnt}}$ contains the entries of E_{cnt} corresponding to the queried keywords.

– **Statement:** $x_{\mathsf{sub}} := (\mathsf{Com}_{E_{\mathsf{cnt}}}, \mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \mathcal{Q})$.

– **Witness:** $w_{\mathsf{sub}} := (E_{\mathsf{cnt}}, E^{\mathcal{Q}}_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{cnt}}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}})$.

– **Relation:** We define $\mathcal{R}_{\mathsf{sub}}(x_{\mathsf{sub}}, w_{\mathsf{sub}}) = 1$ if:

1. *Valid commitments:* $(E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{cnt}}})$ and $(E^{\mathcal{Q}}_{\mathsf{cnt}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}})$ are valid openings of $\mathsf{Com}_{E_{\mathsf{cnt}}}$ and $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$, respectively.
2. *Consistent query counts:* For every $i \in [\ell]$, $E^{\mathcal{Q}}_{\mathsf{cnt}}[i] = E_{\mathsf{cnt}}[q_i]$.

[9] An implementation may use a smaller h when its supported document sets satisfy $|\mathcal{ID}^k_{\mathcal{D}}| < h$ for every $k \in [m]$.

The "membership" relation $\mathcal{R}_{\mathsf{mem}}$. The relation $\mathcal{R}_{\mathsf{mem}}$ certifies that the identifiers represented by each $E^{q_i}_{\mathsf{Enc}}$ are associated with q_i in the committed document-set representation and that the represented set has cardinality $E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$.

– **Statement:** $x_{\mathsf{mem}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T, \mathcal{Q}, \mathsf{Com}_{\mathsf{aux}})$.

– **Witness:** $w_{\mathsf{mem}} := \big(E_{\mathsf{key}}, E_{\mathsf{doc}}, E^{\mathcal{Q}}_{\mathsf{cnt}}, \{E^{q_i}_{\mathsf{Enc}}\}_{i\in[\ell]}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{op}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]}\big)$.

– **Relation:** We define $\mathcal{R}_{\mathsf{mem}}(x_{\mathsf{mem}}, w_{\mathsf{mem}}) = 1$ if:

1. *Valid commitments:* $(E_{\mathsf{key}}, \mathsf{op}_{E_{\mathsf{key}}})$, $(E_{\mathsf{doc}}, \mathsf{op}_{E_{\mathsf{doc}}})$, and $(E^{\mathcal{Q}}_{\mathsf{cnt}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}})$ are valid openings of $\mathsf{Com}_{E_{\mathsf{key}}}$, $\mathsf{Com}_{E_{\mathsf{doc}}}$, and $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$, respectively. For every $i \in [\ell]$, $(E^{q_i}_{\mathsf{Enc}}, \mathsf{op}_{E^{q_i}_{\mathsf{Enc}}})$ is a valid opening of $\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}$.
2. *Valid nonzero entries and padding:* For every $i \in [\ell]$, letting $c_i := E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$, $c_i \in \{0, \ldots, h-1\}$, $E^{q_i}_{\mathsf{Enc}}[j] \in \mathcal{ID}$ for every $j \in [c_i]$, and $E^{q_i}_{\mathsf{Enc}}[j] = 0$ for every $j \in \{c_i + 1, \ldots, h\}$.
3. *Distinctness:* For every $i \in [\ell]$ and distinct $j, j' \in [c_i]$, $E^{q_i}_{\mathsf{Enc}}[j] \neq E^{q_i}_{\mathsf{Enc}}[j']$.
4. *Membership:* For every $i \in [\ell]$ and $j \in [c_i]$, there exists $k \in [T]$ such that $E_{\mathsf{key}}[k] = q_i$ and $E_{\mathsf{doc}}[k] = E^{q_i}_{\mathsf{Enc}}[j]$.

Together with $\mathcal{R}_{\mathsf{wf}}$, $\mathcal{R}_{\mathsf{cnt}}$, and $\mathcal{R}_{\mathsf{sub}}$, these conditions imply $\mathcal{E}^{q_i}_{\mathsf{doc}} = \mathcal{ID}^{q_i}_{\mathcal{D}}$ for every $i \in [\ell]$.

The "intersection" relation $\mathcal{R}_{\cap}$. Finally, $\mathcal{R}_{\cap}$ checks that $\mathcal{J}$ is exactly the intersection of these query-specific identifier sets.

– **Statement:** $x_{\cap} := (\mathcal{J}, \{\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$.

– **Witness:** $w_{\cap} := (\{E^{q_i}_{\mathsf{Enc}}\}_{i\in[\ell]}, \{\mathsf{op}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$.

– **Relation:** We define $\mathcal{R}_{\cap}(x_{\cap}, w_{\cap}) = 1$ if:

1. *Valid commitments:* For every $i \in [\ell]$, $(E^{q_i}_{\mathsf{Enc}}, \mathsf{op}_{E^{q_i}_{\mathsf{Enc}}})$ is a valid opening of $\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}$.
2. *Correct intersection:* $\mathcal{J} = \bigcap_{i=1}^{\ell} \mathcal{E}^{q_i}_{\mathsf{doc}}$.

Combining the five relations yields

$$\mathcal{J} = \bigcap_{i=1}^{\ell} \mathcal{E}^{q_i}_{\mathsf{doc}} = \bigcap_{i=1}^{\ell} \mathcal{ID}^{q_i}_{\mathcal{D}} = \mathsf{Res}_{\mathcal{D}}(\mathcal{Q}).$$

6.2.2 Online Query Phase Protocol

Fix public parameters $N, m, n, c, h, d, q_{\mathsf{max}} \in \mathbb{N}$, where N is the capacity of the keyword–identifier representation, $\mathcal{K} = [m]$, $\mathcal{ID} = [n]$, c bounds the query size, q_{max} bounds the number of responses under one commitment, h is the capacity of each query-specific identifier vector, and d bounds the length of every committed vector. Let Π_{wf}, Π_{cnt}, Π_{sub}, Π_{mem}, and $\Pi_{\cap}$ be zero-knowledge arguments of knowledge for $\mathcal{R}_{\mathsf{wf}}$, $\mathcal{R}_{\mathsf{cnt}}$, $\mathcal{R}_{\mathsf{sub}}$, $\mathcal{R}_{\mathsf{mem}}$, and $\mathcal{R}_{\cap}$, respectively, as in Definition 2. Let VC be a vector commitment scheme as defined in Definition 1. In Figure 3, we show how to compose these arguments and the vector commitment scheme to design an online query phase protocol for our Sunshine System.

Protocol Π_{online}: Online Query Phase

$\underline{\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda).}$
Compute $\mathsf{pp}_{\mathsf{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$, $\mathsf{pp}_\cap \leftarrow \Pi_\cap.\mathsf{Setup}(1^\lambda)$,

$$\mathsf{pp}_{\mathsf{wf}} \leftarrow \Pi_{\mathsf{wf}}.\mathsf{Setup}(1^\lambda), \quad \mathsf{pp}_{\mathsf{cnt}} \leftarrow \Pi_{\mathsf{cnt}}.\mathsf{Setup}(1^\lambda),$$
$$\mathsf{pp}_{\mathsf{sub}} \leftarrow \Pi_{\mathsf{sub}}.\mathsf{Setup}(1^\lambda), \quad \mathsf{pp}_{\mathsf{mem}} \leftarrow \Pi_{\mathsf{mem}}.\mathsf{Setup}(1^\lambda).$$

Output $\mathsf{pp} := (N, m, n, c, h, d, q_{\max}, \mathsf{pp}_{\mathsf{VC}}, \mathsf{pp}_{\mathsf{wf}}, \mathsf{pp}_{\mathsf{cnt}}, \mathsf{pp}_{\mathsf{sub}}, \mathsf{pp}_{\mathsf{mem}}, \mathsf{pp}_\cap)$.

$\underline{(\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}).}$
Construct $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T)$ from $\mathcal{D}$ and compute

$$(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{key}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{key}}),$$
$$(\mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{doc}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{doc}}),$$
$$(\mathsf{Com}_{E_{\mathsf{cnt}}}, \mathsf{op}_{E_{\mathsf{cnt}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{cnt}}).$$

Output $\mathsf{com} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and $\mathsf{op} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{cnt}}})$.

$\underline{(\mathcal{J}, \pi_{\mathsf{online}}) \leftarrow \mathsf{Response}(\mathsf{pp}, \mathsf{com}, \mathsf{op}, \mathcal{D}, \mathcal{Q}).}$
Write $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$ in canonical order. Compute $\mathcal{J} := \mathsf{Res}_{\mathcal{D}}(\mathcal{Q})$ and $E^{\mathcal{Q}}_{\mathsf{cnt}} := (E_{\mathsf{cnt}}[q_1], \ldots, E_{\mathsf{cnt}}[q_\ell])$. For each $i \in [\ell]$, construct $E^{q_i}_{\mathsf{Enc}} \in \mathbb{F}^h$ whose nonzero entries are exactly the elements of $\mathcal{ID}^{q_i}_{\mathcal{D}}$ (see Section 6.2.1).
Compute $(\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E^{\mathcal{Q}}_{\mathsf{cnt}})$ and, $\forall i \in [\ell]$, compute $(\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}, \mathsf{op}_{E^{q_i}_{\mathsf{Enc}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E^{q_i}_{\mathsf{Enc}})$. Set $\mathsf{Com}_{\mathsf{aux}} := (\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}\}_{i \in [\ell]})$, and compute

$$\pi_{\mathsf{wf}} \leftarrow \Pi_{\mathsf{wf}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{wf}}, x_{\mathsf{wf}}, w_{\mathsf{wf}}),$$
$$\pi_{\mathsf{cnt}} \leftarrow \Pi_{\mathsf{cnt}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{cnt}}, x_{\mathsf{cnt}}, w_{\mathsf{cnt}}),$$
$$\pi_{\mathsf{sub}} \leftarrow \Pi_{\mathsf{sub}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{sub}}, x_{\mathsf{sub}}, w_{\mathsf{sub}}),$$
$$\pi_{\mathsf{mem}} \leftarrow \Pi_{\mathsf{mem}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{mem}}, x_{\mathsf{mem}}, w_{\mathsf{mem}}),$$
$$\pi_\cap \leftarrow \Pi_\cap.\mathsf{Prove}(\mathsf{pp}_\cap, x_\cap, w_\cap).$$

Output $\pi_{\mathsf{online}} := (\mathsf{Com}_{\mathsf{aux}}, \pi_{\mathsf{wf}}, \pi_{\mathsf{cnt}}, \pi_{\mathsf{sub}}, \pi_{\mathsf{mem}}, \pi_\cap)$ together with $\mathcal{J}$.

$\underline{b \leftarrow \mathsf{Verify}(\mathsf{pp}, \mathsf{com}, \mathcal{Q}, \mathcal{J}, \pi_{\mathsf{online}}).}$
Parse $\mathsf{com} = (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$. Reject unless $1 \le |\mathcal{Q}| \le c$, $\mathcal{Q} \subseteq [m]$, $\mathcal{J} \subseteq [n]$, $0 \le T < N$.
Parse $\pi_{\mathsf{online}} = (\mathsf{Com}_{\mathsf{aux}}, \pi_{\mathsf{wf}}, \pi_{\mathsf{cnt}}, \pi_{\mathsf{sub}}, \pi_{\mathsf{mem}}, \pi_\cap)$ and form $x_{\mathsf{wf}}, x_{\mathsf{cnt}}, x_{\mathsf{sub}}, x_{\mathsf{mem}}, x_\cap$.
Compute

$$b_{\mathsf{wf}} \leftarrow \Pi_{\mathsf{wf}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{wf}}, x_{\mathsf{wf}}, \pi_{\mathsf{wf}}),$$
$$b_{\mathsf{cnt}} \leftarrow \Pi_{\mathsf{cnt}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{cnt}}, x_{\mathsf{cnt}}, \pi_{\mathsf{cnt}}),$$
$$b_{\mathsf{sub}} \leftarrow \Pi_{\mathsf{sub}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{sub}}, x_{\mathsf{sub}}, \pi_{\mathsf{sub}}),$$
$$b_{\mathsf{mem}} \leftarrow \Pi_{\mathsf{mem}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{mem}}, x_{\mathsf{mem}}, \pi_{\mathsf{mem}}),$$
$$b_\cap \leftarrow \Pi_\cap.\mathsf{Verify}(\mathsf{pp}_\cap, x_\cap, \pi_\cap).$$

Output $b := b_{\mathsf{wf}} \wedge b_{\mathsf{cnt}} \wedge b_{\mathsf{sub}} \wedge b_{\mathsf{mem}} \wedge b_\cap$.

Figure 3: Online query phase of the Sunshine System.

Theorem 1 (Online Query Phase)**.** *Assume that* VC *satisfies Definition 1 and has commitments of size* $\mathsf{poly}(\lambda)$*. Assume that* Π_{wf}*,* Π_{cnt}*,* Π_{sub}*,* Π_{mem}*, and* $\Pi_\cap$ *satisfy Definition 2 for* $\mathcal{R}_{\mathsf{wf}}$*,* $\mathcal{R}_{\mathsf{cnt}}$*,* $\mathcal{R}_{\mathsf{sub}}$*,*

$\mathcal{R}_{\mathsf{mem}}$, *and* $\mathcal{R}_{\cap}$, *respectively.* [10] [11] *Then the algorithms* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Response}, \mathsf{Verify})$ *described in Figure 3 constitute a protocol* Π_{online} *for the Online Query Phase, satisfying completeness, knowledge soundness, commitment hiding, and zero knowledge as defined in Definition 5. Moreover,* Π_{online} *also satisfies succinctness as defined in Definition 5.*[12]

Proof. We prove that the protocol in Figure 3 satisfies every property required by Definition 5.

Completeness. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$. Let $\mathcal{D}$ be a well-formed document set and let $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$ be a valid query. Suppose $(\mathsf{com}, \mathsf{op})$ and $(\mathcal{J}, \pi_{\mathsf{online}})$ are generated by the honest Commit and $\mathsf{Response}$ algorithms, respectively. By construction, $\mathcal{J} = \mathsf{Res}_{\mathcal{D}}(\mathcal{Q})$, all public checks pass, and the statements and witnesses formed by $\mathsf{Response}$ satisfy $\mathcal{R}_{\mathsf{wf}}$, $\mathcal{R}_{\mathsf{cnt}}$, $\mathcal{R}_{\mathsf{sub}}$, $\mathcal{R}_{\mathsf{mem}}$, and $\mathcal{R}_{\cap}$. Completeness of the five arguments therefore implies $\mathsf{Verify}(\mathsf{pp}, \mathsf{com}, \mathcal{Q}, \mathcal{J}, \pi_{\mathsf{online}}) = 1$.

Knowledge soundness. Fix an n.u. stateful PPT adversary $\mathsf{Adv} = (\mathsf{Adv}_1, \mathsf{Adv}_2)$ and a valid query $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and run $\mathsf{com}^\star \leftarrow \mathsf{Adv}_1(\mathsf{pp})$ and $(\mathcal{J}^\star, \pi^\star_{\mathsf{online}}) \leftarrow \mathsf{Adv}_2(\mathsf{pp}, \mathcal{Q})$. Let Acc denote the event $\mathsf{Verify}(\mathsf{pp}, \mathsf{com}^\star, \mathcal{Q}, \mathcal{J}^\star, \pi^\star_{\mathsf{online}}) = 1$. When Acc occurs, write $\mathsf{com}^\star = (\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{doc}}}, \mathsf{Com}^\star_{E_{\mathsf{cnt}}}, T^\star)$ and $\pi^\star_{\mathsf{online}} = (\mathsf{Com}^\star_{\mathsf{aux}}, \pi^\star_{\mathsf{wf}}, \pi^\star_{\mathsf{cnt}}, \pi^\star_{\mathsf{sub}}, \pi^\star_{\mathsf{mem}}, \pi^\star_{\cap})$, where $\mathsf{Com}^\star_{\mathsf{aux}} = (\mathsf{Com}^\star_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{Com}^\star_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$. We define

$$
\begin{aligned}
x^\star_{\mathsf{wf}} &:= (\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{doc}}}, T^\star),\\
x^\star_{\mathsf{cnt}} &:= (\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{cnt}}}, T^\star),\\
x^\star_{\mathsf{sub}} &:= (\mathsf{Com}^\star_{E_{\mathsf{cnt}}}, \mathsf{Com}^\star_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \mathcal{Q}),\\
x^\star_{\mathsf{mem}} &:= (\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{doc}}}, T^\star, \mathcal{Q}, \mathsf{Com}^\star_{\mathsf{aux}}),\\
x^\star_{\cap} &:= (\mathcal{J}^\star, \{\mathsf{Com}^\star_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]}).
\end{aligned}
$$

For each $\rho \in \{\mathsf{wf}, \mathsf{cnt}, \mathsf{sub}, \mathsf{mem}, \cap\}$, by knowledge soundness of Π_ρ assumed in Theorem 1, there exists a PPT extractor Ext_ρ. Given an accepting transcript of Π_ρ, write $w^\star_\rho \leftarrow \mathsf{Ext}_\rho(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho)$. The witness $w^\star_\rho$ for $\mathcal{R}_\rho$ contains all vectors and corresponding commitment openings required by that relation.

The composed extractor. Using these extractors, we define $\mathsf{Ext}_{\mathsf{online}}$. If Acc does not occur, $\mathsf{Ext}_{\mathsf{online}}$ outputs $\perp$ and terminates. Otherwise, $\mathsf{Ext}_{\mathsf{online}}$ proceeds as follows.

1. For every ρ, $\mathsf{Ext}_{\mathsf{online}}$ invokes Ext_ρ to obtain
$$w^\star_\rho \leftarrow \mathsf{Ext}_\rho(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho).$$
2. For every ρ, $\mathsf{Ext}_{\mathsf{online}}$ checks whether $(x^\star_\rho, w^\star_\rho) \in \mathcal{R}_\rho$. Let $\mathsf{Bad}_{\mathsf{ext}}$ denote the event that Acc occurs and $(x^\star_\rho, w^\star_\rho) \notin \mathcal{R}_\rho$ for some ρ. If $\mathsf{Bad}_{\mathsf{ext}}$ occurs, $\mathsf{Ext}_{\mathsf{online}}$ outputs $\perp$ and terminates.

[10] For zero knowledge, we use the independent setup calls in Figure 3 and each argument's bounded adaptive guarantee for q_{max} proof queries.

[11] For knowledge soundness, it suffices that the five arguments are *jointly* knowledge sound: for every n.u. stateful PPT adversary, there exist PPT extractors Ext_ρ such that, whenever all five sub-verifiers in Figure 3 accept, $(x_\rho, \mathsf{Ext}_\rho(\mathsf{pp}_\rho, x_\rho, \pi_\rho)) \in \mathcal{R}_\rho$ for every ρ, except with negligible probability. This is the only form of knowledge soundness used in the proof, and it is the form provided by the concrete instantiations of Section A.2, in which Π_{cnt}, Π_{mem}, and $\Pi_{\cap}$ rely on the checks performed by Π_{wf} and Π_{mem} (see Lemmas 3, 5 and 6).

[12] For this claim, we additionally require the proof size of $\Pi_{\cap}$ be $\mathsf{poly}(\lambda, |\mathcal{Q}|)$, independently of $|\mathcal{J}|$.

3. $\mathsf{Ext}_{\mathsf{online}}$ compares the vector values assigned to every commitment shared by different relations. In particular, $\mathsf{Ext}_{\mathsf{online}}$ checks consistency of

$$\begin{aligned} &E_{\mathsf{key}} && \text{across } \mathcal{R}_{\mathsf{wf}}, \mathcal{R}_{\mathsf{cnt}}, \mathcal{R}_{\mathsf{mem}}, \\ &E_{\mathsf{doc}} && \text{across } \mathcal{R}_{\mathsf{wf}}, \mathcal{R}_{\mathsf{mem}}, \\ &E_{\mathsf{cnt}} && \text{across } \mathcal{R}_{\mathsf{cnt}}, \mathcal{R}_{\mathsf{sub}}, \\ &E^{\mathcal{Q}}_{\mathsf{cnt}} && \text{across } \mathcal{R}_{\mathsf{sub}}, \mathcal{R}_{\mathsf{mem}}, \\ &E^{q_i}_{\mathsf{Enc}} && \text{across } \mathcal{R}_{\mathsf{mem}}, \mathcal{R}_{\cap} \quad (i \in [\ell]). \end{aligned}$$

   Let $\mathsf{Bad}_{\mathsf{bind}}$ denote the event that $\mathsf{Acc} \wedge \neg\mathsf{Bad}_{\mathsf{ext}}$ occurs and two extracted witnesses assign different vectors to the same shared commitment. If $\mathsf{Bad}_{\mathsf{bind}}$ occurs, $\mathsf{Ext}_{\mathsf{online}}$ outputs $\perp$ and terminates.

4. Denote the consistent vectors by $E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, E^{\mathcal{Q}}_{\mathsf{cnt}}$ and $\{E^{q_i}_{\mathsf{Enc}}\}_{i\in[\ell]}$. For each identifier appearing in the first $T^\star$ positions of E_{doc}, define $\mathcal{W}_{\mathsf{id}} := \{E_{\mathsf{key}}[j] \mid j \in [T^\star] \wedge E_{\mathsf{doc}}[j] = \mathsf{id}\}$ and set $\mathcal{D}^\star := \{(\mathsf{id}, \mathcal{W}_{\mathsf{id}}) \mid \exists j \in [T^\star] : E_{\mathsf{doc}}[j] = \mathsf{id}\}$. Take $\mathsf{op}^\star_{E_{\mathsf{key}}}$ and $\mathsf{op}^\star_{E_{\mathsf{doc}}}$ from $w^\star_{\mathsf{wf}}$, and $\mathsf{op}^\star_{E_{\mathsf{cnt}}}$ from $w^\star_{\mathsf{cnt}}$. Assemble $\mathsf{op}^\star := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathsf{op}^\star_{E_{\mathsf{key}}}, \mathsf{op}^\star_{E_{\mathsf{doc}}}, \mathsf{op}^\star_{E_{\mathsf{cnt}}})$ and output $(\mathcal{D}^\star, \mathsf{op}^\star)$.

We claim that $\mathsf{Ext}_{\mathsf{online}}$ satisfies the knowledge-soundness requirement of Definition 5. We first show that both bad events $\mathsf{Bad}_{\mathsf{ext}}$ and $\mathsf{Bad}_{\mathsf{bind}}$ occur with negligible probability. We then prove that, when Acc occurs and neither $\mathsf{Bad}_{\mathsf{ext}}$ nor $\mathsf{Bad}_{\mathsf{bind}}$ occurs, the output $(\mathcal{D}^\star, \mathsf{op}^\star)$ of $\mathsf{Ext}_{\mathsf{online}}$ satisfies the conditions specified in the knowledge soundness property in Definition 5.

Probability that event $\mathsf{Bad}_{\mathsf{ext}}$ *happens.* Fix ρ. Acc implies $\Pi_\rho.\mathsf{Verify}(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho) = 1$. By knowledge soundness of Π_ρ, $\Pr[\mathsf{Acc} \wedge (x^\star_\rho, w^\star_\rho) \notin \mathcal{R}_\rho] \le \mathsf{negl}_\rho(\lambda)$. Since there are only five sub-arguments,

$$\Pr[\mathsf{Bad}_{\mathsf{ext}}] \le \sum_{\rho} \mathsf{negl}_\rho(\lambda) = \mathsf{negl}(\lambda).$$

Probability that event $\mathsf{Bad}_{\mathsf{bind}}$ *happens.* Suppose $\mathsf{Bad}_{\mathsf{bind}}$ occurs. Then two extracted witnesses assign different vectors $V_0 \neq V_1$ to the same commitment $\mathsf{Com}^\star$. Since $\neg\mathsf{Bad}_{\mathsf{ext}}$ holds, these witnesses also provide valid openings op_0 and op_1 of $\mathsf{Com}^\star$ to V_0 and V_1, respectively. Choose a coordinate j such that $V_0[j] \neq V_1[j]$, and compute $(U_b, \pi_b) \leftarrow \mathsf{VC.Open}(\mathsf{pp}_{\mathsf{VC}}, \mathsf{Com}^\star, \{j\}, \mathsf{op}_b)$ for $b \in \{0,1\}$. By completeness of VC, $\mathsf{VC.Verify}(\mathsf{pp}_{\mathsf{VC}}, \mathsf{Com}^\star, \{j\}, U_b, \pi_b) = 1$ for $b \in \{0,1\}$, while $U_0 \neq U_1$. A non-negligible probability of $\mathsf{Bad}_{\mathsf{bind}}$ would therefore contradict subvector binding of VC. Hence,

$$\Pr[\mathsf{Bad}_{\mathsf{bind}}] \le \mathsf{negl}(\lambda).$$

We define $\mathsf{Good} := \neg\mathsf{Bad}_{\mathsf{ext}} \wedge \neg\mathsf{Bad}_{\mathsf{bind}}$. By the bounds above, $\Pr[\mathsf{Good}] \ge 1 - \mathsf{negl}(\lambda)$. Under Acc, $\mathsf{Ext}_{\mathsf{online}}$ outputs $\perp$ only if $\mathsf{Bad}_{\mathsf{ext}}$ or $\mathsf{Bad}_{\mathsf{bind}}$ occurs. Therefore,

$$\Pr[\mathsf{Acc} \wedge (\mathsf{Ext}_{\mathsf{online}} \text{ outputs } \perp)] \le \mathsf{negl}(\lambda).$$

Correctness of the extracted output. Let Fail denote the event that $\mathsf{Ext}_{\mathsf{online}}$ outputs $(\mathcal{D}^\star, \mathsf{op}^\star)$ for which

$$[\mathcal{D}^\star \text{ is not well formed}] \vee [\mathsf{op}^\star \text{ is not a valid opening of } \mathsf{com}^\star \text{ to } \mathcal{D}^\star] \vee [\mathcal{J}^\star \neq \mathsf{Res}_{\mathcal{D}^\star}(\mathcal{Q})].$$

Suppose, for the sake of contradiction, that $\Pr[\mathsf{Acc} \wedge \mathsf{Fail}]$ is non-negligible. Under $\mathsf{Acc} \wedge \mathsf{Fail}$, both checks of $\mathsf{Ext}_{\mathsf{online}}$ have passed, so Good holds. Thus every extracted witness satisfies its relation,

and the witnesses agree on the vectors assigned to shared commitments. We consider the three possible cases for Fail.

Case 1: $\mathcal{D}^\star$ is not well formed. The extracted witness $w^\star_{\mathsf{wf}}$ satisfies $\mathcal{R}_{\mathsf{wf}}$. Hence, for every $j \in [T^\star]$, $E_{\mathsf{key}}[j] \in \mathcal{K}$ and $E_{\mathsf{doc}}[j] \in \mathcal{ID}$, every position after $T^\star$ contains $(0,0)$, and no keyword–identifier pair occurs twice among the first $T^\star$ positions. By construction, $\mathcal{D}^\star$ contains one document for each distinct identifier appearing in the first $T^\star$ entries of E_{doc}, and every such document has a nonempty keyword set contained in $\mathcal{K}$. Therefore $\mathcal{D}^\star$ is well-formed, a contradiction.

Case 2: $\mathsf{op}^\star$ is not a valid opening of $\mathsf{com}^\star$ to $\mathcal{D}^\star$. $w^\star_{\mathsf{wf}}$ provides valid openings of $\mathsf{Com}^\star_{E_{\mathsf{key}}}$ and $\mathsf{Com}^\star_{E_{\mathsf{doc}}}$, while $w^\star_{\mathsf{cnt}}$ provides a valid opening of $\mathsf{Com}^\star_{E_{\mathsf{cnt}}}$. Consistency of the shared commitment $\mathsf{Com}^\star_{E_{\mathsf{key}}}$ ensures that the keyword vector used by $\mathcal{R}_{\mathsf{cnt}}$ is the same vector used to construct $\mathcal{D}^\star$. Since $w^\star_{\mathsf{cnt}}$ satisfies $\mathcal{R}_{\mathsf{cnt}}$, $E_{\mathsf{cnt}}[k] = |\{j \in [T^\star] : E_{\mathsf{key}}[j] = k\}|$ for every $k \in \mathcal{K}$. Thus $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T^\star)$ is a valid committed representation of $\mathcal{D}^\star$, and $\mathsf{op}^\star$ is a valid opening of $\mathsf{com}^\star$ to this document set, a contradiction.

Case 3: $\mathcal{J}^\star \neq \mathsf{Res}_{\mathcal{D}^\star}(\mathcal{Q})$. For every $i \in [\ell]$, define $\mathcal{ID}^{q_i}_{\mathcal{D}^\star} = \{E_{\mathsf{doc}}[j] \mid j \in [T^\star] \wedge E_{\mathsf{key}}[j] = q_i\}$. Let $c_i := E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$ and let $\mathcal{E}^{q_i}_{\mathsf{doc}} := \{E^{q_i}_{\mathsf{Enc}}[j] \mid j \in [c_i]\}$ be the identifier set represented by the corresponding query-specific vector. Since $\mathcal{R}_{\mathsf{wf}}$ forbids repeated keyword–identifier pairs, the positions containing q_i contain distinct document identifiers. The relations $\mathcal{R}_{\mathsf{cnt}}$ and $\mathcal{R}_{\mathsf{sub}}$ give $|\mathcal{ID}^{q_i}_{\mathcal{D}^\star}| = E_{\mathsf{cnt}}[q_i] = E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$. The relation $\mathcal{R}_{\mathsf{mem}}$ gives $\mathcal{E}^{q_i}_{\mathsf{doc}} \subseteq \mathcal{ID}^{q_i}_{\mathcal{D}^\star}$ and $|\mathcal{E}^{q_i}_{\mathsf{doc}}| = E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$. Thus, $\mathcal{E}^{q_i}_{\mathsf{doc}} = \mathcal{ID}^{q_i}_{\mathcal{D}^\star}$ for every $i \in [\ell]$. Finally, $\mathcal{R}_\cap$ gives

$$\mathcal{J}^\star = \bigcap_{i=1}^{\ell} \mathcal{E}^{q_i}_{\mathsf{doc}} = \bigcap_{i=1}^{\ell} \mathcal{ID}^{q_i}_{\mathcal{D}^\star} = \mathsf{Res}_{\mathcal{D}^\star}(\mathcal{Q}),$$

which contradicts this case.

None of the three cases can occur under $\mathsf{Acc} \wedge \mathsf{Good}$. This contradicts the assumption that $\Pr[\mathsf{Acc} \wedge \mathsf{Fail}]$ is non-negligible. Therefore,

$$\Pr[\mathsf{Acc} \wedge \mathsf{Fail}] \leq \mathsf{negl}(\lambda).$$

Together with the previous bound on the probability that Acc occurs and $\mathsf{Ext}_{\mathsf{online}}$ outputs $\perp$, this establishes the knowledge-soundness requirement of Definition 5.

Commitment hiding. Let $\mathcal{D}_0$ and $\mathcal{D}_1$ be two distinct well-formed document sets with the same value T. Write

$$\mathsf{com}_b = (\mathsf{Com}_{E_{\mathsf{key}},b}, \mathsf{Com}_{E_{\mathsf{doc}},b}, \mathsf{Com}_{E_{\mathsf{cnt}},b}, T), \quad b \in \{0,1\}.$$

Starting from com_0, consider three hybrids that replace $\mathsf{Com}_{E_{\mathsf{key}},0}$, $\mathsf{Com}_{E_{\mathsf{doc}},0}$, and $\mathsf{Com}_{E_{\mathsf{cnt}},0}$ by their counterparts in com_1, one at a time. Each replacement is computationally indistinguishable by hiding of VC. Hence $\{(\mathsf{pp}, \mathsf{com}_0)\} \approx_c \{(\mathsf{pp}, \mathsf{com}_1)\}$. This establishes commitment hiding.

Zero knowledge. For every $\rho \in \{\mathsf{wf}, \mathsf{cnt}, \mathsf{sub}, \mathsf{mem}, \cap\}$, let $(\mathsf{Sim}_{\rho,1}, \mathsf{Sim}_{\rho,2})$ denote the setup and proof simulators for the bounded adaptive zero-knowledge guarantee of Π_ρ used in Theorem 1. The setup simulator produces $(\widetilde{\mathsf{pp}}_\rho, \tau_\rho)$, and the proof simulator uses these outputs and a statement to generate a proof without its witness. We construct the simulators $\mathsf{Sim}_{\mathsf{online},1}$ and $\mathsf{Sim}_{\mathsf{online},2}$ required by Definition 5. We consider at most q_{max} adaptive queries under one fixed commitment, with all five subproofs included in each response.

Setup simulation. On input 1^λ, $\mathsf{Sim}_{\mathsf{online},1}$ samples $\widetilde{\mathsf{pp}}_{\mathsf{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. For every ρ, it independently invokes the setup simulator $(\widetilde{\mathsf{pp}}_\rho, \tau_\rho) \leftarrow \mathsf{Sim}_{\rho,1}(1^\lambda)$. It assembles $\widetilde{\mathsf{pp}}$ in the same format as pp, collects every τ_ρ in τ_{online}, and outputs $(\widetilde{\mathsf{pp}}, \tau_{\mathsf{online}})$.

Proof simulation. For each oracle query $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$, $\mathsf{Sim}_{\mathsf{online},2}$ receives $(\widetilde{\mathsf{pp}}, \tau_{\mathsf{online}}, \widetilde{\mathsf{com}}, \mathcal{Q}, \mathcal{J})$ and proceeds as follows.

1. Parse $\widetilde{\mathsf{com}} = (\widetilde{\mathsf{Com}}_{E_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{doc}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{cnt}}}, T)$.

2. Generate the shared auxiliary commitments with fresh independent randomness by computing $(\widetilde{\mathsf{Com}}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \cdot) \leftarrow \mathsf{VC.Commit}(\widetilde{\mathsf{pp}}_{\mathsf{VC}}, 0^\ell)$ and, for every $i \in [\ell]$, $(\widetilde{\mathsf{Com}}_{E^{q_i}_{\mathsf{Enc}}}, \cdot) \leftarrow \mathsf{VC.Commit}(\widetilde{\mathsf{pp}}_{\mathsf{VC}}, 0^h)$. Define $\widetilde{\mathsf{Com}}_{\mathsf{aux}} := (\widetilde{\mathsf{Com}}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\widetilde{\mathsf{Com}}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$.

3. Form the five sub-statements

$$\begin{aligned}
\widetilde{x}_{\mathsf{wf}} &:= (\widetilde{\mathsf{Com}}_{E_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{doc}}}, T),\\
\widetilde{x}_{\mathsf{cnt}} &:= (\widetilde{\mathsf{Com}}_{E_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{cnt}}}, T),\\
\widetilde{x}_{\mathsf{sub}} &:= (\widetilde{\mathsf{Com}}_{E_{\mathsf{cnt}}}, \widetilde{\mathsf{Com}}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \mathcal{Q}),\\
\widetilde{x}_{\mathsf{mem}} &:= (\widetilde{\mathsf{Com}}_{E_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{doc}}}, T, \mathcal{Q}, \widetilde{\mathsf{Com}}_{\mathsf{aux}}),\\
\widetilde{x}_{\cap} &:= (\mathcal{J}, \{\widetilde{\mathsf{Com}}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]}).
\end{aligned}$$

   Invoke the five proof simulators in protocol order:

$$\begin{aligned}
\widetilde{\pi_{\mathsf{wf}}} &\leftarrow \mathsf{Sim}_{\mathsf{wf},2}(\widetilde{\mathsf{pp}}_{\mathsf{wf}}, \tau_{\mathsf{wf}}, \widetilde{x}_{\mathsf{wf}}),\\
\widetilde{\pi_{\mathsf{cnt}}} &\leftarrow \mathsf{Sim}_{\mathsf{cnt},2}(\widetilde{\mathsf{pp}}_{\mathsf{cnt}}, \tau_{\mathsf{cnt}}, \widetilde{x}_{\mathsf{cnt}}),\\
\widetilde{\pi_{\mathsf{sub}}} &\leftarrow \mathsf{Sim}_{\mathsf{sub},2}(\widetilde{\mathsf{pp}}_{\mathsf{sub}}, \tau_{\mathsf{sub}}, \widetilde{x}_{\mathsf{sub}}),\\
\widetilde{\pi_{\mathsf{mem}}} &\leftarrow \mathsf{Sim}_{\mathsf{mem},2}(\widetilde{\mathsf{pp}}_{\mathsf{mem}}, \tau_{\mathsf{mem}}, \widetilde{x}_{\mathsf{mem}}),\\
\widetilde{\pi_{\cap}} &\leftarrow \mathsf{Sim}_{\cap,2}(\widetilde{\mathsf{pp}}_{\cap}, \tau_{\cap}, \widetilde{x}_{\cap}).
\end{aligned}$$

4. Output

$$\widetilde{\pi_{\mathsf{online}}} := (\widetilde{\mathsf{Com}}_{\mathsf{aux}}, \widetilde{\pi_{\mathsf{wf}}}, \widetilde{\pi_{\mathsf{cnt}}}, \widetilde{\pi_{\mathsf{sub}}}, \widetilde{\pi_{\mathsf{mem}}}, \widetilde{\pi_{\cap}}).$$

Setup is performed only once. Each response uses fresh randomness for its auxiliary commitments, while the component simulators retain their internal state across responses.

Hybrid distributions. Fix a well-formed document set $\mathcal{D}$ and an n.u. PPT adversary making at most q_{max} valid queries. Each hybrid describes the adversary's complete view of the execution. Queries are chosen adaptively within each hybrid, and each query $\mathcal{Q}$ receives the response $\mathcal{J} := \mathsf{Res}_{\mathcal{D}}(\mathcal{Q})$.

- H_0 is the real Online transcript.
- H_1 uses $\mathsf{Sim}_{\mathsf{online},1}$ for setup. The public commitment is generated honestly once. For each query, the auxiliary commitments are generated honestly, while all five subproofs are generated by their corresponding proof simulators.
- H_2 is obtained from H_1 by replacing the commitment to $E^{\mathcal{Q}}_{\mathsf{cnt}}$ in every response with a commitment to 0^ℓ. All simulated proofs are regenerated for the new sub-statements.
- Set $\mathsf{H}_{3,0} := \mathsf{H}_2$. For every $i \in [c]$, $\mathsf{H}_{3,i}$ is obtained from $\mathsf{H}_{3,i-1}$ by replacing the i-th query-specific identifier commitment in each response with a commitment to 0^h. Queries with fewer than i keywords are unchanged. All simulated proofs are regenerated for the new sub-statements.

Indistinguishability.

$\mathsf{H}_0 \approx_c \mathsf{H}_1$**.** For every $k \in \{0, \ldots, 5\}$, let G_k replace the setup and proof of the first k arguments, in the order $\Pi_{\mathsf{wf}}, \Pi_{\mathsf{cnt}}, \Pi_{\mathsf{sub}}, \Pi_{\mathsf{mem}}, \Pi_{\cap}$, by their setup and proof simulators. The other arguments use their honest algorithms. The VC parameters and all commitments are generated honestly in every hybrid. Thus $\mathsf{G}_0 = \mathsf{H}_0$ and $\mathsf{G}_5 = \mathsf{H}_1$.

Fix $k \in [5]$, and let ρ denote the argument replaced in this transition. In the bounded adaptive zero-knowledge experiment for Π_ρ, run the setup simulators for the previous arguments and the honest setup algorithms for the following arguments, and generate the VC parameters honestly. For each Online query, honest commitment generation provides a valid (x_ρ, w_ρ), which is submitted to the proof oracle. Place the returned proof in the position of π_ρ and generate the other subproofs according to the hybrid. Continue answering subsequent adaptive queries in the same manner.

There are at most q_{max} calls to the oracle. Since these setup calls use independent randomness, the real and simulated experiments for Π_ρ give G_{k-1} and G_k, respectively. Bounded adaptive zero knowledge of Π_ρ therefore gives $\mathsf{G}_{k-1} \approx_c \mathsf{G}_k$. By transitivity,

$$\mathsf{H}_0 = \mathsf{G}_0 \approx_c \mathsf{G}_1 \approx_c \cdots \approx_c \mathsf{G}_5 = \mathsf{H}_1.$$

$\mathsf{H}_1 \approx_c \mathsf{H}_2$**.** Replace $E^{\mathcal{Q}}_{\mathsf{cnt}}$ one response at a time. At the selected response, a reduction to hiding of VC submits $(E^{\mathcal{Q}}_{\mathsf{cnt}}, 0^\ell)$ as its equal-length challenge vectors, with an empty opening index set. It uses the returned commitment to form the sub-statements and generates their proofs with the component simulators. It then continues answering the adversary's adaptive queries according to the hybrid. The component simulation parameters are generated independently of the VC setup, and the reduction does not need an opening of the challenge commitment. The two challenge choices therefore give the adjacent hybrids. Each replacement is computationally indistinguishable by hiding of VC. There are at most q_{max} replacements, giving $\mathsf{H}_1 \approx_c \mathsf{H}_2$.

$\mathsf{H}_{3,i-1} \approx_c \mathsf{H}_{3,i}$ **for every** $i \in [c]$**.** Fix $i \in [c]$. Replace the i-th query-specific identifier commitment one response at a time, skipping queries with fewer than i keywords. The hiding reduction uses $(E^{q_i}_{\mathsf{Enc}}, 0^h)$ as its challenge vectors and an empty opening index set. It forms the sub-statements using the returned commitment, generates the subproofs with their simulators, and continues the adaptive execution. No opening of the challenge commitment is needed. Each replacement is computationally indistinguishable by hiding of VC. There are at most q_{max} replacements for each i, so

$$\mathsf{H}_2 = \mathsf{H}_{3,0} \approx_c \mathsf{H}_{3,1} \approx_c \cdots \approx_c \mathsf{H}_{3,c}.$$

In $\mathsf{H}_{3,c}$, every query-dependent commitment is to a zero vector of the prescribed public length, and every subproof is simulated. This is exactly the simulated experiment in Definition 5. Combining the preceding transitions,

$$\mathsf{H}_0 \approx_c \mathsf{H}_1 \approx_c \mathsf{H}_2 = \mathsf{H}_{3,0} \approx_c \cdots \approx_c \mathsf{H}_{3,c}.$$

Since q_{max} and c are polynomially bounded, this establishes zero knowledge.

Succinctness. By Definition 2, each Π_ρ has proof size $\mathsf{poly}(\lambda, |x_\rho|)$ and verification time $\mathsf{poly}(\lambda, |x_\rho|, \log |w_\rho|)$. For the fixed public parameters, $|x_\rho| = \mathsf{poly}(\lambda, |\mathcal{Q}|)$ for $\rho \neq \cap$. Together with the assumed bound on $|\pi_\cap|$ and the commitment-size bound for VC, this implies $|\pi_{\mathsf{online}}| = \mathsf{poly}(\lambda, |\mathcal{Q}|)$. The verifier reads and checks its statement and invokes the five sub-verifiers, giving total runtime

$$\mathsf{poly}(\lambda, |\mathcal{Q}|, |\mathcal{J}|) + \sum_\rho \mathsf{poly}(\lambda, |x_\rho|, \log |w_\rho|) = \mathsf{poly}(\lambda, |\mathcal{Q}|, |\mathcal{J}|, \log |\mathcal{D}|).$$

Thus Π_{online} satisfies succinctness as defined in Definition 5. □

We instantiate VC with the hiding KZG-based vector commitment scheme of Section A.1, and instantiate Π_{wf}, Π_{cnt}, Π_{sub}, Π_{mem}, and $\Pi_{\cap}$ with the arguments of knowledge described in Section A.2. Completeness and knowledge soundness follow from the component guarantees and the corresponding arguments in the proof of Theorem 1. Zero knowledge of the common-SRS instantiation is established jointly in Section A.2, under the masking bounds of Section A.1. Together with commitment hiding and the cost analysis in Section A.3, we obtain the following corollary.

Corollary 1 (Sunshine Systems: Online Query Phase)**.** *There exists a protocol for the Online Query Phase of a Sunshine System satisfying Definition 5 in the SRS model and programmable ROM, with knowledge soundness against algebraic adversaries (see Section 4.2).*

Remark 4 (Standalone and audit-assisted online proofs)**.** *The response proof in Figure 3 gives a standalone realization of the online query phase required by Definition 5. In a full deployment, however, we expect the online query phase to be used together with the audit mechanism of Definition 6.*

In this audit-assisted setting, the Audit Phase has already established $\mathcal{R}_{\mathsf{wf}}$ and $\mathcal{R}_{\mathsf{cnt}}$ for the current commitment com. Hence, an online response need only include $\pi_{\mathsf{query}} := (\mathsf{Com}_{\mathsf{aux}}, \pi_{\mathsf{sub}}, \pi_{\mathsf{mem}}, \pi_{\cap})$. The verifier first checks that the query uses the current com recorded by the auditor and then verifies π_{sub}, π_{mem}, and $\pi_{\cap}$, avoiding repeated proofs of $\mathcal{R}_{\mathsf{wf}}$ and $\mathcal{R}_{\mathsf{cnt}}$. Section 7 formalizes how the Audit Phase establishes and preserves these reusable guarantees. For a fixed com satisfying these conditions, the query-proof simulation uses only the simulators for Π_{sub}, Π_{mem}, and $\Pi_{\cap}$.

Prover runtime. With the bounded-query masks of Section A.2, generating π_{wf} and π_{cnt} requires quasilinear work in N and $m + cq_{\mathsf{max}}$. After lookup preprocessing, the Base-only audit-assisted prover is quasilinear in $m + cq_{\mathsf{max}}$ and ℓh. A Cache of size δ adds lookup work depending on ℓ, h, δ. For fixed $m, c, q_{\mathsf{max}}, \ell, h, \delta$, the per-query bound is independent of N. Detailed bounds are given in Section A.3. In contrast, directly proving the equivalent query over the full pair table requires processing the database for each query, as in our PoneglyphDB [40] baseline. Table 3 gives the concrete comparison.

7 Instantiating the Audit Phase

We now present an efficient realization of the audit phase (see Definition 6) using the same vector-commitment and argument interfaces as in Section 6.

7.1 Overview

As discussed in Remark 4, the well-formedness and count-consistency relations certify properties that depend only on the public commitment to a document set. Audits establish these guarantees once per commitment and enforce that updates are append only, allowing for reuse across queries.

The agency first publishes an initial commitment com, which an auditor (e.g., a government oversight office or civil society) records and verifies for well-formedness and correct keyword counts. Query issuers either perform these checks themselves or obtain com from an auditor. Audit-assisted queries use the latest accepted commitment.

For updates, the agency publishes a new commitment, followed immediately or later by a zero-knowledge proof that the appended documents are well-formed, their identifiers are fresh, all existing keyword–identifier pairs are preserved, exactly the new pairs are appended, and keyword counts are updated correctly. The auditor records the new commitment and verifies the proof

against it. Until verification succeeds, audit-assisted queries continue using the previously accepted commitment and its document set.

Auxiliary objects. Let $\mathcal{D}$ be a document set, let Δ be a document set to be appended to $\mathcal{D}$, and define $\mathcal{D}^+ := \mathcal{D} \cup \Delta$. We reuse the document representation from Section 6.1. Let $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T)$ denote the representation of $\mathcal{D}$, and let $(E^+_{\mathsf{key}}, E^+_{\mathsf{doc}}, E^+_{\mathsf{cnt}}, T^+)$ denote the representation of $\mathcal{D}^+$. The corresponding public commitments are

$$\mathsf{com} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T),$$
$$\mathsf{com}^+ := (\mathsf{Com}_{E^+_{\mathsf{key}}}, \mathsf{Com}_{E^+_{\mathsf{doc}}}, \mathsf{Com}_{E^+_{\mathsf{cnt}}}, T^+).$$

For any document set $\mathcal{X}$, define its document-identifier set as $\mathcal{ID}_{\mathcal{X}} := \{\mathsf{id} \mid (\mathsf{id}, \mathcal{W}) \in \mathcal{X}\}$. In particular, we write $\mathcal{ID}_{\mathcal{D}}$, $\mathcal{ID}_{\Delta}$, and $\mathcal{ID}_{\mathcal{D}^+}$ for the identifier sets of $\mathcal{D}$, Δ, and $\mathcal{D}^+$, respectively. As part of the audit protocol, the agency also commits to vector encodings of $\mathcal{ID}_{\mathcal{D}}$ and $\mathcal{ID}_{\mathcal{D}^+}$; we denote these commitments by $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$ and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$.

Audit relations. The Audit proof first certifies the current public commitment and then its append-only transition.

Relation $\mathcal{R}_{\mathsf{state}}$. We use $\mathcal{R}_{\mathsf{state}}$ to link the current public commitment to a well-formed document set, its keyword counts, and its identifier set:

- **Statement:** $x_{\mathsf{state}} = (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}})$.
- **Witness:** $w_{\mathsf{state}} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathcal{ID}_{\mathcal{D}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{cnt}}}, \mathsf{op}_{\mathcal{ID}_{\mathcal{D}}})$.
- **Relation:** We have $(x_{\mathsf{state}}, w_{\mathsf{state}}) \in \mathcal{R}_{\mathsf{state}}$ if:
  1. *Valid commitments:* The commitments $\mathsf{Com}_{E_{\mathsf{key}}}$, $\mathsf{Com}_{E_{\mathsf{doc}}}$, $\mathsf{Com}_{E_{\mathsf{cnt}}}$, and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$ open to E_{key}, E_{doc}, E_{cnt}, and $\mathcal{ID}_{\mathcal{D}}$, respectively, under the corresponding openings in w_{state}.
  2. *Well-formedness.* The statement $(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$ and its corresponding witness satisfy $\mathcal{R}_{\mathsf{wf}}$.
  3. *Count consistency.* The statement $(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and its corresponding witness satisfy $\mathcal{R}_{\mathsf{cnt}}$.
  4. *Identifier consistency.* $\mathcal{ID}_{\mathcal{D}}$ is exactly the set of document identifiers in the first T entries of E_{doc}:
  $$\mathcal{ID}_{\mathcal{D}} = \{E_{\mathsf{doc}}[j] \mid j \in [T]\}.$$

Relation $\mathcal{R}_{\mathsf{upd}}$. The relation $\mathcal{R}_{\mathsf{upd}}$ reuses the commitment to the old identifier set and relates the two public commitments through the appended rows and their identifiers. Let $\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}} := (\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$. We define $\mathcal{R}_{\mathsf{upd}}$ as follows:

- **Statement:** $x_{\mathsf{upd}} := (\mathsf{com}, \mathsf{com}^+, \mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}})$, where $\mathsf{com} = (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and $\mathsf{com}^+ = (\mathsf{Com}_{E^+_{\mathsf{key}}}, \mathsf{Com}_{E^+_{\mathsf{doc}}}, \mathsf{Com}_{E^+_{\mathsf{cnt}}}, T^+)$.
- **Witness:**
$$w_{\mathsf{upd}} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathcal{ID}_{\mathcal{D}}, E^+_{\mathsf{key}}, E^+_{\mathsf{doc}}, E^+_{\mathsf{cnt}}, \mathcal{ID}_{\mathcal{D}^+},$$
$$\mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{cnt}}}, \mathsf{op}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{op}_{E^+_{\mathsf{key}}}, \mathsf{op}_{E^+_{\mathsf{doc}}}, \mathsf{op}_{E^+_{\mathsf{cnt}}}, \mathsf{op}_{\mathcal{ID}_{\mathcal{D}^+}}).$$

- **Relation:** Let $B := T^+ - T$ and require $0 \le T \le T^+ < N$. For every $j \in [B]$, define $E^{\Delta}_{\mathsf{key}}[j] := E^{+}_{\mathsf{key}}[T+j]$ and $E^{\Delta}_{\mathsf{doc}}[j] := E^{+}_{\mathsf{doc}}[T+j]$. Thus, $E^{\Delta}_{\mathsf{key}}, E^{\Delta}_{\mathsf{doc}} \in \mathbb{F}^B$ contain the appended keyword–identifier pairs. Define $E^{\Delta}_{\mathsf{cnt}}[k] := |\{j \in [B] : E^{\Delta}_{\mathsf{key}}[j] = k\}|$ for every $k \in [m]$, and $\mathcal{ID}_{\Delta} := \{E^{\Delta}_{\mathsf{doc}}[j] \mid j \in [B]\}$. These objects are determined by the witness and the public values T, T^+.

  We have $(x_{\mathsf{upd}}, w_{\mathsf{upd}}) \in \mathcal{R}_{\mathsf{upd}}$ if:

  1. *Valid commitments.* The commitments in com, com^+, $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$, and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ open to their corresponding witness values under the openings in w_{upd}.
  2. *Well-formed appended rows.* For every $j \in [B]$, $E^{\Delta}_{\mathsf{key}}[j] \in [m]$ and $E^{\Delta}_{\mathsf{doc}}[j] \in [n]$, and no keyword–identifier pair appears twice among the appended rows.
  3. *Fresh identifiers.* The identifier set of the updated document set is the disjoint union $\mathcal{ID}_{\mathcal{D}^+} = \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$.
  4. *Append-only consistency.* The representation opened from com^+ preserves the first T rows opened from com, appends exactly the rows induced by Δ, and is zero thereafter:

$$(E^{+}_{\mathsf{key}}[j], E^{+}_{\mathsf{doc}}[j]) = \begin{cases} (E_{\mathsf{key}}[j], E_{\mathsf{doc}}[j]), & 1 \le j \le T, \\ (E^{\Delta}_{\mathsf{key}}[j-T], E^{\Delta}_{\mathsf{doc}}[j-T]), & T < j \le T^+, \\ (0,0), & T^+ < j \le N. \end{cases}$$

  5. *Count consistency.* The updated count vector satisfies $E^{+}_{\mathsf{cnt}} = E_{\mathsf{cnt}} + E^{\Delta}_{\mathsf{cnt}}$.

Concrete zero-knowledge arguments for these relations are given in Section B.1. The following lemma shows that the conditions of $\mathcal{R}_{\mathsf{state}}$ are preserved by an append-only update. Consequently, no separate proof of $\mathcal{R}_{\mathsf{state}}$ is required for the updated commitment.

Lemma 1 (Preservation of the public commitment conditions)**.** *Let*

$$\begin{aligned} x_{\mathsf{state}} &:= (\mathsf{com}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}), \\ x^{+}_{\mathsf{state}} &:= (\mathsf{com}^+, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}), \\ x_{\mathsf{upd}} &:= (\mathsf{com}, \mathsf{com}^+, \mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}}), \end{aligned}$$

where $\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}} := (\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$. *Suppose* $(x_{\mathsf{state}}, w_{\mathsf{state}}) \in \mathcal{R}_{\mathsf{state}}$ *and* $(x_{\mathsf{upd}}, w_{\mathsf{upd}}) \in \mathcal{R}_{\mathsf{upd}}$, *and the two witnesses assign the same vector values to every shared commitment. Then the updated vectors and openings in* w_{upd} *form a witness for* $\mathcal{R}_{\mathsf{state}}$ *at* x^{+}_{state}.

Proof. Let w^{+}_{state} consist of $E^{+}_{\mathsf{key}}, E^{+}_{\mathsf{doc}}, E^{+}_{\mathsf{cnt}}, \mathcal{ID}_{\mathcal{D}^+}$ and their corresponding commitment openings, as contained in w_{upd}. We verify the remaining three conditions of $\mathcal{R}_{\mathsf{state}}$.

- *Well-formedness.* First, since x_{state} satisfies $\mathcal{R}_{\mathsf{state}}$, the first T rows opened from com satisfy $\mathcal{R}_{\mathsf{wf}}$. By the well-formedness and append-only conditions of $\mathcal{R}_{\mathsf{upd}}$, the next $B := T^+ - T$ rows are valid keyword–identifier pairs with no repeated pair, and every row after T^+ is zero. Moreover, $\mathcal{ID}_{\mathcal{D}^+} = \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$ implies that no identifier appearing in the appended rows appears among the first T rows. Hence no keyword–identifier pair is repeated across the two parts. Therefore, $(\mathsf{Com}_{E^{+}_{\mathsf{key}}}, \mathsf{Com}_{E^{+}_{\mathsf{doc}}}, T^+)$ and its corresponding witness satisfy $\mathcal{R}_{\mathsf{wf}}$.

- *Count consistency.* Second, for every $k \in [m]$, $\mathcal{R}_{\mathsf{state}}$ for x_{state} and the definition of $E^{\Delta}_{\mathsf{cnt}}$ give

$$E_{\mathsf{cnt}}[k] = \big|\{j \in [T] : E_{\mathsf{key}}[j] = k\}\big|$$

and

$$E^{\Delta}_{\mathsf{cnt}}[k] = \big|\{j \in [B] : E^{\Delta}_{\mathsf{key}}[j] = k\}\big|.$$

Using the append-only and count-consistency conditions of $\mathcal{R}_{\mathsf{upd}}$, we obtain

$$E^{+}_{\mathsf{cnt}}[k] = E_{\mathsf{cnt}}[k] + E^{\Delta}_{\mathsf{cnt}}[k] = \big|\{j \in [T^{+}] : E^{+}_{\mathsf{key}}[j] = k\}\big|.$$

Thus, $(\mathsf{Com}_{E^{+}_{\mathsf{key}}}, \mathsf{Com}_{E^{+}_{\mathsf{cnt}}}, T^{+})$ and its corresponding witness satisfy $\mathcal{R}_{\mathsf{cnt}}$.

- *Identifier consistency.* Finally, append-only consistency gives

$$\{E^{+}_{\mathsf{doc}}[j] \mid j \in [T^{+}]\} = \{E_{\mathsf{doc}}[j] \mid j \in [T]\} \cup \{E^{\Delta}_{\mathsf{doc}}[j] \mid j \in [B]\}.$$

Since $\mathcal{ID}_{\mathcal{D}} = \{E_{\mathsf{doc}}[j] \mid j \in [T]\}$ and $\mathcal{ID}_{\Delta} = \{E^{\Delta}_{\mathsf{doc}}[j] \mid j \in [B]\}$, the fresh-identifier condition of $\mathcal{R}_{\mathsf{upd}}$ implies

$$\mathcal{ID}_{\mathcal{D}^{+}} = \{E^{+}_{\mathsf{doc}}[j] \mid j \in [T^{+}]\}.$$

Hence all three conditions of $\mathcal{R}_{\mathsf{state}}$ hold for x^{+}_{state}, and therefore $(x^{+}_{\mathsf{state}}, w^{+}_{\mathsf{state}}) \in \mathcal{R}_{\mathsf{state}}$. □

7.2 Audit Protocol

Fix public integers $N, m, n, N_{\Delta}, h_D, h_{\Delta}, d \in \mathbb{N}$. The parameters N, m, n, d are as in Section 6.2.2; N_{Δ} is the public encoding capacity of the appended keyword–identifier vectors, with $B = T^{+} - T \le N_{\Delta}$ (see Section B.1.3); h_D and h_{Δ} are the public capacities of the identifier encodings associated with a complete document set and an appended document set, respectively. Let Π_{state} and Π_{upd} be zero-knowledge arguments of knowledge for $\mathcal{R}_{\mathsf{state}}$ and $\mathcal{R}_{\mathsf{upd}}$, respectively, satisfying the completeness, knowledge-soundness, and zero-knowledge requirements of Definition 2. Let VC be a vector commitment scheme as defined in Definition 1. In Figure 4, we show how to compose these arguments and the vector commitment scheme to design an audit phase protocol for our Sunshine System. The setup procedures in Figures 3 and 4 show the parameters used by each phase. In a complete system, one global setup samples $\mathsf{pp}_{\mathsf{VC}}$ once, includes all listed parameters in pp, and shares $\mathsf{pp}_{\mathsf{VC}}$ across both phases.

Theorem 2 (Audit Phase)**.** *Assume that* VC *satisfies Definition 1 and that* Π_{state} *and* Π_{upd} *satisfy the completeness, knowledge-soundness, and zero-knowledge requirements of Definition 2 for* $\mathcal{R}_{\mathsf{state}}$ *and* $\mathcal{R}_{\mathsf{upd}}$*, respectively.* [13] *Then the algorithms* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Update}, \mathsf{Audit})$ *described in Figure 4 constitute a protocol* Π_{audit} *for the Audit Phase satisfying completeness, knowledge soundness, and zero knowledge as defined in Definition 6.*

Proof. We prove that the protocol in Figure 4 satisfies every property required by Definition 6.

Completeness. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^{\lambda})$. Let $\mathcal{D}$ and Δ satisfy the conditions in Definition 6. Suppose $(\mathsf{com}, \mathsf{op})$ and $(\mathsf{com}^{+}, \mathsf{op}^{+})$ are generated honestly for $\mathcal{D}$ and $\mathcal{D} \cup \Delta$, respectively, and π_{audit} is generated by the honest Update algorithm. By Definition 6, the updated pair vectors preserve

[13] As in Theorem 1, it suffices that Π_{state} and Π_{upd} are *jointly* knowledge sound, i.e., that extraction of witnesses for $\mathcal{R}_{\mathsf{state}}$ and $\mathcal{R}_{\mathsf{upd}}$ succeeds whenever both sub-verifiers in Figure 4 accept. This is the guarantee provided by the concrete instantiation (Lemma 9), in which Π_{upd} relies on the accepted Π_{state} proof for the current state.

Protocol Π_{audit}: Audit Phase

$\underline{\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda).}$
Compute $\mathsf{pp}_{\mathsf{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$, $\mathsf{pp}_{\mathsf{state}} \leftarrow \Pi_{\mathsf{state}}.\mathsf{Setup}(1^\lambda)$, and $\mathsf{pp}_{\mathsf{upd}} \leftarrow \Pi_{\mathsf{upd}}.\mathsf{Setup}(1^\lambda)$.
Output $\mathsf{pp} := (N, m, n, N_\Delta, h_D, h_\Delta, d, \mathsf{pp}_{\mathsf{VC}}, \mathsf{pp}_{\mathsf{state}}, \mathsf{pp}_{\mathsf{upd}})$.

$\underline{(\mathsf{com}, \mathsf{op}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathcal{D}).}$
Construct $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T)$ from $\mathcal{D}$ and compute

$$(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{key}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{key}}),$$
$$(\mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{doc}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{doc}}),$$
$$(\mathsf{Com}_{E_{\mathsf{cnt}}}, \mathsf{op}_{E_{\mathsf{cnt}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, E_{\mathsf{cnt}}).$$

Output $\mathsf{com} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and $\mathsf{op} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{cnt}}})$.

$\underline{\pi_{\mathsf{audit}} \leftarrow \mathsf{Update}(\mathsf{pp}, \mathcal{D}, \mathsf{com}, \mathsf{op}, \Delta, \mathsf{com}^+, \mathsf{op}^+).}$
Set $\mathcal{D}^+ := \mathcal{D} \cup \Delta$. Parse

$$\mathsf{com} = (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$$

and

$$\mathsf{com}^+ = (\mathsf{Com}_{E^+_{\mathsf{key}}}, \mathsf{Com}_{E^+_{\mathsf{doc}}}, \mathsf{Com}_{E^+_{\mathsf{cnt}}}, T^+),$$

together with their openings. Then, compute

$$\mathcal{ID}_{\mathcal{D}} := \{\mathsf{id} \mid (\mathsf{id}, \mathcal{W}) \in \mathcal{D}\}, \qquad \mathcal{ID}_{\mathcal{D}^+} := \{\mathsf{id} \mid (\mathsf{id}, \mathcal{W}) \in \mathcal{D}^+\}.$$

Encode $\mathcal{ID}_{\mathcal{D}}$ and $\mathcal{ID}_{\mathcal{D}^+}$ as vectors I and I^+, respectively, and compute

$$(\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{op}_{\mathcal{ID}_{\mathcal{D}}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, I),$$
$$(\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}, \mathsf{op}_{\mathcal{ID}_{\mathcal{D}^+}}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}_{\mathsf{VC}}, I^+).$$

Set $\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}} := (\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$. Then, compute

$$\pi_{\mathsf{state}} \leftarrow \Pi_{\mathsf{state}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{state}}, x_{\mathsf{state}}, w_{\mathsf{state}}),$$
$$\pi_{\mathsf{upd}} \leftarrow \Pi_{\mathsf{upd}}.\mathsf{Prove}(\mathsf{pp}_{\mathsf{upd}}, x_{\mathsf{upd}}, w_{\mathsf{upd}}).$$

Output $\pi_{\mathsf{audit}} := (\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}}, \pi_{\mathsf{state}}, \pi_{\mathsf{upd}})$.

$\underline{b \leftarrow \mathsf{Audit}(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}}).}$
Parse $\pi_{\mathsf{audit}} = (\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}}, \pi_{\mathsf{state}}, \pi_{\mathsf{upd}})$, where $\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}} = (\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$.
Reject unless $0 \le T \le T^+ < N$, $T^+ - T \le N_\Delta$, and every public object has the shape prescribed by N, m, N_Δ, h_D, and h_Δ.
Form x_{state} and x_{upd}, and compute

$$b_{\mathsf{state}} \leftarrow \Pi_{\mathsf{state}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{state}}, x_{\mathsf{state}}, \pi_{\mathsf{state}}),$$
$$b_{\mathsf{upd}} \leftarrow \Pi_{\mathsf{upd}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{upd}}, x_{\mathsf{upd}}, \pi_{\mathsf{upd}}).$$

Output $b := b_{\mathsf{state}} \wedge b_{\mathsf{upd}}$.

Figure 4: Sunshine System: Audit Phase.

the first T entries and append the pairs induced by Δ. The statements and witnesses formed by the honest Update algorithm therefore satisfy $\mathcal{R}_{\mathsf{state}}$ and $\mathcal{R}_{\mathsf{upd}}$. Completeness of Π_{state} and Π_{upd} therefore implies $\mathsf{Audit}(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}}) = 1$.

Knowledge soundness. Fix an n.u. stateful PPT adversary Adv. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and run $(\mathsf{com}^\star, \mathsf{com}^{+\star}, \pi^\star_{\mathsf{audit}}) \leftarrow \mathsf{Adv}(\mathsf{pp})$. Let Acc denote the event $\mathsf{Audit}(\mathsf{pp}, \mathsf{com}^\star, \mathsf{com}^{+\star}, \pi^\star_{\mathsf{audit}}) = 1$. When Acc occurs, write

$$\begin{aligned}\mathsf{com}^\star &:= (\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{doc}}}, \mathsf{Com}^\star_{E_{\mathsf{cnt}}}, T^\star),\\ \mathsf{com}^{+\star} &:= (\mathsf{Com}^\star_{E^+_{\mathsf{key}}}, \mathsf{Com}^\star_{E^+_{\mathsf{doc}}}, \mathsf{Com}^\star_{E^+_{\mathsf{cnt}}}, T^{+\star}),\\ \pi^\star_{\mathsf{audit}} &:= (\mathsf{Com}^{\mathsf{audit},\star}_{\mathsf{aux}}, \pi^\star_{\mathsf{state}}, \pi^\star_{\mathsf{upd}}),\end{aligned}$$

where $\mathsf{Com}^{\mathsf{audit},\star}_{\mathsf{aux}} := (\mathsf{Com}^\star_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}^\star_{\mathcal{ID}_{\mathcal{D}^+}})$. We define

$$\begin{aligned}x^\star_{\mathsf{state}} &:= (\mathsf{com}^\star, \mathsf{Com}^\star_{\mathcal{ID}_{\mathcal{D}}}),\\ x^\star_{\mathsf{upd}} &:= (\mathsf{com}^\star, \mathsf{com}^{+\star}, \mathsf{Com}^{\mathsf{audit},\star}_{\mathsf{aux}}).\end{aligned}$$

For each $\rho \in \{\mathsf{state}, \mathsf{upd}\}$, by knowledge soundness of Π_ρ assumed in Theorem 2, there exists a PPT extractor Ext_ρ. Given an accepting transcript of Π_ρ, write $w^\star_\rho \leftarrow \mathsf{Ext}_\rho(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho)$. The witness $w^\star_\rho$ for $\mathcal{R}_\rho$ contains all vectors and corresponding commitment openings required by that relation.

The composed extractor. Using these extractors, we define $\mathsf{Ext}_{\mathsf{audit}}$. If Acc does not occur, $\mathsf{Ext}_{\mathsf{audit}}$ outputs $\perp$ and terminates. Otherwise, $\mathsf{Ext}_{\mathsf{audit}}$ proceeds as follows.

1. For every ρ, $\mathsf{Ext}_{\mathsf{audit}}$ invokes Ext_ρ to obtain

$$w^\star_\rho \leftarrow \mathsf{Ext}_\rho(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho).$$

2. For every ρ, $\mathsf{Ext}_{\mathsf{audit}}$ checks whether $(x^\star_\rho, w^\star_\rho) \in \mathcal{R}_\rho$. Let $\mathsf{Bad}_{\mathsf{ext}}$ denote the event that Acc occurs and $(x^\star_\rho, w^\star_\rho) \notin \mathcal{R}_\rho$ for some ρ. If $\mathsf{Bad}_{\mathsf{ext}}$ occurs, $\mathsf{Ext}_{\mathsf{audit}}$ outputs $\perp$ and terminates.

3. $\mathsf{Ext}_{\mathsf{audit}}$ compares the vector values assigned to every commitment shared by $\mathcal{R}_{\mathsf{state}}$ and $\mathcal{R}_{\mathsf{upd}}$. In particular, $\mathsf{Ext}_{\mathsf{audit}}$ checks consistency at $\mathsf{Com}^\star_{E_{\mathsf{key}}}, \mathsf{Com}^\star_{E_{\mathsf{doc}}}, \mathsf{Com}^\star_{E_{\mathsf{cnt}}}$, and $\mathsf{Com}^\star_{\mathcal{ID}_{\mathcal{D}}}$. Let $\mathsf{Bad}_{\mathsf{bind}}$ denote the event that $\mathsf{Acc} \wedge \neg\mathsf{Bad}_{\mathsf{ext}}$ occurs and two extracted witnesses assign different vectors to the same shared commitment. If $\mathsf{Bad}_{\mathsf{bind}}$ occurs, $\mathsf{Ext}_{\mathsf{audit}}$ outputs $\perp$ and terminates.

4. Denote the consistent vectors by $E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, E^+_{\mathsf{key}}, E^+_{\mathsf{doc}}, E^+_{\mathsf{cnt}}$, and the encoded identifier sets by $\mathcal{ID}_{\mathcal{D}}, \mathcal{ID}_{\mathcal{D}^+}$. For each identifier appearing in the first $T^\star$ positions of E_{doc}, define $\mathcal{W}_{\mathsf{id}} := \{E_{\mathsf{key}}[j] \mid j \in [T^\star] \wedge E_{\mathsf{doc}}[j] = \mathsf{id}\}$, and set $\mathcal{D} := \{(\mathsf{id}, \mathcal{W}_{\mathsf{id}}) \mid \exists j \in [T^\star] : E_{\mathsf{doc}}[j] = \mathsf{id}\}$. Let $B^\star := T^{+\star} - T^\star$ and, for every $j \in [B^\star]$, define $E^\Delta_{\mathsf{key}}[j] := E^+_{\mathsf{key}}[T^\star + j]$ and $E^\Delta_{\mathsf{doc}}[j] := E^+_{\mathsf{doc}}[T^\star + j]$. Let E^Δ_{cnt} and $\mathcal{ID}_\Delta$ be the count vector and identifier set induced by these appended rows, as specified by $\mathcal{R}_{\mathsf{upd}}$. Define

$$\Delta := \{(\mathsf{id}, \{E^\Delta_{\mathsf{key}}[j] \mid j \in [B^\star] \wedge E^\Delta_{\mathsf{doc}}[j] = \mathsf{id}\}) \mid \mathsf{id} \in \mathcal{ID}_\Delta\}.$$

   Take the openings of the three commitments in $\mathsf{com}^\star$ from $w^\star_{\mathsf{state}}$, and those in $\mathsf{com}^{+\star}$ from $w^\star_{\mathsf{upd}}$. Assemble

$$\begin{aligned}\mathsf{op}^\star &:= (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, \mathsf{op}^\star_{E_{\mathsf{key}}}, \mathsf{op}^\star_{E_{\mathsf{doc}}}, \mathsf{op}^\star_{E_{\mathsf{cnt}}}),\\ \mathsf{op}^{+\star} &:= (E^+_{\mathsf{key}}, E^+_{\mathsf{doc}}, E^+_{\mathsf{cnt}}, \mathsf{op}^\star_{E^+_{\mathsf{key}}}, \mathsf{op}^\star_{E^+_{\mathsf{doc}}}, \mathsf{op}^\star_{E^+_{\mathsf{cnt}}})\end{aligned}$$

   and output $(\mathcal{D}, \Delta, \mathsf{op}^\star, \mathsf{op}^{+\star})$.

We claim that $\mathsf{Ext}_{\mathsf{audit}}$ satisfies the knowledge-soundness requirement of Definition 6. We first show that both bad events $\mathsf{Bad}_{\mathsf{ext}}$ and $\mathsf{Bad}_{\mathsf{bind}}$ occur with negligible probability. We then prove that, when Acc occurs and neither $\mathsf{Bad}_{\mathsf{ext}}$ nor $\mathsf{Bad}_{\mathsf{bind}}$ occurs, the output $(\mathcal{D}, \Delta, \mathsf{op}^\star, \mathsf{op}^{+\star})$ of $\mathsf{Ext}_{\mathsf{audit}}$ satisfies the conditions specified in the knowledge soundness property in Definition 6.

Probability that event $\mathsf{Bad}_{\mathsf{ext}}$ *happens.* Fix ρ. Acc implies $\Pi_\rho.\mathsf{Verify}(\mathsf{pp}_\rho, x^\star_\rho, \pi^\star_\rho) = 1$. By knowledge soundness of Π_ρ, $\Pr[\mathsf{Acc} \wedge (x^\star_\rho, w^\star_\rho) \notin \mathcal{R}_\rho] \le \mathsf{negl}_\rho(\lambda)$. Since there are only two sub-arguments,

$$\Pr[\mathsf{Bad}_{\mathsf{ext}}] \le \sum_\rho \mathsf{negl}_\rho(\lambda) = \mathsf{negl}(\lambda).$$

Probability that event $\mathsf{Bad}_{\mathsf{bind}}$ *happens.* Suppose $\mathsf{Bad}_{\mathsf{bind}}$ occurs. Then two extracted witnesses assign different vectors $V_0 \neq V_1$ to the same commitment $\mathsf{Com}^\star$. Since $\neg\mathsf{Bad}_{\mathsf{ext}}$ holds, these witnesses also provide valid openings op_0 and op_1 of $\mathsf{Com}^\star$ to V_0 and V_1, respectively. Choose a coordinate j such that $V_0[j] \neq V_1[j]$, and compute $(U_b, \pi_b) \leftarrow \mathsf{VC.Open}(\mathsf{pp}_{\mathsf{VC}}, \mathsf{Com}^\star, \{j\}, \mathsf{op}_b)$ for $b \in \{0,1\}$. By completeness of VC, $\mathsf{VC.Verify}(\mathsf{pp}_{\mathsf{VC}}, \mathsf{Com}^\star, \{j\}, U_b, \pi_b) = 1$ for $b \in \{0,1\}$ while $U_0 \neq U_1$. A non-negligible probability of $\mathsf{Bad}_{\mathsf{bind}}$ would therefore contradict subvector binding of VC. Hence,

$$\Pr[\mathsf{Bad}_{\mathsf{bind}}] \le \mathsf{negl}(\lambda).$$

We define $\mathsf{Good} := \neg\mathsf{Bad}_{\mathsf{ext}} \wedge \neg\mathsf{Bad}_{\mathsf{bind}}$. By the bounds above, $\Pr[\mathsf{Good}] \ge 1 - \mathsf{negl}(\lambda)$. Under Acc, $\mathsf{Ext}_{\mathsf{audit}}$ outputs $\perp$ only if $\mathsf{Bad}_{\mathsf{ext}}$ or $\mathsf{Bad}_{\mathsf{bind}}$ occurs. Therefore,

$$\Pr[\mathsf{Acc} \wedge (\mathsf{Ext}_{\mathsf{audit}} \text{ outputs } \perp)] \le \mathsf{negl}(\lambda).$$

Correctness of the extracted output. Let Fail denote the event that $\mathsf{Ext}_{\mathsf{audit}}$ outputs $(\mathcal{D}, \Delta, \mathsf{op}^\star, \mathsf{op}^{+\star})$ for which

$$\begin{aligned}
&[\mathcal{D} \text{ is not well formed}] \vee [\Delta \text{ is not well formed}] \vee [\mathcal{D} \cup \Delta \text{ is not well formed}] \\
&\vee [\exists\, \mathsf{id}, \mathcal{W}, \mathcal{W}' : (\mathsf{id}, \mathcal{W}) \in \mathcal{D} \wedge (\mathsf{id}, \mathcal{W}') \in \Delta] \\
&\vee [\mathsf{op}^\star \text{ is not a valid opening of } \mathsf{com}^\star \text{ to } \mathcal{D}] \\
&\vee [\mathsf{op}^{+\star} \text{ is not a valid opening of } \mathsf{com}^{+\star} \text{ to } \mathcal{D} \cup \Delta].
\end{aligned}$$

Suppose, for the sake of contradiction, that $\Pr[\mathsf{Acc} \wedge \mathsf{Fail}]$ is non-negligible. Under $\mathsf{Acc} \wedge \mathsf{Fail}$, both checks of $\mathsf{Ext}_{\mathsf{audit}}$ have passed, so Good holds. Thus every extracted witness satisfies its relation, and the witnesses agree on the vectors assigned to shared commitments. We consider the three possible cases for Fail.

Case 1: $\mathcal{D}$, Δ, *or* $\mathcal{D} \cup \Delta$ *is not well formed.* The extracted witness $w^\star_{\mathsf{state}}$ satisfies $\mathcal{R}_{\mathsf{state}}$, so the first $T^\star$ positions of $(E_{\mathsf{key}}, E_{\mathsf{doc}})$ contain valid keyword–identifier pairs with no repeated pair. Collecting these rows by identifier produces a well-formed document set $\mathcal{D}$. The well-formedness of appended rows guaranteed by $\mathcal{R}_{\mathsf{upd}}$ similarly establishes that Δ is well-formed.

Identifier consistency of $\mathcal{R}_{\mathsf{state}}$ and freshness in $\mathcal{R}_{\mathsf{upd}}$ give $\mathcal{ID}_{\mathcal{D}} = \{E_{\mathsf{doc}}[j] \mid j \in [T^\star]\}$ and $\mathcal{ID}_{\mathcal{D}^+} = \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$. Thus $\mathcal{D}$ and Δ have disjoint identifier sets, and their union is well-formed as well. All three document sets are therefore well formed, contradicting this case.

Case 2: $\mathcal{D}$ *and* Δ *share an identifier.* Suppose that $(\mathsf{id}, \mathcal{W}) \in \mathcal{D}$ and $(\mathsf{id}, \mathcal{W}') \in \Delta$ for some $\mathsf{id}, \mathcal{W}, \mathcal{W}'$. Identifier consistency of $\mathcal{R}_{\mathsf{state}}$ and the definition of $\mathcal{ID}_{\Delta}$ imply $\mathsf{id} \in \mathcal{ID}_{\mathcal{D}} \cap \mathcal{ID}_{\Delta}$. However, the fresh-identifier condition of $\mathcal{R}_{\mathsf{upd}}$ gives $\mathcal{ID}_{\mathcal{D}^+} = \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$, so $\mathcal{ID}_{\mathcal{D}} \cap \mathcal{ID}_{\Delta} = \emptyset$, a contradiction.

Case 3: One of the two commitment openings is invalid. $w^\star_{\mathsf{state}}$ provides valid openings of all three commitments in $\mathsf{com}^\star$. Its well-formedness condition identifies the pair vectors with the representation used to construct $\mathcal{D}$, and its count-consistency condition gives $E_{\mathsf{cnt}}[k] = |\{j \in [T^\star] : E_{\mathsf{key}}[j] = k\}|$ for every $k \in \mathcal{K}$. Thus $(E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, T^\star)$ is a valid committed representation of $\mathcal{D}$, and $\mathsf{op}^\star$ opens $\mathsf{com}^\star$ to this document set.

For the updated commitment, consistency of the shared commitments ensures that the old vectors used by $\mathcal{R}_{\mathsf{upd}}$ are the same vectors used to construct $\mathcal{D}$. Append-only consistency gives

$$(E^+_{\mathsf{key}}[j], E^+_{\mathsf{doc}}[j]) = \begin{cases} (E_{\mathsf{key}}[j], E_{\mathsf{doc}}[j]), & 1 \le j \le T^\star, \\ (E^\Delta_{\mathsf{key}}[j - T^\star], E^\Delta_{\mathsf{doc}}[j - T^\star]), & T^\star < j \le T^{+\star}, \\ (0,0), & T^{+\star} < j \le N. \end{cases}$$

Moreover, count consistency gives, for every $k \in \mathcal{K}$,

$$\begin{aligned} E^+_{\mathsf{cnt}}[k] &= E_{\mathsf{cnt}}[k] + E^\Delta_{\mathsf{cnt}}[k] \\ &= |\{j \in [T^\star] : E_{\mathsf{key}}[j] = k\}| + |\{j \in [B^\star] : E^\Delta_{\mathsf{key}}[j] = k\}| \\ &= |\{j \in [T^{+\star}] : E^+_{\mathsf{key}}[j] = k\}|. \end{aligned}$$

Hence the new vectors form a valid representation of $\mathcal{D} \cup \Delta$, and $w^\star_{\mathsf{upd}}$ provides valid openings of their commitments. Therefore both $\mathsf{op}^\star$ and $\mathsf{op}^{+\star}$ are valid openings to the required document sets, a contradiction.

None of the three cases can occur under $\mathsf{Acc} \wedge \mathsf{Good}$. This contradicts the assumption that $\Pr[\mathsf{Acc} \wedge \mathsf{Fail}]$ is non-negligible. Therefore,

$$\Pr[\mathsf{Acc} \wedge \mathsf{Fail}] \le \mathsf{negl}(\lambda).$$

Together with the previous bound on the probability that Acc occurs and $\mathsf{Ext}_{\mathsf{audit}}$ outputs $\perp$, this establishes the knowledge-soundness requirement of Definition 6.

Zero knowledge. For every $\rho \in \{\mathsf{state}, \mathsf{upd}\}$, let $(\mathsf{Sim}_{\rho,1}, \mathsf{Sim}_{\rho,2})$ denote the setup and proof simulators guaranteed by Definition 2 for Π_ρ. The setup simulator produces $(\widetilde{\mathsf{pp}}_\rho, \tau_\rho)$, and the proof simulator uses these outputs and a statement to generate a proof without its witness. We construct the simulators $\mathsf{Sim}_{\mathsf{Audit},1}$ and $\mathsf{Sim}_{\mathsf{Audit},2}$ required by Definition 6.

Setup simulation. On input 1^λ, $\mathsf{Sim}_{\mathsf{Audit},1}$ first samples $\widetilde{\mathsf{pp}}_{\mathsf{VC}} \leftarrow \mathsf{VC.Setup}(1^\lambda, d)$. For every ρ, it independently invokes the setup simulator $(\widetilde{\mathsf{pp}}_\rho, \tau_\rho) \leftarrow \mathsf{Sim}_{\rho,1}(1^\lambda)$. It assembles $\widetilde{\mathsf{pp}}$ in the same format as pp, collects every τ_ρ in τ_{Audit}, and outputs $(\widetilde{\mathsf{pp}}, \tau_{\mathsf{Audit}})$.

Proof simulation. On input $(\widetilde{\mathsf{pp}}, \tau_{\mathsf{Audit}}, \widetilde{\mathsf{com}}, \widetilde{\mathsf{com}}^+)$, $\mathsf{Sim}_{\mathsf{Audit},2}$ proceeds as follows.

1. Parse $\widetilde{\mathsf{com}} := (\widetilde{\mathsf{Com}}_{E_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{doc}}}, \widetilde{\mathsf{Com}}_{E_{\mathsf{cnt}}}, T)$ and $\widetilde{\mathsf{com}}^+ := (\widetilde{\mathsf{Com}}_{E^+_{\mathsf{key}}}, \widetilde{\mathsf{Com}}_{E^+_{\mathsf{doc}}}, \widetilde{\mathsf{Com}}_{E^+_{\mathsf{cnt}}}, T^+)$.
2. Using the identifier set encoding of public capacity h_D, generate $\widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}}}$ and $\widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}^+}}$ as commitments to the empty set under $\widetilde{\mathsf{pp}}_{\mathsf{VC}}$ with fresh independent randomness. Define $\widetilde{\mathsf{Com}}^{\mathsf{audit}}_{\mathsf{aux}} := (\widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}}}, \widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}^+}})$.
3. Form the two sub-statements

$$\begin{aligned} \widetilde{x_{\mathsf{state}}} &:= (\widetilde{\mathsf{com}}, \widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}}}), \\ \widetilde{x_{\mathsf{upd}}} &:= (\widetilde{\mathsf{com}}, \widetilde{\mathsf{com}}^+, \widetilde{\mathsf{Com}}^{\mathsf{audit}}_{\mathsf{aux}}). \end{aligned}$$

Invoke the two proof simulators in protocol order:

$$\widetilde{\pi_{\mathsf{state}}} \leftarrow \mathsf{Sim}_{\mathsf{state},2}(\widetilde{\mathsf{pp}_{\mathsf{state}}}, \tau_{\mathsf{state}}, \widetilde{x_{\mathsf{state}}}),$$
$$\widetilde{\pi_{\mathsf{upd}}} \leftarrow \mathsf{Sim}_{\mathsf{upd},2}(\widetilde{\mathsf{pp}_{\mathsf{upd}}}, \tau_{\mathsf{upd}}, \widetilde{x_{\mathsf{upd}}}).$$

4. Output $\widetilde{\pi_{\mathsf{audit}}} := (\widetilde{\mathsf{Com}}^{\mathsf{audit}}_{\mathsf{aux}}, \widetilde{\pi_{\mathsf{state}}}, \widetilde{\pi_{\mathsf{upd}}})$.

Hybrid distributions. Fix $\mathcal{D}$ and Δ satisfying the zero-knowledge premise of Definition 6. Each hybrid is a distribution of the complete view $(\mathsf{pp}, \mathsf{com}, \mathsf{com}^+, \pi_{\mathsf{audit}})$, generated according to the rules below.

- H_0 is the real Audit transcript.
- H_1 uses $\mathsf{Sim}_{\mathsf{Audit},1}$ for setup. Both public commitments and the auxiliary identifier-set commitments are generated honestly for $\mathcal{D}$ and $\mathcal{D} \cup \Delta$, while both subproofs are generated by their corresponding proof simulators.
- H_2 is obtained from H_1 by replacing the commitment to $\mathcal{ID}_{\mathcal{D}}$ with $\widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}}}$. The statements and their simulated proofs for Π_{state} and Π_{upd} are regenerated using the new commitments.
- H_3 is obtained from H_2 by replacing the commitment to $\mathcal{ID}_{\mathcal{D}^+}$ with $\widetilde{\mathsf{Com}}_{\mathcal{ID}_{\mathcal{D}^+}}$. The statement and its simulated proof for Π_{upd} are regenerated using the new commitments.

Indistinguishability.

$\mathsf{H}_0 \approx_c \mathsf{H}_1$**.** For every $k \in \{0, 1, 2\}$, let G_k replace the setup and proof of the first k arguments, in the order $\Pi_{\mathsf{state}}, \Pi_{\mathsf{upd}}$, by their setup and proof simulators. The other arguments use their honest algorithms. The VC parameters and all commitments are generated honestly in every hybrid. Thus $\mathsf{G}_0 = \mathsf{H}_0$ and $\mathsf{G}_2 = \mathsf{H}_1$.

Fix $k \in [2]$, and let ρ denote the argument replaced in this transition. In the zero-knowledge experiment for Π_ρ, run the setup simulators for the previous arguments and the honest setup algorithms for the following arguments, and generate the VC parameters honestly. The honest commitment generation provides a valid (x_ρ, w_ρ), which is submitted to that experiment. Place the returned proof in the position of π_ρ and generate the other subproof according to the hybrid. Since these setup calls use independent randomness, the real and simulated experiments for Π_ρ produce G_{k-1} and G_k, respectively. Zero knowledge of Π_ρ therefore gives $\mathsf{G}_{k-1} \approx_c \mathsf{G}_k$. By transitivity,

$$\mathsf{H}_0 = \mathsf{G}_0 \approx_c \mathsf{G}_1 \approx_c \mathsf{G}_2 = \mathsf{H}_1.$$

$\mathsf{H}_1 \approx_c \mathsf{H}_2$**.** The two hybrids differ only in whether $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$ commits to $\mathcal{ID}_{\mathcal{D}}$ or to the empty set, using the same public capacity. By hiding of VC, these commitments are computationally indistinguishable. Once either commitment is fixed, the corresponding statements for Π_{state} and Π_{upd} are formed, and their proofs are generated by the corresponding simulators as prescribed by $\mathsf{Sim}_{\mathsf{Audit},2}$. The other commitments remain unchanged. Therefore, $\mathsf{H}_1 \approx_c \mathsf{H}_2$.

$\mathsf{H}_2 \approx_c \mathsf{H}_3$**.** The two hybrids differ in whether the new $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ commits to $\mathcal{ID}_{\mathcal{D}^+}$ or to the empty set, using the same public capacity. By hiding of VC, these commitments are computationally indistinguishable. Once either commitment is fixed, the corresponding statement for Π_{upd} is formed, and its proof is generated by $\mathsf{Sim}_{\mathsf{upd},2}$. The other commitments remain unchanged, and all other subproofs follow the same generation procedures. Therefore, $\mathsf{H}_2 \approx_c \mathsf{H}_3$.

In H_3, both identifier-set commitments are to the empty set, and both subproofs are simulated. Its proof component has exactly the output distribution of $\mathsf{Sim}_{\mathsf{Audit},2}$, so the full view is the simulated experiment in Definition 6. Combining the preceding transitions,

$$\mathsf{H}_0 \approx_c \mathsf{H}_1 \approx_c \mathsf{H}_2 \approx_c \mathsf{H}_3.$$

This establishes zero knowledge. □

We instantiate VC with the hiding KZG-based vector commitment scheme of Section A.1, and instantiate Π_{state} and Π_{upd} with the arguments of knowledge described in Section B.1. The component guarantees and the relation-level arguments of Theorem 2, together with the common-SRS simulation in Section B.1, yield the following corollary.

Corollary 2 (Sunshine Systems: Audit Phase)**.** *There exists a protocol for the Audit Phase of a Sunshine System satisfying Definition 6 in the SRS model and programmable ROM, with knowledge soundness against algebraic adversaries (see Section 4.2).*

Remark 5 (Response monotonicity)**.** *Every accepted update preserves all previously committed documents and appends only documents with fresh identifiers. Hence, for every query $\mathcal{Q}$, $\mathsf{Res}_{\mathcal{D}}(\mathcal{Q}) \subseteq \mathsf{Res}_{\mathcal{D}\cup\Delta}(\mathcal{Q})$. This follows directly from the append-only semantics of $\mathcal{R}_{\mathsf{upd}}$.*

Remark 6 (Standalone and subsequent audits)**.** *Figure 4 describes a standalone Audit invocation, with proof $\pi_{\mathsf{audit}} = (\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}}, \pi_{\mathsf{state}}, \pi_{\mathsf{upd}})$. In a sequential execution, the auditor first accepts $x^{(0)}_{\mathsf{state}} = (\mathsf{com}^{(0)}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(0)}}})$ together with $\pi^{(0)}_{\mathsf{state}}$, and records $x^{(0)}_{\mathsf{state}}$ as the current verified Audit record.*

For each subsequent transition from $\mathsf{com}^{(r-1)}$ to $\mathsf{com}^{(r)}$, the commitment $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r-1)}}}$ is reused from the current audit record. The agency publishes a new commitment $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r)}}}$ and a fresh proof $\pi^{(r)}_{\mathsf{upd}}$. The auditor forms

$$\begin{aligned}\mathsf{Com}^{\mathsf{audit},(r)}_{\mathsf{aux}} &:= \left(\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r-1)}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r)}}}\right),\\ x^{(r)}_{\mathsf{upd}} &:= \left(\mathsf{com}^{(r-1)}, \mathsf{com}^{(r)}, \mathsf{Com}^{\mathsf{audit},(r)}_{\mathsf{aux}}\right).\end{aligned}$$

If $\mathsf{com}^{(r)}$ was published before the proof, the auditor also checks that it matches the value recorded. It accepts only if $\mathsf{com}^{(r-1)}$ and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r-1)}}}$ match its current verified Audit record and $\Pi_{\mathsf{upd}}.\mathsf{Verify}(\mathsf{pp}_{\mathsf{upd}}, x^{(r)}_{\mathsf{upd}}, \pi^{(r)}_{\mathsf{upd}}) = 1$. By Lemma 1, $x^{(r)}_{\mathsf{state}} = (\mathsf{com}^{(r)}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^{(r)}}})$ then satisfies $\mathcal{R}_{\mathsf{state}}$ and becomes the new verified Audit record.

Remark 7 (Rebasing)**.** *Our instantiation of Π_{mem} (see Section A.2.5) relies on a lookup argument (see Section 4.3). As discussed in Section 2, state-of-the-art lookup arguments use table-dependent preprocessing to reduce online time.*

We use the Base+Cache framework of Dutta et al. [21] (see Section 4.3) to maintain this preprocessing. The table represented by a previously verified commitment forms the Base, while subsequently appended pairs form the Cache and are incorporated through a sparse lookup correction. As Cache grows, the system may perform a rebase*: the table represented by the latest verified commitment becomes the new Base, the Cache is cleared, and the table-dependent preprocessing is recomputed. Rebasing changes neither the public commitment nor its randomness, so it requires no new Audit proof and does not reset the query limit.*

Prover runtime. Under the masking-degree bounds in Section A.2, generating the initial π_{state} requires quasilinear work in N, $m + cq_{\mathsf{max}}$, and h_D. For a subsequent update with $B = T^+ - T$ appended pairs, generating π_{upd} requires quasilinear work in $m + cq_{\mathsf{max}}$, N_Δ, h_D, and h_Δ. Unlike the per-keyword construction discussed in Section 2, our update proof checks document permanence against one global identifier set, rather than separately against every keyword's identifier set. A standalone Audit incurs the sum of the costs of generating π_{state} and π_{upd}. Rebasing has quasilinear cost in N and mh, where the lookup preprocessing includes m identifier vectors of capacity h. The detailed bounds, including amortized rebasing costs, are given in Section B.2.

8 Implementation and Evaluation

We implement our Sunshine System and present a concrete evaluation of its performance. All the necessary artifacts to replicate our experiments are available at https://anonymous.4open.science/r/sunshine-system-6388/.

8.1 Implementation

We implement our Sunshine System for keyword queries in Rust. The cryptographic backend uses the Arkworks ecosystem [4] over BLS12-381. All reported experiments use release builds and run on a 2024 Mac mini with an Apple M4 processor and 16 GB of RAM, a standard commodity computer available refurbished for $\approx$ \$500. Unless explicitly labeled otherwise, all benchmarks use one thread and we report the average of five runs.

We evaluate our system on the Enron email corpus [44], a standard information-processing dataset, though not government-generated. We select $m = 2^{10}$ keywords from the most frequent non-stopword, non-numeric terms and cap each document at 32 keywords, splitting longer documents as needed.[14] The resulting dataset contains $50,520$ documents and $1,048,570$ keyword–identifier pairs. Queries operate on these records, so splitting may alter conjunction matches when an email's keywords span multiple records. Our default log capacity is $N = 2^{20}$, and we initialize the Base with $40,904$ documents and $838,846$ keyword–identifier pairs, about 80% of the capacity. Each update batch contains at most 256 keyword–identifier pairs, and the default configuration rebases after 64 batches.

Our default configuration uses $h = 2^{13}$, $h_D = 2^{16}$, and $h_\Delta = 2^9$ for the query-specific, full-document, and appended identifier vectors, respectively. It supports $q_{\mathsf{max}} = 2^{10}$ adaptive audit-assisted queries per commitment, with at most $c = 16$ keywords per query[15]. The count mask has degree at most 16,388. The keyword, document, identifier, and selector masks have degree at most 32. The query-count mask has degree at most $2\ell + 1$. The experimental artifacts for $N \le 2^{20}$ contain both G_1 power families through exponent 2^{21} and G_2 powers through $2^{22} + 2$. For $N = 2^{22}$, these ranges extend to 2^{23} and $2^{24} + 2$, respectively. The SRS construction and the masking-degree derivation are given in Sections A.1 and A.2.

8.2 Evaluation Results

We benchmark three phases: (1) the online query phase, in which the prover answers queries, (2) the audit phase, in which the prover updates the set of documents, and (3) rebasing, in which the

[14] The 32-keyword cap is a choice we made for benchmarking.

[15] We estimate that this is a reasonable query budget between updates; larger budgets can be supported by adjusting the masking parameters.

ℓ	Prover	Π_{sub}	Π_{mem}	$\Pi_{\cap}$	Verifier	Proof size (KB)	Peak memory (GB)
Base-only							
2	8.48	1.36	2.79	4.31	0.12	27.11	6.1
4	12.53	1.36	4.18	6.95	0.21	49.73	6.2
8	20.56	1.45	6.39	12.63	0.39	94.98	6.2
16	41.23	1.46	14.24	25.37	0.75	185.47	6.3
Half-full Cache (32 batches)							
2	10.30	1.36	4.20	4.31	0.12	27.11	7.3
4	16.95	1.37	8.15	6.98	0.21	49.73	7.4
8	27.60	1.45	13.01	12.65	0.39	94.98	7.4
16	66.84	1.46	39.44	25.37	0.75	185.47	7.4

Table 2: Audit-assisted Online query performance (Remark 4) for a query with ℓ keywords. The Prover column reports total online proving time; Π_{sub}, Π_{mem}, and $\Pi_{\cap}$ report its subprotocol components. All times are in seconds.

prover recomputes the preprocessing.

Online query phase. For the query-size benchmarks, we consider a system with an initial Base of 838,846 keyword–identifier pairs and compare two settings: *Base-only* and a *half-full* Cache containing 7,834 additional pairs from 32 batches. In Table 2, we show the proving and verification times, proof sizes and peak memory for answering queries with ℓ keywords for $\ell \in \{2, 4, 8, 16\}$. Additionally, we show the runtimes for the subprotocols Π_{sub}, Π_{mem}, and $\Pi_{\cap}$. Each query was chosen to return five documents. Recall that this application does not require sub-second prover times, as agencies are not responding to these requests at scale. Our results demonstrate that proofs can be generated within a reasonable workflow for a government worker.

Comparison with existing systems. We express our conjunctive keyword query as the following SQL over a table `docs(e_key,e_doc)` of keyword–identifier pairs:

```
SELECT e_doc FROM docs
WHERE e_key = q_1 OR ··· OR e_key = q_ℓ
GROUP BY e_doc
HAVING COUNT(*) = ℓ;
```

Since the pairs and query keywords are distinct, the count condition selects exactly the identifiers associated with all ℓ keywords. We prove the corresponding computation in PoneglyphDB [40], RogueDB [64], and TruthTable [57] on the same hardware we use to evaluate our construction and report their online proving and verification times in Table 3.

We use PoneglyphDB [40] as our baseline for zero-knowledge SQL query verification. For the tested Base-only queries with $\ell \in \{2, 4, 8, 16\}$, Sunshine System achieves proving-time speedups of 12.2–23.9× and verification-time speedups of 5.6–32.5× over PoneglyphDB. We also compare with two concurrently developed systems, RogueDB [64] and TruthTable [57]. For the tested Base-only queries, Sunshine System achieves proving-time speedups of 4.3–7.1× over RogueDB and 1.8–26.8× over TruthTable. Verification is 24.3–34.9× faster than RogueDB. TruthTable verifies faster for $\ell \in \{2, 4, 8\}$, whereas Sunshine takes 0.75 s compared with TruthTable's 4.85 s for $\ell = 16$.

Framework	$\ell = 2$	$\ell = 4$	$\ell = 8$	$\ell = 16$
PoneglyphDB [40]	202.53/3.96	237.32/3.85	314.30/3.90	501.22/4.18
RogueDB [64]	36.34/2.97	72.84/6.12	146.76/12.49	292.60/26.02
TruthTable [57]	14.87/0.01	22.45/0.01	54.52/0.02	1,103.90/4.85
Sunshine System [Ours]	8.48/0.12	12.53/0.21	20.56/0.39	41.23/0.75

Table 3: Comparison of online proving and verification times, reported as prover/verifier in seconds.

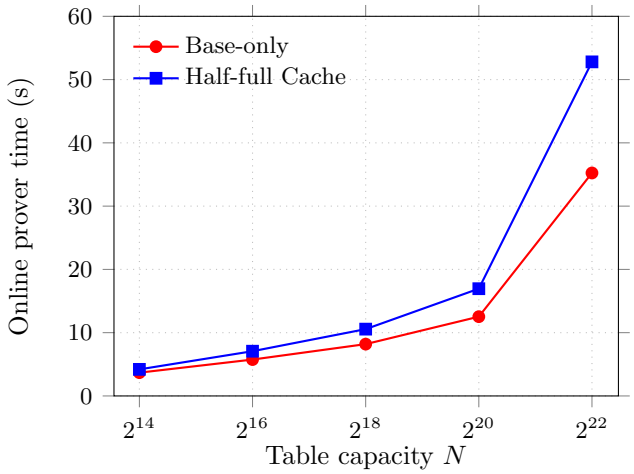


Figure 5: Capacity scaling for four-keyword queries, each returning five identifiers.

Impact of changing the log size. Next, we benchmark the impact of changing the size of the total log capacity N and the extent to which the cache is filled. Figure 5 reports online proving times for $N \in \{2^{14}, 2^{16}, 2^{18}, 2^{20}, 2^{22}\}$ and corresponding capacities $h \in \{2^{10}, 2^{11}, 2^{12}, 2^{13}, 2^{15}\}$, each with the Base at about 80% capacity[16]. We see a dramatic uptick in prover times at 2^{22} in part because the longest keyword-specific identifier list grows from 6,264 to 27,414 identifiers, raising the padded vector capacity from $h = 2^{13}$ to $h = 2^{15}$.

Cache scaling. Since the additional lookup work with non-empty Cache occurs in Π_{mem}, we measure both its prover time and the total online prover time. Under the default configuration, we use a different fixed four-keyword query and vary the Cache size, taking measurements after 1, 16, 32, 48, and 64 update batches. Each batch contains at most 256 keyword–identifier pairs, without splitting a document across batches. The one document returned from the Cache is introduced in the first update batch. Figure 6 shows the resulting proving time.

Our results indicate that the difference between total prover time and the proving time of Π_{mem} remains approximately ten seconds between a nearly empty Cache and a full Cache; even with a full Cache, proof generation times remain reasonable. Proof sizes and verification times remain constant across the measured Cache sizes.

Audit phase. To evaluate the audit phase, we benchmark the runtime of adding $\delta = 7834$ keyword–identifier pairs into the Cache over 32 batches, using our initial Base as the starting point. We found that computing π_{upd} for each batch requires an average of 157.83 s, producing a 317.28 KB proof that could be verified in 2.62 s. These results are for per-update π_{upd} only, and do not include

[16]For $N = 2^{22}$, we process additional Enron emails using the same keyword dictionary and preprocessing rules.

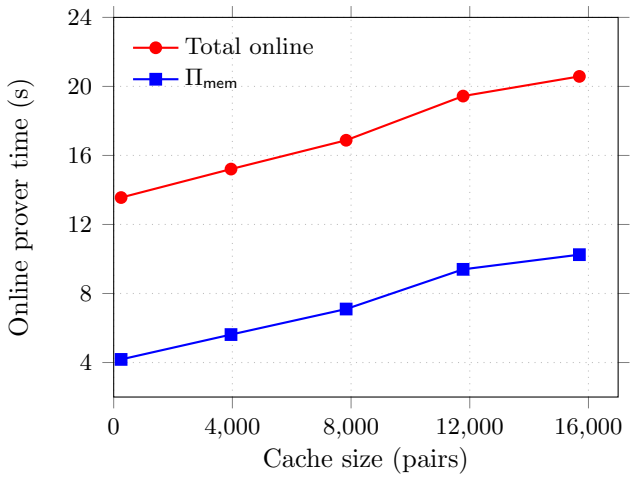


Figure 6: Prover times as a function of the cache size.

Threads	Time (hours)	Speedup	Peak Memory (GB)
1	3.51	1.00×	3.3
4	1.14	3.08×	4.2
8	0.88	4.00×	4.5

Table 4: Rebasing performance with different numbers of threads.

the one-time cost of running Π_{state}; see Section B.2 for details. These figures can be significantly improved with parallelism; with the maximum parallelism available on our prover, prover time dropped to 34.81 s and verification to 1.15 s. This indicates that the overhead of regularly generating these update proofs can reasonably fit into everyday workflows. For example, documents might be accumulated over a day or week, and the agency might generate the corresponding π_{upd} proofs at the end of that period. We also note that the proof size and verification time indicate that it would be reasonable for a government accountability agency or public interest group to verify these update proofs regularly.

Rebasing. When the Cache is full, the prover merges it into the Base and recomputes the table-dependent lookup preprocessing. To evaluate this cost, we begin with a base of size $T_{\mathsf{base}} = 838,846$, add 64 batches with a total of 15,694 keyword–identifier pairs, and then rebase. Table 4 shows rebase performance as a function of the number of threads used to recompute the preprocessing.

Strict single-thread rebasing takes 12,621.93 s, or approximately 3.5 hours; we are able to improve this to under an hour using parallelization. The returns on parallelization start to diminish as the process begins to compete with the operating system for threads. Recall that our setting simply requires the rebase computation to complete overnight during downtime. As such, our results show that our approach significantly exceeds the required performance at the measured scale. The runtime of rebasing is quasilinear in the table size, so larger deployments may require longer maintenance windows or more computational resources. Even so, our results suggest that, with appropriate computational resources, our approach can scale to larger sets of documents without interfering with everyday workflows. If the rebasing becomes too slow, a prover could maintain a small number of separate tables and generate proofs independently with respect to each table.

Discussion. Our evaluation demonstrates that our proposed system could work in real-world

settings, but our evaluation stops short of considering full-scale government agencies. Namely, 1,048,570 keyword–identifier pairs is significant, but police departments are likely to exceed this number over time. We note that the asymptotics of our protocol are actually quite good—after preprocessing, online prover time is independent of the Base size for fixed $m, c, q_{\mathsf{max}}, \ell, h$ and Cache size, allowing proving times to stay reasonably constant as the database grows. Moreover, our other proof components are dependent on Cache size, the number of keywords, and the identifier-vector capacities h and h_D, all of which can be bounded as the total number of documents grows within the configured capacity limits. Rebasing also remains an important factor in long-term scalability.

References

[1] Ezzudin Alkotob, Giulio Berra, Benjamin Beurdouche, Richard Hansen, Daniel Huigens, Dennis Jackson, Cory Francis Myers, and Michael Rosenberg. Trustworthy javascript for the web. Available at https://rwc26.waict.dev/, 2026.

[2] Mike Ananny and Kate Crawford. Seeing without knowing: Limitations of the transparency ideal and its application to algorithmic accountability. *new media & society*, 20(3):973–989, 2018.

[3] Anonymous. Anonymous letter to the Mayor of the Village of Chester regarding Police Chief Timothy McGuire. Available at https://www.documentcloud.org/documents/25255980-mcguire-anonymous-letter/, January 2023. Published by New York Focus and MuckRock, Oct. 2024.

[4] arkworks contributors. `arkworks` zkSNARK ecosystem. https://arkworks.rs, 2022.

[5] Sumeet Bajaj and Radu Sion. Correctdb: Sql engine with practical query authentication. *Proc. VLDB Endow.*, 6(7):529–540, May 2013. doi:10.14778/2536349.2536353.

[6] Kenneth A. Bamberger, Ran Canetti, Shafi Goldwasser, Rebecca Wexler, and Evan J. Zimmerman. Verification dilemmas in law and the promise of Zero-Knowledge proofs. *Berkeley Technology Law Journal*, 37(1):1–70, 2022. URL: https://lawcat.berkeley.edu/record/1257857, doi:10.15779/Z38P55DH8N.

[7] David Banisar. Freedom of information and access to government record laws around the world. Available at https://www.ndi.org/sites/default/files/freeinfo_010504.pdf, May 2004.

[8] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[9] Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to iops and stateless blockchains. In *Annual International Cryptology Conference*, pages 561–586. Springer, 2019.

[10] Nicholas Brandt, Mia Filić, and Sam A. Markelon. A formal treatment of key transparency systems with scalability improvements. Cryptology ePrint Archive, Report 2024/1938, 2024. URL: https://eprint.iacr.org/2024/1938.

[11] Christopher Buchanan. Calif. pd 'systematically' violated state public records law, judge rules. Available at https://www.police1.com/legal/calif-pd-systematically-violated-state-public-records-law-judge-rules, November 2025.

[12] Matteo Campanelli, Antonio Faonio, Dario Fiore, Tianyu Li, and Helger Lipmaa. Lookup arguments: improvements, extensions and applications to zero-knowledge decision trees. In *IACR International Conference on Public-Key Cryptography*, pages 337–369. Springer, 2024.

[13] Matteo Campanelli, Antonio Faonio, Dario Fiore, Anaïs Querol, and Hadrián Rodríguez. Lunar: a toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove

extensions. Cryptology ePrint Archive, Report 2020/1069, 2020. URL: https://eprint.iacr.org/2020/1069.

[14] Matteo Campanelli, Dario Fiore, and Anaïs Querol. LegoSNARK: Modular design and composition of succinct zero-knowledge proofs. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2075–2092. ACM Press, November 2019. doi:10.1145/3319535.3339820.

[15] Melissa Chase, Apoorvaa Deshpande, Esha Ghosh, and Harjasleen Malvai. SEEMless: Secure end-to-end encrypted messaging with less trust. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 1639–1656. ACM Press, November 2019. doi:10.1145/3319535.3363202.

[16] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 422–439. Springer, 2005.

[17] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. Cryptology ePrint Archive, Report 2019/1047, 2019. URL: https://eprint.iacr.org/2019/1047.

[18] Arka Rai Choudhuri, Sanjam Garg, Aarushi Goel, Sruthi Sekar, and Rohit Sinha. Sublonk: Sublinear prover plonk. *Proc. Priv. Enhancing Technol.*, 2024(3):314–335, 2024. URL: https://doi.org/10.56553/popets-2024-0080, doi:10.56553/POPETS-2024-0080.

[19] Premkumar Devanbu, Michael Gertz, Charles Martel, and Stuart G. Stubblebine. Authentic data publication over the internet. *J. Comput. Secur.*, 11(3):291–314, April 2003.

[20] Zijing Di, Lucas Xia, Wilson Nguyen, and Nirvan Tyagi. Muxproofs: Succinct arguments for machine computation from tuple lookups. Cryptology ePrint Archive, Paper 2023/974, 2023. https://eprint.iacr.org/2023/974. URL: https://eprint.iacr.org/2023/974.

[21] Moumita Dutta, Chaya Ganesh, Sikhar Patranabis, Shubh Prakash, and Nitin Singh. Batching-efficient ram using updatable lookup arguments. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 4077–4091, 2024.

[22] Liam Eagen, Dario Fiore, and Ariel Gabizon. cq: Cached quotients for fast lookups. Cryptology ePrint Archive, Report 2022/1763, 2022. URL: https://eprint.iacr.org/2022/1763.

[23] Liam Eagen, Dario Fiore, and Ariel Gabizon. cq: Cached quotients for fast lookups. *Cryptology ePrint Archive*, 2022.

[24] Apple Security Engineering and Architecture. Advancing imessage security: imessage contact key verification. Available at https://security.apple.com/blog/imessage-contact-key-verification/, October 2023.

[25] Andrés Fábrega, Jack Cable, Michael A Specter, and Sunoo Park. Cryptographic verifiability for voter registration systems. *arXiv preprint arXiv:2503.03974*, 2025.

[26] Valerie Fetzer, Michael Klooß, Jörn Müller-Quade, Markus Raiber, and Andy Rupp. Universally composable auditable surveillance. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part II*, volume 14439 of *LNCS*, pages 453–487. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8724-5_14.

[27] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the theory and application of cryptographic techniques*, pages 186–194. Springer, 1986.

[28] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Berlin, Heidelberg, August 1987. doi:10.1007/3-540-47721-7_12.

[29] Jonathan Frankle, Sunoo Park, Daniel Shaar, Shafi Goldwasser, and Daniel J. Weitzner. Practical accountability of secret processes. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018*, pages 657–674. USENIX Association, August 2018. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/frankie.

[30] freedominfo.org. Foi impact: Success stories. Available at https://web.archive.org/web/20231216023032/http://www.freedominfo.org/2016/09/foia-impact-success-stories/, September 2016.

[31] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Annual International Cryptology Conference*, pages 33–62. Springer, 2018.

[32] Ariel Gabizon and Dmitry Khovratovich. flookup: Fractional decomposition-based lookups in quasi-linear time independent of table size. Cryptology ePrint Archive, Report 2022/1447, 2022. URL: https://eprint.iacr.org/2022/1447.

[33] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *Cryptology ePrint Archive*, 2019.

[34] Esha Ghosh, Olga Ohrimenko, Dimitrios Papadopoulos, Roberto Tamassia, and Nikos Triandopoulos. Zero-knowledge accumulators and set algebra. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 67–100. Springer, Berlin, Heidelberg, December 2016. doi:10.1007/978-3-662-53890-6_3.

[35] Shafi Goldwasser and Sunoo Park. Public accountability vs. secret laws: can they coexist? a cryptographic proposal. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society*, pages 99–110, 2017.

[36] Google. Binary transparency. Available at https://binary.transparency.dev/.

[37] Google. Trillian: an open-source verifiable log. Available at https://transparency.dev/.

[38] Matthew Green, Gabriel Kaptchuk, and Gijs Van Laer. Abuse resistant law enforcement access systems. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 553–583. Springer, Cham, October 2021. doi:10.1007/978-3-030-77883-5_19.

[39] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-snarks. In *Annual International Cryptology Conference*, pages 698–728. Springer, 2018.

[40] Binbin Gu, Juncheng Fang, and Faisal Nawab. Poneglyphdb: Efficient non-interactive zero-knowledge proofs for arbitrary sql-query verification. *Proceedings of the ACM on Management of Data*, 3(1):1–27, 2025.

[41] Ulrich Haböck. Multivariate lookups based on logarithmic derivatives. *Cryptology ePrint Archive*, 2022.

[42] Hossein Hafezi, Alireza Shirzad, Benedikt Bünz, and Joseph Bonneau. IronDict: Transparent dictionaries from polynomial commitments. Cryptology ePrint Archive, Report 2025/1580, 2025. URL: https://eprint.iacr.org/2025/1580.

[43] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, Berlin, Heidelberg, December 2010. doi:10.1007/978-3-642-17373-8_11.

[44] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *Machine Learning: ECML 2004*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226. Springer, 2004.

[45] Sean Lawlor and Kevin Lewi. Deploying key transparency at whatsapp. Available at https://engineering.fb.com/2023/04/13/security/whatsapp-key-transparency/, April 2023.

[46] Julia Len, Melissa Chase, Esha Ghosh, Kim Laine, and Radames Cruz Moreno. OPTIKS: An optimized key transparency system. In Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024*. USENIX Association, August 2024. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/len.

[47] Feifei Li, Marios Hadjieleftheriou, George Kollios, and Leonid Reyzin. Dynamic authenticated index structures for outsourced databases. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, page 121–132, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1142473.1142488.

[48] Xiling Li, Chenkai Weng, Yongxin Xu, Xiao Wang, and Jennie Rogers. Zksql: Verifiable and efficient query evaluation with zero-knowledge proofs. *Proceedings of the VLDB Endowment*, 16(8), 2023.

[49] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In *Theory of Cryptography Conference*, pages 169–189. Springer, 2012.

[50] Dave Maass, Aaron Mackey, Beryl Lipton, and Hannah Diaz. The foilies 2025. Available at https://www.eff.org/deeplinks/2025/03/foilies-2025, March 2025.

[51] Harjasleen Malvai, Francesca Falzon, Andrew Zitek-Estrada, Sarah Meiklejohn, and Joseph Bonneau. SoK: Cryptographic authenticated dictionaries. In *NDSS 2026*. The Internet Society, February 2026.

[52] Harjasleen Malvai, Lefteris Kokoris-Kogias, Alberto Sonnino, Esha Ghosh, Ercan Oztürk, Kevin Lewi, and Sean F. Lawlor. Parakeet: Practical key transparency for end-to-end encrypted messaging. In *NDSS 2023*. The Internet Society, February 2023.

[53] Charles Martel, Glen Nuckolls, Premkumar Devanbu, Michael Gertz, April Kwong, and Stuart G Stubblebine. A general model for authenticated data structures. *Algorithmica*, 39(1):21–41, 2004.

[54] Ueli Maurer. Zero-knowledge proofs of knowledge for group homomorphisms. *Designs, Codes and Cryptography*, 77(2):663–676, 2015.

[55] Marcela S. Melara, Aaron Blankstein, Joseph Bonneau, Edward W. Felten, and Michael J. Freedman. CONIKS: Bringing key transparency to end users. In Jaeyeon Jung and Thorsten Holz, editors, *USENIX Security 2015*, pages 383–398. USENIX Association, August 2015. URL: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/melara.

[56] Silvio Micali, Michael Rabin, and Joe Kilian. Zero-knowledge sets. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 80–91. IEEE, 2003.

[57] Bharath Namboothiry, Alireza Shirzad, Spencer Solit, Ryan Marcus, and Pratyush Mishra. Truthtable: A verifiable query engine. *Cryptology ePrint Archive*, 2026.

[58] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *Annual International Cryptology Conference*, pages 111–126. Springer, 2002.

[59] Brian D. Nugent. Report on allegations against Chief McGuire (draft). Available at https://www.documentcloud.org/documents/25255981-mcguire-investigation/, April 2023. Ferick Nugent MacCartney PLLC, prepared for the Mayor and Village Board of the Village of Chester, NY. Published by New York Focus and MuckRock, Oct. 2024.

[60] National Freedom of Information Coalition. Blueprint to transparency: Analyzing non-compliance and enforcement of open records laws in select u.s. states. Available at https://www.nfoic.org/wp-content/uploads/2020-02/Blueprint%20to%20Transparency_%20Enforcing%20Open%20Records%20Laws-4.pdf, February 2020.

[61] National Freedom of Information Coalition. State freedom of information laws. Available at https://www.nfoic.org/state-freedom-of-information-laws/. Accessed on 19 Aug, 2026., 2026.

[62] HweeHwa Pang, Arpit Jain, Krithi Ramamritham, and Kian-Lee Tan. Verifying completeness of relational query results in data publishing. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, SIGMOD '05, page 407–418, New York, NY, USA, 2005. Association for Computing Machinery. doi:10.1145/1066157.1066204.

[63] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Optimal verification of operations on dynamic sets. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 91–110. Springer, Berlin, Heidelberg, August 2011. doi:10.1007/978-3-642-22792-9_6.

[64] Christodoulos Pappas, Zhuo Cai, and Dimitrios Papadopoulos. Rogue: Updatable matrix lookup arguments and applications to verifiable databases. Cryptology ePrint Archive, Paper 2026/1890, 2026. URL: https://eprint.iacr.org/2026/1890.

[65] Jim Posen and Assimakis A. Kattis. Caulk+: Table-independent lookup arguments. Cryptology ePrint Archive, Report 2022/957, 2022. URL: https://eprint.iacr.org/2022/957.

[66] Marek Sefranek. How (not) to simulate plonk. In *International Conference on Security and Cryptography for Networks*, pages 96–117. Springer, 2024.

[67] Srinath Setty, Justin Thaler, and Riad Wahby. Unlocking the lookup singularity with lasso. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 180–209. Springer, 2024.

[68] ACLU SoCal. Court rules inglewood police department systematically violated california public records act. Available here, November 2025.

[69] ACLU SoCal. Aclu socal v. inglewood. Available at https://www.aclusocal.org/cases/aclu-socal-v-inglewood/, April 2026.

[70] Sammy Sussman. Can police departments be trusted to release their own misconduct records? apparently not. Available here., October 2024.

[71] Samuel B. Sussman. Verified petition, *Sussman v. Village of Chester*, index no. ef007571-2024 (n.y. sup. ct., orange county, filed sept. 10, 2024). NYSCEF Doc. No. 1. Available at https://iapps.courts.state.ny.us/fbem/DocumentDisplayServlet?documentId=9xTDB33nszpxFqS5D7OmBg==&system=prod, September 2024.

[72] Roberto Tamassia. Authenticated data structures. In Giuseppe Di Battista and Uri Zwick, editors, *Algorithms - ESA 2003*, pages 2–5, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[73] Nirvan Tyagi, Ben Fisch, Andrew Zitek, Joseph Bonneau, and Stefano Tessaro. VeRSA: Verifiable registries with efficient client audits from RSA authenticated dictionaries. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 2793–2807. ACM Press, November 2022. doi:10.1145/3548606.3560605.

[74] Arantxa Zapico, Vitalik Buterin, Dmitry Khovratovich, Mary Maller, Anca Nitulescu, and Mark Simkin. Caulk: Lookup arguments in sublinear time. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 3121–3134. ACM Press, November 2022. doi:10.1145/3548606.3560646.

[75] Arantxa Zapico, Ariel Gabizon, Dmitry Khovratovich, Mary Maller, and Carla Ràfols. Baloo: Nearly optimal lookup arguments. Cryptology ePrint Archive, Report 2022/1565, 2022. URL: https://eprint.iacr.org/2022/1565.

[76] Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. In *2017 IEEE Symposium on Security and Privacy*, pages 863–880. IEEE Computer Society Press, May 2017. doi:10.1109/SP.2017.43.

[77] Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. A zero-knowledge version of vSQL. Cryptology ePrint Archive, Report 2017/1146, 2017. URL: https://eprint.iacr.org/2017/1146.

[78] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. IntegriDB: Verifiable SQL for outsourced databases. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015*, pages 1480–1491. ACM Press, October 2015. doi:10.1145/2810103.2813711.

A Additional Details for Our Online Query Phase Protocol

A.1 Instantiating Vector Commitments

We instantiate VC using a hiding variant of the KZG polynomial commitment scheme [43]. At a high level, a vector is identified with the evaluations of a low-degree polynomial over a fixed public domain, and we commit to a randomized polynomial that agrees with the vector on this domain.

We work in the structured reference string (SRS) model and assume a public SRS supporting all polynomial degrees required by the construction. Such parameters can be generated using a universal and updatable setup [39].

Structured reference string. The setup samples independent trapdoors $\tau, \eta \xleftarrow{\$} \mathbb{F}^*$. Writing $[a]_j := ag_j$ for $j \in \{1, 2\}$, the SRS contains the required powers $[\tau^i]_1$, $[\tau^i]_2$, and $[\eta\tau^i]_1$, but no η-scaled G_2 powers. The η-scaled G_1 powers hide quotient commitments. All component parameters are derived from this SRS.

Let D_1 and D_2 bound the largest published exponents of τ in G_1 and G_2, respectively, with D_1 covering both G_1 power families. We use the (D_1, D_2)-power discrete logarithm (PDL) assumption [49]: recovering τ from $([\tau^i]_1)_{i=0}^{D_1}$ and $([\tau^i]_2)_{i=0}^{D_2}$ is computationally hard.

Evaluation domain. For a vector of public encoding length d, assume that $\mathbb{F}$ contains a primitive d-th root of unity ω (in particular, $d \mid |\mathbb{F}^*|$), and define the public evaluation domain $\mathbb{H}_d := \{1, \omega, \omega^2, \ldots, \omega^{d-1}\}$. Thus, $\mathbb{H}_d$ is precisely the set of d-th roots of unity in $\mathbb{F}$, and its vanishing polynomial is $Z_{\mathbb{H}_d}(X) = X^d - 1$. Vectors are zero-padded to a supported public domain when necessary; the padding is included in the corresponding relation.

Encoding and commitment. Let $V = (V[1], \ldots, V[d]) \in \mathbb{F}^d$. There is a unique polynomial $f_V \in \mathbb{F}_{<d}[X]$ satisfying $f_V(\omega^{i-1}) = V[i]$ for every $i \in [d]$. We refer to f_V as the interpolation polynomial of V.

Each vector V has a public masking-degree bound t_V. The prover samples $\rho_V \xleftarrow{\$} \mathbb{F}_{<t_V+1}[X]$ and defines $\widehat{f}_V(X) := f_V(X) + (X^d - 1)\rho_V(X)$. This preserves all entries $\widehat{f}_V(\omega^{i-1}) = V[i]$ for every $i \in [d]$.

For each encoding E, let R_E bound the number of distinct off-domain evaluations added by one response, maximized over $|\mathcal{Q}| \le c$, and let A_E cover its other evaluations, including the Π_{state} and Π_{upd} invocations in which E is used. We choose $t_E \ge A_E + q_{\mathsf{max}} R_E$. The additional coefficient accounts for the evaluation $\widehat{f}_E(\tau)$ used in the commitment. An evaluation of a linear combination counts toward the masking budget of each polynomial in that combination; fresh response-specific encodings have separate budgets. All bounds are fixed before commitment. A full response counts all five sub-arguments. In audit-assisted queries, previously generated query-independent proofs are counted once in A_E.

The commitment algorithm outputs $\mathsf{Com}_V := \mathsf{KZG.Commit}(\widehat{f}_V)$ and $\mathsf{op}_V := (V, \rho_V)$. Since $\deg \widehat{f}_V \le d + t_V$, the SRS must support this degree and those of the derived polynomials. Given (V, ρ_V), one can reconstruct the encoding, check its degree bound, and recompute the commitment.

For each persistent database vector $V \in \{E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}\}$, Com_V includes $c_{V,1} := [\widehat{f}_V(\tau)]_1$ and $c_{V,2} := [\widehat{f}_V(\tau)]_2$. The verifier checks $e(c_{V,1}, [1]_2) = e([1]_1, c_{V,2})$. Derived table commitments use the same linear combination in both groups.

Degree checks. We use degree shifting for KZG commitments [17, Section 6.2.2 and Appendix C] with the pairing-based check described in [13, Section 7.8.2]. All shifts are relative to the full published SRS bound. For masked quotient commitments, the check bounds both the ordinary and masking polynomial components. For commitments without an η-scaled component, the verifier also checks consistency of their G_1 and G_2 representations. Degree bounds already checked for the same input commitments are reused.

Evaluating an extracted encoding on $\mathbb{H}_d$ recovers V, and $\rho_V = (\widehat{f}_V - f_V)/(X^d - 1)$ recovers its opening randomness. The enforced encoding-degree bound gives $\deg \rho_V \le t_V$.

Subvector openings. To open a subvector indexed by $\mathcal{S} \subseteq [d]$, the prover opens $\widehat{f}_V$ at the corresponding evaluation points $\mathbb{H}_{\mathcal{S}} := \{\omega^{i-1} : i \in \mathcal{S}\}$. Since $\widehat{f}_V(\omega^{i-1}) = V[i]$, these evaluations

reveal exactly $V|_{\mathcal{S}}$. We realize this efficiently using a batched KZG opening [9] for all points in $\mathbb{H}_{\mathcal{S}}$.

Properties. Completeness follows directly from completeness of the underlying KZG opening protocol. The evaluation-binding property of KZG implies subvector binding: if two accepting openings of the same commitment at the same index set contained different subvectors, then they would disagree at some evaluation point in the corresponding subset of $\mathbb{H}_d$, contradicting evaluation binding. Including the paired representation preserves this property, since conflicting openings would also give conflicting openings of the G_1 component.

For $\tau \notin \mathbb{H}_d$, the evaluation $\widehat{f}_V(\tau)$ and evaluations at any $r \leq t_V$ distinct points outside $\mathbb{H}_d \cup \{\tau\}$ are jointly uniform and independent of V [66, Lemma 1]. The same argument applies adaptively by conditioning on earlier evaluations and reusing the value at repeated points. Given τ, KZG opening proofs can be simulated from the commitment and opened values. The G_1 and G_2 representations encode the same value $\widehat{f}_V(\tau)$, so they add no independent evaluation.

Finally, the construction is linearly homomorphic. For vectors $V_1, V_2 \in \mathbb{F}^d$ and scalars $a, b \in \mathbb{F}$, interpolation is linear: $f_{aV_1+bV_2} = af_{V_1} + bf_{V_2}$. Moreover,

$$a\widehat{f}_{V_1} + b\widehat{f}_{V_2} = f_{aV_1+bV_2} + (X^d - 1)(a\rho_{V_1} + b\rho_{V_2}),$$

which is a randomized encoding of $aV_1 + bV_2$ with masking degree at most $\max\{t_{V_1}, t_{V_2}\}$. Together with the linear homomorphism of KZG commitments, this gives the linear homomorphism required by Definition 1.

For extraction, we follow the AGM polynomial-identity analysis of [12, Appendix C.1]. The independent η-scaled G_1 powers are handled by two PDL reductions: a pair of polynomials (P, Q), not both zero, satisfying $P(\tau) + \eta Q(\tau) = 0$ either gives a nonzero polynomial with root τ when η is known, or reveals η from its coefficients when $P + \eta Q$ is identically zero. The degree and consistency checks therefore yield bounded-degree openings that agree across shared commitments.

A.2 Instantiating $\Pi_{\mathsf{wf}}, \Pi_{\mathsf{cnt}}, \Pi_{\mathsf{sub}}, \Pi_{\mathsf{mem}}$ and $\Pi_{\cap}$

We first describe the auxiliary arguments and then the five Online protocols. They use the SRS and programmable ROM, with knowledge soundness against algebraic adversaries. Each response uses fresh randomness for new auxiliary commitments and proofs. Sub-arguments referring to the same vector use the same commitment. For cost accounting, let $L_{\mathsf{cnt}} := m + t_{E_{\mathsf{cnt}}} + 1$ bound the number of coefficients in the randomized count polynomial.

A.2.1 Auxiliary Relations and Arguments

We start by introducing some notation.

PLONK-style algebraic arguments. Several auxiliary relations reduce to proving that a constraint polynomial G vanishes over a public evaluation domain $\mathbb{H}_d$. We use PLONK-style quotient-polynomial arguments [33] to prove $G(X) = Q(X)Z_{\mathbb{H}_d}(X)$, where $Z_{\mathbb{H}_d}$ is the vanishing polynomial of $\mathbb{H}_d$. We use the randomized encodings above for the input vectors and auxiliary traces recording intermediate computation values. The prover commits to the quotient as $C_Q = [Q(\tau)]_1 + [\eta\rho_Q(\tau)]_1$ with an independent masking polynomial ρ_Q. Its number of independent uniform coefficients exceeds the maximum number of distinct points at which it is opened. After this commitment is fixed, Fiat–Shamir [28] determines an evaluation challenge z. The quotient opening includes $Q(z)$ and $\rho_Q(z)$ and is verified against C_Q using the hiding KZG opening protocol. The verifier checks all openings, the prescribed degree bounds, and $G(z) = Q(z)Z_{\mathbb{H}_d}(z)$.

All checks use the same commitments and evaluations for shared polynomials. Fiat–Shamir inputs bind the public statement and the preceding protocol messages. Throughout this section, we use the term *PLONK-style* to refer to this quotient-polynomial methodology.

For committed vectors V_j and public coefficients a_j, positions t_j, and value u, the same PLONK-style quotient argument proves $\sum_j a_j V_j[t_j] = u$ via $\sum_j a_j \widehat{f}_{V_j}(\xi_j X) - u = (X-1)Q(X)$, where ξ_j is the domain point for $V_j[t_j]$. For a private u, the prover commits to the length-one vector (u), called a singleton, and includes its entry with coefficient -1. Equalities are batched using powers of a fresh Fiat–Shamir challenge after the input commitments are fixed. After the quotient commitments are fixed, a common challenge z is sampled, and the verifier checks the identity using openings of the original commitments at $\xi_j z$ outside their encoding domains. Repeated evaluations of the same polynomial at the same point are reused.

Auxiliary selectors for padded vectors. The relations in Section 6.2.1 operate on fixed-capacity padded committed vectors. To prove statements about their nonzero entries without revealing their cardinalities, our concrete protocols associate each such vector with a *private selector*.

For a vector $V = (v_1, \ldots, v_r, 0, \ldots, 0) \in \mathbb{F}^h, 0 \le r < h$, the prover defines $S_V := (1^r, 0^{h-r}) \in \{0,1\}^h$ and commits to both V and S_V. We refer to the entries $V[i]$ with $S_V[i] = 1$ as the *selected entries* of V. Whenever several auxiliary arguments share a value vector or a selector, the verifier uses the same commitment in all corresponding statements.

For readability, we describe the range, grand-product, and distinctness arguments below for full vectors and suppress selector commitments in their padded invocations. Range checks apply only to selected entries, and adjacent-difference checks apply only to consecutive selected entries; unselected positions contribute 1 to grand products. Multiset cardinality identities concern selected entries, not the public padding capacities.

We now describe the auxiliary relations and arguments. For every committed vector appearing in an auxiliary relation, the witness contains the vector and its opening, and the relation checks that opening. We omit these repeated checks from the constraint lists below. The generic auxiliary cost bounds assume masking degrees at most linear in the input lengths. For operations on the count polynomial, we use L_{cnt} rather than m.

I. Lookup Arguments. A lookup argument proves that every entry of a committed lookup vector appears in a committed or public table.

- **Relation:** Let $U \in \mathbb{F}^d$ be a table vector and $V \in \mathbb{F}^r$ a lookup vector. The statement is $x_{\mathsf{look}} := (\mathsf{Com}_U, \mathsf{Com}_V, d, r)$, and the witness is $w_{\mathsf{look}} := (U, V, \vec{\sigma}, \mathsf{op}_U, \mathsf{op}_V)$, where $\vec{\sigma} = (\sigma_1, \ldots, \sigma_r) \in [d]^r$ specifies the private lookup positions. The relation $\mathcal{R}_{\mathsf{look}}$ requires $V[i] = U[\sigma_i]$ for every $i \in [r]$. Equivalently, every entry of V must occur in U; repeated lookup values and repeated lookup positions are allowed.

  If U is public, its commitment opening is omitted from the witness, and the statement instead includes the public preprocessing information associated with U.

- **Instantiation:** We instantiate $\mathcal{R}_{\mathsf{look}}$ using the exact zero-knowledge lookup argument $\mathsf{zkcq+}$ [12]. Its parameters are derived from our common SRS, following the parameter requirements of that construction. The lookup argument uses Com_V provided by surrounding arguments and samples fresh masks for its internal multiplicity and inverse-polynomial commitments. The verifier uses Com_U and Com_V directly, without separate scalar openings of the polynomials encoding U and V. The verifier checks the resulting lookup identities using the preprocessed table information and a constant number of pairing equations.

- **Cost:** For a lookup vector of public length r, after table-dependent preprocessing, the prover cost is $O(r \log r)_{\mathbb{F}} + O(r)_{\mathbb{G}}$, the verifier cost is $O(\log r)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

II. LogUp Arguments. A LogUp argument proves equality between two weighted multisets.

- **Relation:** Let $U \in \mathbb{F}^r$ and $V \in \mathbb{F}^s$ be value vectors, and let $a \in \mathbb{F}^r$ and $b \in \mathbb{F}^s$ be the corresponding weight vectors. Let $\mathsf{priv} \subseteq \{U, V, a, b\}$ denote the vectors that are private. We define the statement as $x_{\mathsf{logup}} := (\{\mathsf{Com}_W\}_{W \in \mathsf{priv}}, \{W\}_{W \notin \mathsf{priv}}, r, s)$, and the witness as $w_{\mathsf{logup}} := (\{W, \mathsf{op}_W\}_{W \in \mathsf{priv}})$.

  The relation $\mathcal{R}_{\mathsf{logup}}$ requires the rational-function identity

$$\sum_{i=1}^{r} \frac{a[i]}{X - U[i]} = \sum_{j=1}^{s} \frac{b[j]}{X - V[j]}.$$

  Equivalently, the weighted multisets represented by (U, a) and (V, b) are equal. Public vectors and all-one weight vectors are included directly in the statement and omitted from the witness.

- **Instantiation:** We instantiate $\mathcal{R}_{\mathsf{logup}}$ using LogUp [41], with its constraints proved by the PLONK-style quotient arguments above. After the statement commitments are fixed, derive a Fiat–Shamir challenge β outside the represented value domains. The prover commits to weighted inverse vectors and running sums initialized at zero. For each $W \in \mathsf{priv}$, the inverse check opens $\widehat{f}_W$ at one point. The recurrences use the inverse encodings. PLONK-style quotient arguments prove that the sums on both sides of the relation, evaluated at $X = \beta$, equal the value of the same committed singleton.

- **Cost:** The prover cost is $O(r \log r + s \log s)_{\mathbb{F}} + O(r + s)_{\mathbb{G}}$, the verifier cost is $O(\log r + \log s)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

III. Public-Support Arguments. A public-support argument proves that a committed vector is supported on a specified public portion of its domain. We use both a prefix form and an interval form.

- **Relation:** In the *prefix form*, let $V \in \mathbb{F}^d$ and let $0 \le T \le d$ be public. The statement is $x^{\mathsf{pre}}_{\mathsf{supp}} := (\mathsf{Com}_V, T, d)$, and the witness is $w^{\mathsf{pre}}_{\mathsf{supp}} := (V, \mathsf{op}_V)$. The relation $\mathcal{R}^{\mathsf{pre}}_{\mathsf{supp}}$ requires $V[j] = 0$ for every $T < j \le d$. For $T = d$, this condition is vacuous.

  In the *interval form*, let $W \in \mathbb{F}^d$ and $U \in \mathbb{F}^B$, and let T, B be public with $T + B \le d$. The statement is $x^{\mathsf{int}}_{\mathsf{supp}} := (\mathsf{Com}_W, \mathsf{Com}_U, T, B, d)$, and the witness is $w^{\mathsf{int}}_{\mathsf{supp}} := (W, U, \mathsf{op}_W, \mathsf{op}_U)$. The relation $\mathcal{R}^{\mathsf{int}}_{\mathsf{supp}}$ requires $W = 0^T \| U \| 0^{d-T-B}$. The vector $U \in \mathbb{F}^B$ may use a public encoding capacity $N_\Delta \ge B$, in which case Com_U commits to $U \| 0^{N_\Delta - B}$. The encoding capacity is part of the public parameters, and these additional positions are constrained to zero.

- **Instantiation:** We instantiate the prefix form using a PLONK-style quotient-polynomial argument [33] and the interval form using a generalized Schnorr representation argument [54], both over our hiding KZG commitments.

  For the prefix form, let $L^{(d)}_j$ be the Lagrange basis polynomial for position j in $\mathbb{H}_d$, and define $M_T(X) := \sum_{j=T+1}^{d} L^{(d)}_j(X)$. Thus, on $\mathbb{H}_d$, M_T is zero at the first T positions and one at the remaining positions. The prover commits to a quotient polynomial Q_{supp} and proves

$$\widehat{f}_V(X) M_T(X) = Q_{\mathsf{supp}}(X) Z_{\mathbb{H}_d}(X).$$

On $\mathbb{H}_d$, this identity enforces exactly the required zero suffix. In the preprocessed variant, the commitment to M_T is derived deterministically from (d, T) and the SRS. After the commitments are fixed, the verifier checks KZG openings at the Fiat–Shamir challenge point and the corresponding quotient identity.

For the interval form, we use a generalized Schnorr representation argument [54] for the two commitments. It proves that their randomized encodings have the form

$$\widehat{f}_W(X) = \sum_{i=1}^{B} u_i L_{T+i}^{(d)}(X) + Z_{\mathbb{H}_d}(X)\rho_W(X),$$
$$\widehat{f}_U(X) = \sum_{i=1}^{B} u_i L_i^{(N_\Delta)}(X) + Z_{\mathbb{H}_{N_\Delta}}(X)\rho_U(X),$$

using the same hidden values $u_1, \ldots, u_B$ and separate masks of the prescribed degrees. The two representation checks share the value responses. Since the masks vanish on their respective domains, these forms enforce the required window, equality of the corresponding entries, and zero padding of the encoding, without revealing the entries of U.

- **Cost:** For the prefix form, assume that the public evaluation domain has been preprocessed. Computing M_T and its commitment costs $O(d \log d)_{\mathbb{F}} + O(d)_{\mathbb{G}}$. They can be reused while (d, T) is unchanged. Given this preprocessing, the prover cost is $O(d \log d)_{\mathbb{F}} + O(d)_{\mathbb{G}}$, the verifier cost is $O(\log d)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

  For the interval form, given the public Lagrange and masking bases and the commitment openings, the prover cost and verifier cost are both $O(B)_{\mathbb{F}} + O(B)_{\mathbb{G}}$, for a fixed masking degree. Its size is $O(B)_{\mathbb{F}} + O(1)_{\mathbb{G}}$, and it requires no pairings. The prover also reconstructs the coefficients of ρ_W and ρ_U and derives the public bases for their encoding domains. For $N_\Delta \neq d$, these steps add $O(N_\Delta \log N_\Delta)_{\mathbb{F}} + O(N_\Delta \log N_\Delta)_{\mathbb{G}}$ per column. The bases are public and reusable for a fixed SRS and domain, whereas ρ_W and ρ_U depend on the current commitment openings.

IV. Prefix-Selector Arguments. A prefix-selector argument proves that a committed vector is of the form $(1^r, 0^{h-r})$ for some hidden $r < h$.

- **Relation:** Let $S \in \mathbb{F}^h$, for public $h \geq 2$. The statement is $x_{\mathsf{pref}} := (\mathsf{Com}_S, h)$, and the witness is $w_{\mathsf{pref}} := (S, \mathsf{op}_S)$. The relation $\mathcal{R}_{\mathsf{pref}}$ requires that there exists $r \in \{0, \ldots, h-1\}$ such that $S = (1^r, 0^{h-r})$.

- **Instantiation:** Let ω be a primitive h-th root of unity, and let $\widehat{f}_S$ encode S. Define

$$Z_{<h-1}(X) := \prod_{i=0}^{h-2}(X - \omega^i), \qquad Z_{<h-2}(X) := \prod_{i=0}^{h-3}(X - \omega^i).$$

  Using the PLONK-style quotient-polynomial argument described above, the prover commits to quotient polynomials Q_{bool} and Q_{step} and proves

$$\widehat{f}_S(X)\big(\widehat{f}_S(X) - 1\big) = Q_{\mathsf{bool}}(X) Z_{<h-1}(X)$$

  and

$$\widehat{f}_S(\omega X)\big(1 - \widehat{f}_S(X)\big) = Q_{\mathsf{step}}(X) Z_{<h-2}(X).$$

It additionally gives a KZG opening proving $\widehat{f}_S(\omega^{h-1}) = 0$. The first identity enforces booleanity on the first $h-1$ positions, the second rules out a transition from 0 to 1, and the final opening reserves the last position for padding. Together, these conditions imply $S = (1^r, 0^{h-r})$ for some hidden $r < h$.

- **Cost:** The prover cost is $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$, the verifier cost is $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

V. Zero-Padding Arguments. A zero-padding argument proves that every unselected entry of a committed value vector is zero.

- **Relation:** Let $X, S \in \mathbb{F}^h$. The statement is $x_{\mathsf{zpad}} := (\mathsf{Com}_X, \mathsf{Com}_S, h)$, and the witness is $w_{\mathsf{zpad}} := (X, S, \mathsf{op}_X, \mathsf{op}_S)$. The relation $\mathcal{R}_{\mathsf{zpad}}$ requires $(1 - S[i])X[i] = 0$ for every $i \in [h]$.
- **Instantiation:** Let $\widehat{f}_X$ and $\widehat{f}_S$ denote the polynomials underlying the hiding KZG commitments to X and S. The prover commits to a quotient polynomial Q_{zpad} and proves

$$\big(1 - \widehat{f}_S(X)\big)\widehat{f}_X(X) = Q_{\mathsf{zpad}}(X) Z_{\mathbb{H}_h}(X)$$

  using a PLONK-style quotient-polynomial argument.
- **Cost:** The prover cost is $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$, the verifier cost is $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

VI. Padding Arguments. A padding argument proves that a committed selector identifies a prefix and that every unselected entry of the corresponding value vector is zero.

- **Relation:** Let $X, S_X \in \mathbb{F}^h$. The statement is $x_{\mathsf{pad}} := (\mathsf{Com}_X, \mathsf{Com}_{S_X}, h)$, and the witness is $w_{\mathsf{pad}} := (X, S_X, \mathsf{op}_X, \mathsf{op}_{S_X})$. The relation $\mathcal{R}_{\mathsf{pad}}$ requires both $\mathcal{R}_{\mathsf{pref}}$ and $\mathcal{R}_{\mathsf{zpad}}$ to hold. Equivalently, there exists $r < h$ such that $S_X = (1^r, 0^{h-r})$ and $X[i] = 0$ for every $r < i \le h$.
- **Instantiation:** We instantiate Π_{pad} by composing Π_{pref} and Π_{zpad}, reusing the same selector commitment Com_{S_X} in both component arguments.
- **Cost:** The prover cost is $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$, the verifier cost is $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

VII. Range Arguments. A range argument proves that every entry of a committed vector lies in a specified public interval.

- **Relation:** Let $V \in \mathbb{F}^h$ and let $a \le b$ be public integers whose canonical embeddings in $\mathbb{F}$ are understood. The statement is $x_{\mathsf{range}} := (\mathsf{Com}_V, a, b, h)$, and the witness is $w_{\mathsf{range}} := (V, \mathsf{op}_V)$. The relation $\mathcal{R}_{\mathsf{range}}$ requires $V[i] \in \{a, a+1, \ldots, b\}$ for every $i \in [h]$.
- **Instantiation:** We instantiate $\mathcal{R}_{\mathsf{range}}$ using $\mathsf{zkcq}+$ [12] with the public table $(a, a+1, \ldots, b)$. The table is padded to a supported domain by repeating b, preserving the exact interval. Its preprocessing is reused across invocations.
- **Cost:** After preprocessing the public range table, the prover cost is $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$, the verifier cost is $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

VIII. Cardinality Arguments. A cardinality argument proves that the number of positions selected by a committed selector equals a hidden value contained in another committed vector.

- **Relation:** Let $S \in \mathbb{F}^h$ and $V \in \mathbb{F}^d$, and let $t \in [d]$ be a public position. The statement is $x_{\mathsf{card}} := (\mathsf{Com}_S, \mathsf{Com}_V, t, h, d)$, and the witness is $w_{\mathsf{card}} := (S, V, \mathsf{op}_S, \mathsf{op}_V)$. The relation $\mathcal{R}_{\mathsf{card}}$ requires

$$\sum_{j=1}^{h} S[j] = V[t].$$

  Selector validity is established separately by Π_{pref}. Thus, when $h < |\mathbb{F}|$, this sum equals the number of selected positions.

- **Instantiation:** The prover commits to a running-sum vector $A \in \mathbb{F}^h$ and uses PLONK-style checks to prove $A[1] = 0$ and $A[j+1] - A[j] = S[j]$ for $j \in [h-1]$. It commits to a private singleton (c) and uses the PLONK-style quotient argument above to prove $(A + S)[h] = c = V[t]$ with respect to $\mathsf{Com}_A + \mathsf{Com}_S$, Com_V, and the same singleton commitment, without opening c.

  For batched cardinality, fix $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$ and all selector commitments, then derive $\rho \in \mathbb{F} \setminus \{0\}$ via Fiat–Shamir. Set $\overline{S} := \sum_{i=1}^{\ell} \rho^{i-1} S_i$ and $c_\rho := \sum_{i=1}^{\ell} \rho^{i-1} E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$. The prover commits to (c_ρ) and uses the PLONK-style quotient argument above to prove the defining equality for c_ρ against $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$ and the singleton commitment. The prefix form of Π_{supp} proves the prescribed zero padding of $E^{\mathcal{Q}}_{\mathsf{cnt}}$. The verifier derives $\mathsf{Com}_{\overline{S}}$ homomorphically and applies the running-sum construction once, using the same singleton commitment. Prefix checks apply to the original selectors, not to $\overline{S}$.

- **Cost:** For its use in an ℓ-keyword query, the cardinality encodings use $O(\ell)$ masking coefficients. Given prepared public domain bases, a single invocation has prover cost $O((h + d + \ell)\log(h + d + \ell))_{\mathbb{F}} + O(h + d + \ell)_{\mathbb{G}}$, verifier cost $O(\log h + \log d + \ell)_{\mathbb{F}} + O(\ell)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and proof size $O(\ell)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

  For a batched invocation, the prover cost is $O(\ell h + (h + \ell)\log(h + \ell))_{\mathbb{F}} + O(\ell h + \ell \log \ell)_{\mathbb{G}}$. The verifier cost is $O(\log h + \ell)_{\mathbb{F}} + O(\ell \log \ell + \ell)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$ and the proof size is $O(\ell)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

IX. Grand-Product Arguments. A grand-product argument proves multiset equality or multiset decomposition among committed vectors.

- **Relation:** For *multiset equality*, let $X, Y \in \mathbb{F}^r$. The statement is $x^{\mathsf{eq}}_{\mathsf{gp}} := (\mathsf{Com}_X, \mathsf{Com}_Y, r)$, and the witness is $w^{\mathsf{eq}}_{\mathsf{gp}} := (X, Y, \mathsf{op}_X, \mathsf{op}_Y)$. The relation $\mathcal{R}^{\mathsf{eq}}_{\mathsf{gp}}$ requires that X and Y induce the same multiset, denoted $X \equiv Y$.

  For *multiset decomposition*, let $X \in \mathbb{F}^r$, $Y \in \mathbb{F}^s$, and $Z \in \mathbb{F}^t$, where $r = s + t$. The statement is $x^{\mathsf{dec}}_{\mathsf{gp}} := (\mathsf{Com}_X, \mathsf{Com}_Y, \mathsf{Com}_Z, r, s, t)$, and the witness is $w^{\mathsf{dec}}_{\mathsf{gp}} := (X, Y, Z, \mathsf{op}_X, \mathsf{op}_Y, \mathsf{op}_Z)$. The relation $\mathcal{R}^{\mathsf{dec}}_{\mathsf{gp}}$ requires $X \equiv Y \uplus Z$. If one of the vectors is public, it is included directly in the statement and omitted from the witness.

- **Instantiation:** We instantiate these relations using a PLONK-style grand-product argument [33]. After the input-vector commitments and any selector commitments are fixed, the prover and verifier derive $\gamma \xleftarrow{\$} \mathbb{F}$ via Fiat–Shamir. Multiset equality is reduced to

$$\prod_{i=1}^{r} (X[i] + \gamma) = \prod_{i=1}^{r} (Y[i] + \gamma),$$

while multiset decomposition is reduced to

$$\prod_{i=1}^{r}(X[i]+\gamma) = \left(\prod_{j=1}^{s}(Y[j]+\gamma)\right)\left(\prod_{k=1}^{t}(Z[k]+\gamma)\right).$$

The prover commits to masked running-product traces and to the products over X, Y, and, when present, Z as private singletons. PLONK-style quotient arguments prove the recurrences, the equality of each computed product to its committed value, and the displayed product identities. Each input value or selector encoding is opened at at most two off-domain points. Only prescribed public boundary values are opened on the encoding domain.

- **Cost:** If D denotes the total public length of the input vectors, the prover cost is $O(D \log D)_{\mathbb{F}} + O(D)_{\mathbb{G}}$, the verifier cost is $O(\log D)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

X. Distinctness Arguments. A distinctness argument proves that the entries of a committed vector are pairwise distinct.

- **Relation:** Let $V \in \mathbb{F}^h$. The statement is $x_{\mathsf{dist}} := (\mathsf{Com}_V, h)$, and the witness is $w_{\mathsf{dist}} := (V, \mathsf{op}_V)$. The relation $\mathcal{R}_{\mathsf{dist}}$ requires $V[i] \neq V[j]$ for every distinct $i, j \in [h]$.
- **Instantiation:** In the Online and Audit protocols, the entries of V have already been proved to lie in $[0, U]$, canonically embedded in $\mathbb{F}$, with respect to Com_V. We reuse these bounds and choose a public integer $M \geq \max\{1, U\}$ with $U + M < |\mathbb{F}|$. The prover commits to a sorted vector V^{sort} and invokes Π_{gp} to prove $V \equiv V^{\mathsf{sort}}$. It then commits to the adjacent-difference vector $D[i] := V^{\mathsf{sort}}[i+1] - V^{\mathsf{sort}}[i]$ for $1 \leq i < h$. A PLONK-style quotient argument [33] proves these difference constraints against the same committed vectors.

  To prove $D[i] \in [1, M]$, we either invoke Π_{range} directly or use limb decomposition. For the latter, choose positive integers w, k with $M = 2^{wk}$ and commit to vectors $L_0, \ldots, L_{k-1}$ satisfying

$$D[i] - 1 = \sum_{j=0}^{k-1} 2^{wj}(L_j[i] - 1).$$

  A PLONK-style quotient argument proves this reconstruction using the same commitment to D, while Π_{range} proves $L_j[i] \in [1, 2^w]$ for every i, j. Thus, $L_j[i] - 1$ are the base-2^w digits of $D[i] - 1$. This construction uses a public table of size 2^w rather than M.

  Since Π_{gp} transfers the bound $[0, U]$ from V to V^{sort}, a negative difference has field representative at least $|\mathbb{F}| - U > M$. The positive range also excludes zero, so V^{sort} is strictly increasing and contains no repeated value.

- **Cost:** The cost bounds here and for the composed protocols assume a constant number of limbs. For variable k, the cryptographic costs and proof size increase by at most a factor of $O(k)$; the sorting cost is unchanged. The prover performs $O(h \log h)$ sorting work, in addition to $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$ cryptographic work. The verifier cost is $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

A.2.2 Protocol Π_{wf}

Recall that the statement is $x_{\mathsf{wf}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$, and the witness is $w_{\mathsf{wf}} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}})$. The protocol proves that the first T entries encode valid and distinct keyword–identifier pairs, while all remaining entries are zero.

Prover. The prover proceeds as follows:

- Invokes Π_{supp} on $(\mathsf{Com}_{E_{\mathsf{key}}}, T)$ and $(\mathsf{Com}_{E_{\mathsf{doc}}}, T)$ to prove $E_{\mathsf{key}}[j] = E_{\mathsf{doc}}[j] = 0$ for every $T < j \le N$.
- Invokes Π_{range} to prove $E_{\mathsf{key}}[j] \in [m]$ and $E_{\mathsf{doc}}[j] \in [n]$ for every $j \in [T]$.
- Defines $P := E_{\mathsf{key}} + (m+1)E_{\mathsf{doc}}$. Using linear homomorphism, computes $\mathsf{Com}_P := \mathsf{Com}_{E_{\mathsf{key}}} + (m+1)\mathsf{Com}_{E_{\mathsf{doc}}}$. Invokes Π_{dist} on Com_P to prove that $P[j]$ for $j \in [T]$ are pairwise distinct.[17]
- Outputs the two support proofs, the two range proofs, and the distinctness proof.

Verifier. The verifier computes $\mathsf{Com}_P := \mathsf{Com}_{E_{\mathsf{key}}} + (m+1)\mathsf{Com}_{E_{\mathsf{doc}}}$ using linear homomorphism, verifies all subproofs, and accepts if and only if all of them accept.

Lemma 2 (Protocol Π_{wf})**.** *Let $U := m + (m+1)n$. Assume $U \le M$ and $U + M < |\mathbb{F}|$, where M is the positive-difference bound used by Π_{dist}. Assume that VC satisfies Definition 1, and that Π_{supp}, Π_{range}, and Π_{dist} satisfy Definition 2 for their respective relations. Then Π_{wf} described above satisfies Definition 2 for $\mathcal{R}_{\mathsf{wf}}$ in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries; see Section 4.2.*

Proof sketch. Soundness of Π_{supp} establishes that both committed columns are zero outside the first T positions. Soundness of Π_{range} establishes that the first T entries of E_{key} and E_{doc} lie in $[m]$ and $[n]$, respectively. Linear homomorphism of VC binds Com_P to $P := E_{\mathsf{key}} + (m+1)E_{\mathsf{doc}}$. Under the assumption $m + (m+1)n < |\mathbb{F}|$, the map $(k, \mathsf{id}) \mapsto k + (m+1)\mathsf{id}$ is injective over $[m] \times [n]$. Soundness of Π_{dist} therefore implies that no keyword–identifier pair is repeated among the first T positions. These are exactly the conditions of $\mathcal{R}_{\mathsf{wf}}$.

Completeness follows from the honest representation. Zero knowledge follows from zero knowledge of Π_{supp}, Π_{range}, and Π_{dist}, together with hiding of VC. □

Efficiency. The prover invokes Π_{supp} twice, Π_{range} twice, and Π_{dist} once, on vectors of public capacity N. After the required preprocessing, each invocation costs $O(N \log N)_{\mathbb{F}} + O(N)_{\mathbb{G}}$. Since there are only five invocations, their total prover cost is $O(N \log N)_{\mathbb{F}} + O(N)_{\mathbb{G}}$. Π_{dist} additionally requires $O(N \log N)$ sorting work. Combining the component verifiers gives $O(\log N + \log m + \log n)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$. Each component contributes a constant-size proof, so the total proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

A.2.3 Protocol Π_{cnt}

Recall that the statement is $x_{\mathsf{cnt}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$, and the witness is $w_{\mathsf{cnt}} := (E_{\mathsf{key}}, E_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{cnt}}})$. The protocol proves that $E_{\mathsf{cnt}}[k]$ equals the number of occurrences of keyword k among the first T entries of E_{key}. In the concrete protocols, Π_{cnt} uses the same $\mathsf{Com}_{E_{\mathsf{key}}}$ and T as Π_{wf}, which proves $E_{\mathsf{key}}[j] \in \mathcal{K}$ for every $j \in [T]$.

Prover. The prover proceeds as follows:

[17] The concrete implementation restricts this distinctness check using the public prefix selector $(1^T, 0^{N-T})$. Since this selector is determined by the public value T, we omit it according to the auxiliary-selector convention in Section A.2.1.

- After the statement commitments are fixed, derives $\beta \xleftarrow{\$} \mathbb{F} \setminus (\{0\} \cup [m])$ via Fiat–Shamir.
- Invokes LogUp [41] to prove
$$\sum_{j=1}^{T} \frac{1}{\beta - E_{\mathsf{key}}[j]} = \sum_{k=1}^{m} \frac{E_{\mathsf{cnt}}[k]}{\beta - k}.$$
- Outputs the resulting LogUp proof π_{cnt}.

Verifier. The verifier derives the same Fiat–Shamir challenge β and accepts if and only if the LogUp proof verifies.

Lemma 3 (Protocol Π_{cnt})**.** *Assume that* VC *satisfies Definition 1, that* Π_{wf} *has the guarantees of Lemma 2, and that* LogUp *[41] satisfies Definition 2 for* $\mathcal{R}_{\mathsf{logup}}$*. When used with* Π_{wf} *on the same* $\mathsf{Com}_{E_{\mathsf{key}}}$ *and* T*,* Π_{cnt} *is complete and zero knowledge. Its joint execution with* Π_{wf} *provides knowledge soundness for* $\mathcal{R}_{\mathsf{cnt}}$ *against algebraic adversaries in the SRS and programmable ROM.*

Proof sketch. Knowledge soundness of Π_{wf} ensures that $E_{\mathsf{key}}[j] \in \mathcal{K}$ for every $j \in [T]$. Grouping the terms on the left-hand side by keyword therefore gives each $k \in \mathcal{K}$ a coefficient equal to its number of occurrences. Soundness of LogUp, together with binding of the shared keyword commitment, implies, except with negligible probability, $E_{\mathsf{cnt}}[k] = |\{j \in [T] : E_{\mathsf{key}}[j] = k\}|$ for every $k \in \mathcal{K}$, which is the condition of $\mathcal{R}_{\mathsf{cnt}}$.

Completeness follows from the honest counts. Zero knowledge follows from the corresponding properties of the component arguments and hiding of VC. □

Efficiency. The protocol invokes LogUp once, with public input capacities N and m. Accounting for the randomized count polynomial, the prover cost is $O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}})_{\mathbb{F}} + O(N + L_{\mathsf{cnt}})_{\mathbb{G}}$. The verifier cost is $O(\log N + \log m)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

A.2.4 Protocol Π_{sub}

Recall that the statement is $x_{\mathsf{sub}} := (\mathsf{Com}_{E_{\mathsf{cnt}}}, \mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \mathcal{Q})$, and the witness is $w_{\mathsf{sub}} := (E_{\mathsf{cnt}}, E^{\mathcal{Q}}_{\mathsf{cnt}}, \mathsf{op}_{E_{\mathsf{cnt}}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}})$. The protocol proves that $E^{\mathcal{Q}}_{\mathsf{cnt}}$ contains the entries of E_{cnt} corresponding to the queried keywords.

Prover. The prover proceeds as follows:

- Writes $\mathcal{Q} = \{q_1, \ldots, q_\ell\}$ in canonical order.
- Uses the PLONK-style quotient argument of Section A.2 to prove $E^{\mathcal{Q}}_{\mathsf{cnt}}[i] = E_{\mathsf{cnt}}[q_i]$ for every $i \in [\ell]$. After $\mathsf{Com}_{E_{\mathsf{cnt}}}$ and $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$ are fixed, the equalities are batched into one quotient check. Each of $\mathsf{Com}_{E_{\mathsf{cnt}}}$ and $\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}$ is opened at at most ℓ points obtained by multiplying the domain points of the selected entries by the same challenge z. The selected counts are not opened.
- Outputs the resulting proof π_{sub}.

Verifier. The verifier forms the constraints determined by $\mathcal{Q}$ and accepts if and only if the PLONK-style proof verifies.

Lemma 4 (Protocol Π_{sub})**.** *Assume that* VC *satisfies Definition 1, and that the PLONK-style proving system used above satisfies Definition 2 for the corresponding constraints. Then* Π_{sub} *described above satisfies Definition 2 for* $\mathcal{R}_{\mathsf{sub}}$ *in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries.*

Proof sketch. The proved constraints are exactly the defining conditions of $\mathcal{R}_{\mathsf{sub}}$. Knowledge soundness of the PLONK-style argument therefore implies $E^{\mathcal{Q}}_{\mathsf{cnt}}[i] = E_{\mathsf{cnt}}[q_i]$ for every $i \in [\ell]$. Completeness follows from the honest construction of $E^{\mathcal{Q}}_{\mathsf{cnt}}$, while zero knowledge follows from zero knowledge of the PLONK-style argument and hiding of VC. □

Efficiency. With preprocessed public domains and commitment bases, constructing the batched constraint polynomial, its quotient, and the opening proofs costs $O(L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + \ell \log^2 \ell)_{\mathbb{F}}$. Commitment generation costs $O(L_{\mathsf{cnt}} + \ell)_{\mathbb{G}}$.

The verifier prepares only the public query-dependent polynomials and checks the batched proof, costing $O(\ell \log^2 \ell)_{\mathbb{F}} + O(\ell)_{\mathbb{G}} + O(1)_{\mathsf{pairing}}$. The proof contains $O(\ell)$ field elements and a constant number of group elements, giving size $O(\ell)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

A.2.5 Protocol Π_{mem}

Recall that the statement is $x_{\mathsf{mem}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T, \mathcal{Q}, \mathsf{Com}_{\mathsf{aux}})$, where $\mathsf{Com}_{\mathsf{aux}} := (\mathsf{Com}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$, and the witness is

$$w_{\mathsf{mem}} := \big(E_{\mathsf{key}}, E_{\mathsf{doc}}, E^{\mathcal{Q}}_{\mathsf{cnt}}, \{E^{q_i}_{\mathsf{Enc}}\}_{i\in[\ell]}, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E^{\mathcal{Q}}_{\mathsf{cnt}}}, \{\mathsf{op}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]}\big).$$

The protocol proves that each query-specific vector $E^{q_i}_{\mathsf{Enc}}$ contains distinct valid identifiers associated with q_i, with exactly $E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$ selected entries and zero padding elsewhere. In the complete online protocol, Π_{wf} proves $\mathcal{R}_{\mathsf{wf}}$ for the same $(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$. Π_{mem} does not repeat these checks.

For every $i \in [\ell]$, the prover additionally constructs and commits to an auxiliary selector $S_i \in \mathbb{F}^h$. These selector commitments are proof messages of Π_{mem} and are not part of the statement.

Prover. The prover proceeds as follows:

- For every $i \in [\ell]$, sets $c_i := E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$, constructs $S_i = (1^{c_i}, 0^{h-c_i})$, commits to S_i, and invokes Π_{pad} on $(\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}, \mathsf{Com}_{S_i}, h)$ to prove that S_i identifies a prefix of $E^{q_i}_{\mathsf{Enc}}$ and that every unselected entry is zero.
- Invokes Π_{card} in batch to prove $\sum_{j=1}^{h} S_i[j] = E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$ for every $i \in [\ell]$.
- For every $i \in [\ell]$, invokes Π_{range} to prove that every entry of $E^{q_i}_{\mathsf{Enc}}$ selected by S_i lies in $[n]$, and invokes Π_{dist} to prove that these selected entries are pairwise distinct.
- After the statement commitments and $\{\mathsf{Com}_{S_i}\}_{i\in[\ell]}$ are fixed, derives $\alpha \xleftarrow{\$} \mathbb{F} \setminus \{0\}$ via Fiat–Shamir. Defines $E^{(\alpha)} := E_{\mathsf{key}} + \alpha E_{\mathsf{doc}}$ and, for every $i \in [\ell]$, $A_i := q_i S_i + \alpha E^{q_i}_{\mathsf{Enc}}$. Using linear homomorphism, derives
$$\mathsf{Com}_{E^{(\alpha)}} := \mathsf{Com}_{E_{\mathsf{key}}} + \alpha \mathsf{Com}_{E_{\mathsf{doc}}}$$
and
$$\mathsf{Com}_{A_i} := q_i \mathsf{Com}_{S_i} + \alpha \mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}.$$
- For every $i \in [\ell]$, invokes $\mathsf{zkcq}+$ [12] for $\mathcal{R}_{\mathsf{look}}$ to prove that every entry of A_i appears in the table $E^{(\alpha)}$.

  At a selected position j, the lookup value is $q_i + \alpha E^{q_i}_{\mathsf{Enc}}[j]$. At an unselected position, $S_i[j] = E^{q_i}_{\mathsf{Enc}}[j] = 0$, so $A_i[j] = 0$. The reserved final $(0, 0)$ row guarantees that $E^{(\alpha)}$ contains zero even when $T = N - 1$, allowing every padding entry to satisfy the same lookup check.

- Outputs the selector commitments, the batched cardinality proof, and the padding, range, distinctness, and lookup proofs.

Verifier. The verifier derives the same Fiat–Shamir challenge α, reconstructs $\mathsf{Com}_{E^{(\alpha)}}$ and each Com_{A_i} using linear homomorphism, verifies all subproofs, and accepts if and only if all of them accept.

Lemma 5 (Protocol Π_{mem})**.** *Assume that* VC *satisfies Definition 1 and that* $(\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$ *has been certified by an accepted* Π_{wf} *proof. Assume further that* Π_{pad}, Π_{card}, Π_{range}, *and* Π_{dist} *satisfy Definition 2 for their respective relations, and that* zkcq+ *[12] satisfies Definition 2 for* $\mathcal{R}_{\mathsf{look}}$*. Then* Π_{mem} *described above satisfies Definition 2 for* $\mathcal{R}_{\mathsf{mem}}$ *in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries.*

Proof sketch. Completeness follows from the honest query-specific vectors and their prefix selectors.

For knowledge soundness, the padding and cardinality relations establish that $E^{q_i}_{\mathsf{Enc}}$ has exactly $E^{\mathcal{Q}}_{\mathsf{cnt}}[i]$ selected entries and that every remaining entry is zero. The range and distinctness relations establish that the selected entries are distinct identifiers in $[n]$. The checks use the same commitments for shared vectors, so subvector binding links these conditions to the same query-specific vectors and counts. Linear homomorphism links the lookup commitments to $E^{(\alpha)}$ and A_i defined above.

By soundness of zkcq+ for $\mathcal{R}_{\mathsf{look}}$, every selected position j has a matching table position $k \in [N]$ such that $q_i + \alpha E^{q_i}_{\mathsf{Enc}}[j] = E_{\mathsf{key}}[k] + \alpha E_{\mathsf{doc}}[k]$. Since the statement commitments and $\{\mathsf{Com}_{S_i}\}_{i\in[\ell]}$ are fixed before α is derived, except with negligible probability this equality implies $E_{\mathsf{key}}[k] = q_i$ and $E_{\mathsf{doc}}[k] = E^{q_i}_{\mathsf{Enc}}[j]$. Once the two equations hold, the matching position cannot lie after T, where both columns are zero. Hence $k \in [T]$, and every selected identifier is associated with q_i. These are exactly the conditions of $\mathcal{R}_{\mathsf{mem}}$.

Zero knowledge follows from zero knowledge of Π_{pad}, Π_{card}, Π_{range}, Π_{dist}, and zkcq+, together with hiding of VC. □

Remark 8. *We apply the Base+Cache framework of Dutta et al. [21] to the table-dependent preprocessing used by* Π_{mem}*. The table represented by a previously audited commitment forms the* Base*, while pairs appended later form the* Cache*. The membership protocol reuses the lookup preprocessing associated with the Base and incorporates the Cache through a sparse correction. This optimization affects only the proof of* $\mathcal{R}_{\mathsf{mem}}$ *and does not change the relation itself.*

Efficiency. For every queried keyword, the prover invokes Π_{pad}, Π_{range}, Π_{dist}, and zkcq+ once on vectors of public capacity h. After lookup preprocessing, these invocations cost $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$ per keyword. The cardinality equalities are proved in one batch invocation of Π_{card}. Summing these costs gives the Base-only prover bound $O(\ell h \log h + \ell \log \ell)_{\mathbb{F}} + O(\ell h + \ell \log \ell)_{\mathbb{G}}$.

For a Cache of size δ, the sparse correction adds one δ-dependent term and a correction for each of the ℓ lookups, for an additional cost of $O\big(\delta \log^2 \delta + \ell(\delta + h) \log^2(\delta + h)\big)_{\mathbb{F}} + O(\ell(\delta + h))_{\mathbb{G}}$. Summing their verification costs gives $O(\ell \log h + \ell)_{\mathbb{F}} + O(\ell \log \ell + \ell)_{\mathbb{G}}$ together with $O(\ell)$ pairing terms accumulated into one final multi-pairing check. The proof size is $O(\ell)_{\mathbb{F},\mathbb{G}}$. The Cache does not change the verifier cost or proof size.

A.2.6 Protocol $\Pi_{\cap}$

Recall that the statement is $x_{\cap} := (\mathcal{J}, \{\mathsf{Com}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$, and the witness is $w_{\cap} := (\{E^{q_i}_{\mathsf{Enc}}\}_{i\in[\ell]}, \{\mathsf{op}_{E^{q_i}_{\mathsf{Enc}}}\}_{i\in[\ell]})$. The protocol proves that $\mathcal{J} = \bigcap_{i=1}^{\ell} \mathcal{E}^{q_i}_{\mathsf{doc}}$.

In the complete online protocol, Π_{mem} has already certified the query-specific padded vectors and their auxiliary selectors. The protocol $\Pi_{\cap}$ reuses these commitments and does not repeat the padding, range, or distinctness proofs for its inputs. When $\Pi_{\cap}$ is invoked without Π_{mem}, it includes these padding, range, and distinctness checks.

Prover. The prover proceeds as follows:

- Defines $A_{\mathcal{J}}(Z) := \prod_{x\in\mathcal{J}}(Z-x)$ and $P_i(Z) := \prod_{x\in\mathcal{E}^{q_i}_{\mathsf{doc}}\setminus\mathcal{J}}(Z-x)$ for every $i \in [\ell]$. Computes polynomials B_i of degree less than h such that $\sum_{i=1}^{\ell} B_i(Z)P_i(Z) = 1$ using the extended Euclidean algorithm.
- Commits to the coefficient vectors of P_i and B_i, each padded to length h. After fixing these and the input commitments, derives $\gamma \xleftarrow{\$} \mathbb{F}\setminus\{0\}$ via Fiat–Shamir.
- Invokes a PLONK-style argument [33] to prove $\prod_{x\in\mathcal{E}^{q_i}_{\mathsf{doc}}}(\gamma-x) = A_{\mathcal{J}}(\gamma)P_i(\gamma)$ for each $i \in [\ell]$ and $\sum_{i=1}^{\ell} B_i(\gamma)P_i(\gamma) = 1$, using the fixed coefficient commitments and the identifier and selector commitments certified by Π_{mem}. The values $P_i(\gamma)$, $B_i(\gamma)$, and $\prod_{x\in\mathcal{E}^{q_i}_{\mathsf{doc}}}(\gamma-x)$ remain private. The polynomials encoding $E^{q_i}_{\mathsf{Enc}}$ and S_i are each opened at at most two off-domain points.
- Outputs all intermediate commitments and subproofs.

Verifier. The verifier derives the same challenges and computes $A_{\mathcal{J}}(\gamma)$ from the public response. It verifies all subproofs, and accepts if and only if all of them accept.

Lemma 6 (Protocol $\Pi_{\cap}$)**.** *Assume that* VC *satisfies Definition 1 and that the input query-specific vectors have been certified by* Π_{mem}*. Assume further that the KZG-based PLONK-style argument [33] satisfies Definition 2 for the stated constraints on the input vectors and the coefficient vectors of* P_i *and* B_i*. Then* $\Pi_{\cap}$ *described above satisfies Definition 2 for* $\mathcal{R}_{\cap}$ *in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries.*

Proof sketch. For the honest response, $\gcd(P_1,\dots,P_\ell) = 1$, so polynomials B_i satisfying the stated identity exist. The honest witness satisfies both equalities, and completeness follows from the PLONK-style argument.

For knowledge soundness, the PLONK-style argument establishes the claimed equalities for the committed input and coefficient vectors. Since the input and coefficient commitments are fixed before γ is derived, except with negligible probability, $\prod_{x\in\mathcal{E}^{q_i}_{\mathsf{doc}}}(Z-x) = A_{\mathcal{J}}(Z)P_i(Z)$ for all $i \in [\ell]$ and $\sum_{i=1}^{\ell} B_i(Z)P_i(Z) = 1$. The first identities imply $\mathcal{J} \subseteq \mathcal{E}^{q_i}_{\mathsf{doc}}$ for every i. If an identifier x belongs to every $\mathcal{E}^{q_i}_{\mathsf{doc}}$ but not to $\mathcal{J}$, then $A_{\mathcal{J}}(x) \neq 0$, so $P_i(x) = 0$ for every i. Evaluating the second identity at x gives $0 = 1$, a contradiction. Hence $\mathcal{J} = \bigcap_{i=1}^{\ell} \mathcal{E}^{q_i}_{\mathsf{doc}}$, which is exactly $\mathcal{R}_{\cap}$.

Zero knowledge follows from zero knowledge of the KZG-based PLONK-style argument, together with hiding of VC. □

Efficiency. Constructing the residual polynomials P_i using product trees costs $O(\ell h \log^2 h)_{\mathbb{F}}$. Computing B_i using the extended Euclidean algorithm costs $P_{\mathsf{Euclidean}} = O\big(\ell h \log^2(\ell h)\log\log(\ell h)\big)_{\mathbb{F}}$. Generating the commitments and polynomial proofs costs $O(\ell h \log h)_{\mathbb{F}} + O(\ell h)_{\mathbb{G}}$. Summing these costs gives $O(\ell h \log^2(\ell h)\log\log(\ell h))_{\mathbb{F}} + O(\ell h)_{\mathbb{G}}$. Verification costs $O(\ell \log h + |\mathcal{J}|)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$, with $O(\ell)$ pairing terms accumulated into one final multi-pairing check. The $O(\ell)$ intermediate commitments and subproofs give a total proof size of $O(\ell)_{\mathbb{F},\mathbb{G}}$.

Masking degrees for bounded queries. Consider at most q_{max} audit-assisted queries per commitment, each containing at most c keywords. The verifier reuses the results of Π_{state} and Π_{upd} verification, while query-specific encodings and traces use fresh randomness.

In Π_{state}, Π_{wf} uses at most three off-domain points for each of E_{key} and E_{doc}: one through Π_{supp} and at most two through Π_{gp} on $E_{\mathsf{key}} + (m+1)E_{\mathsf{doc}}$. The LogUp calls in Π_{cnt} and Π_{id} add one point for E_{key} and E_{doc}, respectively, giving at most four points per encoding. The $\mathsf{zkcq}+$ invocations in Π_{mem} and the interval form of Π_{supp} in Π_{upd} do not reveal scalar evaluations of $\widehat{f}_{E_{\mathsf{key}}}$ or $\widehat{f}_{E_{\mathsf{doc}}}$.

For E_{cnt}, each query adds at most c points through Π_{sub}. The Π_{cnt} call in Π_{state} uses at most two points, and each Π_{upd} invocation adds at most two points to either count encoding. An encoding is used either in the initial Π_{state} and the next Π_{upd}, or in two Π_{upd} invocations, first under com^+ and then under com. An evaluation of $\widehat{f}_{E^+_{\mathsf{cnt}}} - \widehat{f}_{E_{\mathsf{cnt}}}$ counts toward the masking budgets of both E^+_{cnt} and E_{cnt}. Thus, $t_{E_{\mathsf{cnt}}} = 4 + cq_{\mathsf{max}}$ is sufficient. The fresh $E^{\mathcal{Q}}_{\mathsf{cnt}}$ encoding uses at most ℓ points in Π_{sub}, ℓ in the PLONK-style quotient argument for c_ρ, and one for padding, so we choose $t_{E^{\mathcal{Q}}_{\mathsf{cnt}}} = 2\ell + 1$.

For identifier and selector encodings, padding uses at most three points, and each product, sum, or adjacent-difference check uses at most two. The combined checks, including the Π_{gp} checks in Π_{upd} on persistent document-identifier encodings, use at most sixteen points per encoding. For these encodings, choose $t_E \geq 16$. Auxiliary encodings have separate budgets, and quotient masks are chosen as described above.

Zero knowledge under the common SRS. We use the randomized encodings of Section A.1, the polynomial-proof simulations of [13, Sections 5.3, 6.1.2, and 7.4–7.6], and the $\mathsf{zkcq}+$ simulator [12].

Simulator. The simulator generates one SRS and retains its trapdoors. It keeps the given database commitments, a record of assigned evaluations, and one random-oracle history. For each adaptive query, it obtains the answer and simulates the new auxiliary commitments and evaluations. Repeated evaluations retain their assigned values. Evaluations of linear combinations are computed from the recorded evaluations of the polynomials being combined. Quotient commitments are sampled before their evaluation challenges; quotient evaluations follow from the checked identities. Openings are simulated using the trapdoor, and lookup proofs and the other checks use the same commitments for each shared vector.

Hybrids. First simulate the opening and lookup messages while retaining the honest shared commitments and evaluations. Then replace the auxiliary commitments and evaluations of the vector encodings by their simulated values, preserving all public dependencies.

Indistinguishability. The first transition follows from the component simulations and independent quotient masking. For the second, condition on the preceding transcript. For each encoding E, the simulation uses at most $1 + A_E + q_{\mathsf{max}}R_E$ evaluations, including $\widehat{f}_E(\tau)$. The masking bound makes the corresponding evaluation map full row rank, giving the same conditional distribution. Induction over the messages and queries proves the claim, apart from component simulation errors, exceptional challenge points, and random-oracle programming failures. Their sum is negligible over polynomially many invocations.

A.3 Cost Analysis

Recall the capacities N, m, h, and write $\ell := |\mathcal{Q}|$ and $a := |\mathcal{J}|$. The vector $E^{\mathcal{Q}}_{\mathsf{cnt}}$ uses a public encoding domain of size $\Theta(\ell)$. With the masking degrees above, $L_{\mathsf{cnt}} = m + cq_{\mathsf{max}} + 5$, while the fresh cardinality encodings use $O(\ell)$ masking coefficients. The bounds exclude one-time setup and reusable preprocessing of public domains and lookup tables. Proof size includes auxiliary commitments but excludes com, $\mathcal{Q}$, and $\mathcal{J}$.

Reusable query-independent cost. The reusable query-independent proofs are π_{wf} and π_{cnt}. The prover cost of Π_{wf} is $O(N \log N)_{\mathbb{F}} + O(N)_{\mathbb{G}}$, while the prover cost of Π_{cnt} is $O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}})_{\mathbb{F}} + O(N + L_{\mathsf{cnt}})_{\mathbb{G}}$. Summing these costs gives

$$P_{\mathsf{query-ind}} = O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}})_{\mathbb{F}} + O(N + L_{\mathsf{cnt}})_{\mathbb{G}}.$$

The $N \log N$ term processes the keyword–identifier table, while the $L_{\mathsf{cnt}} \log L_{\mathsf{cnt}}$ term processes the randomized count polynomial. Combining the two component verifiers gives $V_{\mathsf{query-ind}} = O(\log N + \log m + \log n)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{multi-pairing}}$. Both subproofs have constant size, so the reusable query-independent proof has size $O(1)_{\mathbb{F},\mathbb{G}}$.

Per-query cost. The per-query proof contains $\mathsf{Com}_{\mathsf{aux}}$, π_{sub}, π_{mem}, and $\pi_{\cap}$.

The prover cost of Π_{sub} is $O(L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + \ell \log^2 \ell)_{\mathbb{F}} + O(L_{\mathsf{cnt}} + \ell)_{\mathbb{G}}$. The prover cost of Π_{mem} is $O(\ell h \log h + \ell \log \ell)_{\mathbb{F}} + O(\ell h + \ell \log \ell)_{\mathbb{G}}$, while $\Pi_{\cap}$ costs $O(\ell h \log^2(\ell h) \log\log(\ell h))_{\mathbb{F}} + O(\ell h)_{\mathbb{G}}$. Summing these costs gives the following Base-only bound

$$P_{\mathsf{query}} = O(L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + \ell \log^2 \ell + \ell h \log^2(\ell h) \log\log(\ell h))_{\mathbb{F}} + O(L_{\mathsf{cnt}} + \ell h + \ell \log \ell)_{\mathbb{G}}.$$

The $L_{\mathsf{cnt}} \log L_{\mathsf{cnt}}$ and $\ell \log^2 \ell$ terms come from proving $E^{\mathcal{Q}}_{\mathsf{cnt}}[i] = E_{\mathsf{cnt}}[q_i]$ for every $i \in [\ell]$ with respect to the fixed commitments. The $\ell h \log^2(\ell h) \log\log(\ell h)$ term accounts for the extended Euclidean computation and also covers the residual product trees and the remaining identifier-vector computations. The $\ell \log \ell$ group term includes the count-domain basis construction in the batched cardinality argument.

Combining the three sub-verifiers gives $V_{\mathsf{query}} = O(\ell \log^2 \ell + \ell \log h + a)_{\mathbb{F}} + O(\ell \log \ell + \ell)_{\mathbb{G}}$, together with $O(\ell)$ pairing terms accumulated into one final multi-pairing check. Including the query-dependent commitments, the per-query proof has size $O(\ell)_{\mathbb{F},\mathbb{G}}$.

For a nonempty Cache of public size δ, only the lookup realization inside Π_{mem} changes. Define the additional Cache-dependent prover cost as

$$P_{\mathsf{cache}} = O(\delta \log^2 \delta + \ell(\delta + h) \log^2(\delta + h))_{\mathbb{F}} + O(\ell(\delta + h))_{\mathbb{G}}.$$

The complete per-query prover cost is therefore

$$P^{\mathsf{cache}}_{\mathsf{query}} = P_{\mathsf{query}} + P_{\mathsf{cache}}.$$

The Cache-dependent overhead is described in Remark 8. It does not change the verifier complexity or proof size.

Full standalone cost. A standalone Online proof contains the reusable query-independent proof and the per-query proof. For a Base-only table, summing the two prover costs gives

$$P^{\mathsf{stand}}_{\mathsf{online}} = O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + \ell \log^2 \ell + \ell h \log^2(\ell h) \log\log(\ell h))_{\mathbb{F}} + O(N + L_{\mathsf{cnt}} + \ell h + \ell \log \ell)_{\mathbb{G}}.$$

Combining all five component verifiers gives

$$\begin{aligned} V^{\mathsf{stand}}_{\mathsf{online}} = {} & O(\log N + \log m + \log n + \ell \log^2 \ell + \ell \log h + a)_{\mathbb{F}} \\ & + O(\ell \log \ell + \ell)_{\mathbb{G}} + O(1)_{\mathsf{multi\text{-}pairing}}. \end{aligned}$$

The reusable query-independent proof has constant size, so the total proof size is $O(\ell)_{\mathbb{F},\mathbb{G}}$. For a nonempty Cache of public size δ, the complete standalone prover cost is $P^{\mathsf{stand,cache}}_{\mathsf{online}} = P^{\mathsf{stand}}_{\mathsf{online}} + P_{\mathsf{cache}}$. The Cache does not change the asymptotic standalone verifier cost or proof size. Including the public query and response, total standalone communication is $O(\ell + a)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$.

B Additional Details for Our Audit Phase Protocol

B.1 Instantiating Π_{state} and Π_{upd}

We instantiate Π_{state} and Π_{upd} using the vector commitment of Section A.1 and the auxiliary arguments of Section A.2.1. We first define Π_{id} in Section B.1.1, then construct Π_{state} and Π_{upd} in Sections B.1.2 and B.1.3.

B.1.1 Document-Identifier Argument

We fix public capacities h_D and h_Δ for the document-identifier vectors associated with a complete document set and an appended document set, respectively. These capacities are independent of the corresponding hidden cardinalities. Each capacity is strictly larger than the number of identifiers it must encode, leaving a zero-padding position.

We encode $\mathcal{ID}_{\mathcal{D}}$ using a value vector $I \in \mathbb{F}^{h_D}$ and selector $S_I \in \{0,1\}^{h_D}$, as described in Section A.2.1, and write $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}} := (\mathsf{Com}_I, \mathsf{Com}_{S_I})$. We use the same convention for $\mathcal{ID}_{\mathcal{D}^+}$ and $\mathcal{ID}_\Delta$. This pair implements the identifier-set commitment used by the audit protocol; its opening consists of the openings of both component commitments. The padding and distinctness conditions are enforced by the identifier-set arguments below.

A document-identifier argument proves that the selected entries of a committed identifier vector are exactly the distinct identifiers appearing in the first T positions of a committed document vector.

- **Relation:** Let $E_{\mathsf{doc}} \in \mathbb{F}^N$ and $I, S_I \in \mathbb{F}^{h_D}$. The statement is $x_{\mathsf{id}} := (\mathsf{Com}_{E_{\mathsf{doc}}}, T, \mathsf{Com}_I, \mathsf{Com}_{S_I})$, and the witness is $w_{\mathsf{id}} := (E_{\mathsf{doc}}, I, S_I, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_I, \mathsf{op}_{S_I})$. The relation $\mathcal{R}_{\mathsf{id}}$ requires that (I, S_I) encode exactly the set $\{E_{\mathsf{doc}}[j] \mid j \in [T]\}$.

- **Instantiation:** This instantiation is used together with Π_{wf} on the same $\mathsf{Com}_{E_{\mathsf{doc}}}$ and T. We compose the following arguments. First, Π_{pad} proves that S_I identifies a prefix of I and that every unselected entry is zero. Next, Π_{range} and Π_{dist} prove that the selected entries lie in $[n]$ and are pairwise distinct.

  The prover then constructs a multiplicity vector $\mu \in \mathbb{F}^{h_D}$, where, for each selected position i, $\mu[i] := |\{j \in [T] : E_{\mathsf{doc}}[j] = I[i]\}|$, and $\mu[i] = 0$ at every unselected position. For the honest identifier encoding, every selected multiplicity is positive. Π_{wf} ensures that there are no repeated keyword–identifier pairs, so each multiplicity is at most m. The prover commits to μ, invokes Π_{zpad} to prove that μ is zero outside the selected prefix, and invokes Π_{range} to prove that every selected multiplicity lies in $[1, m]$.

  After these commitments are fixed, the prover and verifier derive $\gamma \xleftarrow{\$} \mathbb{F}\backslash(\{0\} \cup [n])$ via Fiat–Shamir and invoke LogUp [41] to prove

$$\sum_{j=1}^{T} \frac{1}{\gamma - E_{\mathsf{doc}}[j]} = \sum_{i=1}^{h_D} \frac{\mu[i]}{\gamma - I[i]}.$$

  Together, these checks establish that the selected entries of I are exactly the distinct identifiers appearing in the first T entries of E_{doc}.

- **Cost:** The document vector has public length N, while the identifier and multiplicity vectors have public length h_D. The prover cost is $O(N \log N + h_D \log h_D)_{\mathbb{F}} + O(N + h_D)_{\mathbb{G}}$. The verifier cost is $O(\log N + \log h_D + \log n + \log m)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{multi\text{-}pairing}}$, and the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

Lemma 7 (Document-identifier argument)**.** *Assume that* VC *satisfies Definition 1. Assume that* Π_{pad}, Π_{zpad}, Π_{range}, *and* Π_{dist} *satisfy Definition 2 for their respective relations, and that* LogUp *[41] satisfies Definition 2 for* $\mathcal{R}_{\mathsf{logup}}$*. Assume also the guarantees of* Π_{wf} *in Lemma 2. When used with* Π_{wf} *on the same* $\mathsf{Com}_{E_{\mathsf{doc}}}$ *and* T, Π_{id} *is complete and zero knowledge, and its joint execution with* Π_{wf} *provides knowledge soundness for* $\mathcal{R}_{\mathsf{id}}$ *against algebraic adversaries in the SRS and programmable ROM.*

Proof sketch. Completeness follows from the honest identifier encoding: well-formedness places its selected multiplicities in $[1, m]$, and these multiplicities satisfy the LogUp identity.

For knowledge soundness, the padding, range, and distinctness arguments establish that the selected entries of I are distinct valid identifiers. The multiplicity checks ensure that every unselected multiplicity is zero and every selected multiplicity is positive. Soundness of LogUp [41] then implies that the multiset of the first T entries of E_{doc} is exactly the multiset containing each selected identifier $I[i]$ with multiplicity $\mu[i]$. Positivity of the selected multiplicities implies that every selected identifier appears among the first T entries of E_{doc}, while the multiset equality implies that every value in these positions appears among the selected entries of I. Since the selected entries of I are distinct, they encode exactly the set $\{E_{\mathsf{doc}}[j] \mid j \in [T]\}$. Thus, $\mathcal{R}_{\mathsf{id}}$ holds.

Zero knowledge follows from zero knowledge of the component arguments together with hiding of VC. □

B.1.2 Protocol Π_{state}

Recall that the statement is $x_{\mathsf{state}} := (\mathsf{com}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}})$, where $\mathsf{com} = (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}} = (\mathsf{Com}_I, \mathsf{Com}_{S_I})$, and the witness is $w_{\mathsf{state}} := (E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}, I, S_I, \mathsf{op}_{E_{\mathsf{key}}}, \mathsf{op}_{E_{\mathsf{doc}}}, \mathsf{op}_{E_{\mathsf{cnt}}}, \mathsf{op}_I, \mathsf{op}_{S_I})$. The protocol proves that com binds a well-formed document-set representation with correct keyword counts and that $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$ binds exactly its document identifiers.

Prover. The prover proceeds as follows:

- Forms $x_{\mathsf{wf}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, T)$ and invokes Π_{wf} to obtain π_{wf}.
- Forms $x_{\mathsf{cnt}} := (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T)$ and invokes Π_{cnt} to obtain π_{cnt}.
- Forms $x_{\mathsf{id}} := (\mathsf{Com}_{E_{\mathsf{doc}}}, T, \mathsf{Com}_I, \mathsf{Com}_{S_I})$ and invokes Π_{id} to obtain π_{id}.
- Outputs $\pi_{\mathsf{state}} := (\pi_{\mathsf{wf}}, \pi_{\mathsf{cnt}}, \pi_{\mathsf{id}})$.

Verifier. The verifier forms the three sub-statements from x_{state}, verifies π_{wf}, π_{cnt}, and π_{id}, and accepts if and only if all three subproofs accept.

Lemma 8 (Protocol Π_{state})**.** *Assume that* VC *satisfies Definition 1, and that* Π_{wf}, Π_{cnt}, *and* Π_{id} *have the guarantees stated in Lemmas 2, 3 and 7 for the joint execution described above. Then* Π_{state} *described above satisfies Definition 2 for* $\mathcal{R}_{\mathsf{state}}$ *in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries.*

Proof sketch. Completeness follows from completeness of the three component arguments.

For knowledge soundness, Π_{wf}, Π_{cnt}, and Π_{id} establish the well-formedness, count-consistency, and identifier-consistency conditions of $\mathcal{R}_{\mathsf{state}}$, respectively. Their sub-statements use the same commitments to E_{key} and E_{doc}, so subvector binding of VC links all three conditions to the same committed representation. These are exactly the conditions of $\mathcal{R}_{\mathsf{state}}$.

Zero knowledge follows from zero knowledge of Π_{wf}, Π_{cnt}, and Π_{id}, together with hiding of VC. $\square$

Efficiency. The protocol invokes Π_{wf}, Π_{cnt}, and Π_{id} once each. The first two together cost $O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}})_{\mathbb{F}} + O(N + L_{\mathsf{cnt}})_{\mathbb{G}}$, while Π_{id} costs $O(N \log N + h_D \log h_D)_{\mathbb{F}} + O(N + h_D)_{\mathbb{G}}$. Thus, their total prover cost is $O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + h_D \log h_D)_{\mathbb{F}} + O(N + L_{\mathsf{cnt}} + h_D)_{\mathbb{G}}$. Summing the three component verifier costs gives $O(\log N + \log m + \log n + \log h_D)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{multi-pairing}}$. All three component proofs have constant size, so π_{state} has size $O(1)_{\mathbb{F},\mathbb{G}}$.

B.1.3 Protocol Π_{upd}

Recall that the statement is $x_{\mathsf{upd}} := (\mathsf{com}, \mathsf{com}^+, \mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}})$, where

$$\begin{aligned} \mathsf{com} &= (\mathsf{Com}_{E_{\mathsf{key}}}, \mathsf{Com}_{E_{\mathsf{doc}}}, \mathsf{Com}_{E_{\mathsf{cnt}}}, T), \\ \mathsf{com}^+ &= (\mathsf{Com}_{E^+_{\mathsf{key}}}, \mathsf{Com}_{E^+_{\mathsf{doc}}}, \mathsf{Com}_{E^+_{\mathsf{cnt}}}, T^+), \\ \mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}} &= (\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}). \end{aligned}$$

The witness is w_{upd} specified by $\mathcal{R}_{\mathsf{upd}}$. Set $B := T^+ - T$. The protocol proves that com^+ is obtained from com by appending a well-formed document set with fresh identifiers and updating the keyword counts consistently. It is verified relative to an accepted Π_{state} proof for $x_{\mathsf{state}} := (\mathsf{com}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}})$. Let N_Δ be the supported encoding capacity fixed by the public parameters, with $B \le N_\Delta$. The appended keyword and identifier vectors occupy the first B positions and are zero elsewhere.

Prover. The prover proceeds as follows:

- Constructs the appended vectors $E^\Delta_{\mathsf{key}}, E^\Delta_{\mathsf{doc}} \in \mathbb{F}^B$, their count vector $E^\Delta_{\mathsf{cnt}} \in \mathbb{F}^m$, and their identifier set $\mathcal{ID}_\Delta$, as defined in Section 7.1. The appended vectors use this capacity-N_Δ encoding, while their logical length remains B.
- Commits to E^Δ_{key} and E^Δ_{doc}, represents $\mathcal{ID}_\Delta$ by a padded identifier vector I^Δ and selector S^Δ_I, and commits to both. Defines $\mathsf{Com}_{\mathcal{ID}_\Delta} := (\mathsf{Com}_{I^\Delta}, \mathsf{Com}_{S^\Delta_I})$.
- Using linear homomorphism, derives $\mathsf{Com}_{E^\Delta_{\mathsf{cnt}}} := \mathsf{Com}_{E^+_{\mathsf{cnt}}} - \mathsf{Com}_{E_{\mathsf{cnt}}}$ and the corresponding opening. Defines $\mathsf{com}_\Delta := (\mathsf{Com}_{E^\Delta_{\mathsf{key}}}, \mathsf{Com}_{E^\Delta_{\mathsf{doc}}}, \mathsf{Com}_{E^\Delta_{\mathsf{cnt}}}, B)$ and $x^\Delta_{\mathsf{state}} := (\mathsf{com}_\Delta, \mathsf{Com}_{\mathcal{ID}_\Delta})$. Invokes Π_{state} on $x^\Delta_{\mathsf{state}}$ to obtain $\pi^\Delta_{\mathsf{state}}$.
- Invokes Π_{pad} and Π_{dist} on $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$, and invokes Π_{gp} on $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$, $\mathsf{Com}_{\mathcal{ID}_\Delta}$, and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ to prove $\mathcal{ID}_{\mathcal{D}^+} \equiv \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_\Delta$. Denote these proofs collectively by π_{union}.
- For each $V \in \{E_{\mathsf{key}}, E_{\mathsf{doc}}\}$, let V^+ denote the corresponding vector committed by com^+ and V^Δ the corresponding appended vector. Using linear homomorphism of VC, derives $\mathsf{Com}_{V^+-V} := \mathsf{Com}_{V^+} - \mathsf{Com}_V$. Invokes the interval form of Π_{supp} on $(\mathsf{Com}_{V^+-V}, \mathsf{Com}_{V^\Delta}, T, B, N)$ to prove $V^+ - V = 0^T \| V^\Delta \| 0^{N-T^+}$. Denotes the resulting proofs by $\pi^{\mathsf{key},\Delta}_{\mathsf{supp}}$ and $\pi^{\mathsf{doc},\Delta}_{\mathsf{supp}}$.
- Outputs $\pi_{\mathsf{upd}} := \big(\mathsf{Com}_{E^\Delta_{\mathsf{key}}}, \mathsf{Com}_{E^\Delta_{\mathsf{doc}}}, \mathsf{Com}_{\mathcal{ID}_\Delta}, \pi^\Delta_{\mathsf{state}}, \pi_{\mathsf{union}}, \pi^{\mathsf{key},\Delta}_{\mathsf{supp}}, \pi^{\mathsf{doc},\Delta}_{\mathsf{supp}}\big)$.

Verifier. The verifier derives $\mathsf{Com}_{E^\Delta_{\mathsf{cnt}}} := \mathsf{Com}_{E^+_{\mathsf{cnt}}} - \mathsf{Com}_{E_{\mathsf{cnt}}}$ and, for each $V \in \{E_{\mathsf{key}}, E_{\mathsf{doc}}\}$, $\mathsf{Com}_{V^+-V} := \mathsf{Com}_{V^+} - \mathsf{Com}_V$. It then reconstructs the sub-statements above, verifies $\pi^\Delta_{\mathsf{state}}$, π_{union}, and the two interval-support proofs, and accepts if and only if all subproofs accept.

Lemma 9 (Protocol Π_{upd})**.** *Assume that* VC *satisfies Definition 1 and that* $x_{\mathsf{state}} = (\mathsf{com}, \mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}})$ *has been certified by an accepted* Π_{state} *proof. Assume further that* Π_{state} *satisfies Definition 2 for* $\mathcal{R}_{\mathsf{state}}$*, and that* Π_{pad}*,* Π_{dist}*,* Π_{gp}*, and the interval form of* Π_{supp} *satisfy the completeness, knowledge-soundness, and zero-knowledge requirements of Definition 2 for their respective relations. Then* Π_{upd} *is complete and zero knowledge under these assumptions. Its joint execution with* Π_{state} *provides knowledge soundness for* $\mathcal{R}_{\mathsf{upd}}$ *against algebraic adversaries in the SRS and programmable ROM.*

Proof sketch. Completeness follows from the honest append construction.

For knowledge soundness, $\pi^{\Delta}_{\mathsf{state}}$ establishes that the appended keyword–identifier rows form a well-formed document set, that $E^{\Delta}_{\mathsf{cnt}}$ is their count vector, and that $\mathcal{ID}_{\Delta}$ is exactly the identifier set appearing in $E^{\Delta}_{\mathsf{doc}}$. This establishes the well-formedness condition of $\mathcal{R}_{\mathsf{upd}}$.

The padding, distinctness, and grand-product arguments in π_{union} establish $\mathcal{ID}_{\mathcal{D}^+} = \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$, and hence the fresh-identifier condition.

For each $V \in \{E_{\mathsf{key}}, E_{\mathsf{doc}}\}$, soundness of the interval form of Π_{supp} gives $V^+ - V = 0^T \| V^{\Delta} \| 0^{N-T^+}$. The accepted Π_{state} proof for the current state establishes $V[j] = 0$ for every $T < j \le N$. Consequently,

$$V^+[j] = \begin{cases} V[j], & 1 \le j \le T, \\ V^{\Delta}[j-T], & T < j \le T^+, \\ 0, & T^+ < j \le N, \end{cases}$$

which is exactly the append-only consistency condition of $\mathcal{R}_{\mathsf{upd}}$.

Finally, $\pi^{\Delta}_{\mathsf{state}}$ certifies the count vector $E^{\Delta}_{\mathsf{cnt}}$ using the derived commitment $\mathsf{Com}_{E^{\Delta}_{\mathsf{cnt}}} = \mathsf{Com}_{E^{+}_{\mathsf{cnt}}} - \mathsf{Com}_{E_{\mathsf{cnt}}}$. Linear homomorphism and subvector binding link this vector to $E^{+}_{\mathsf{cnt}} - E_{\mathsf{cnt}}$. Hence, $E^{+}_{\mathsf{cnt}} = E_{\mathsf{cnt}} + E^{\Delta}_{\mathsf{cnt}}$, which is the count-consistency condition of $\mathcal{R}_{\mathsf{upd}}$.

Thus, all conditions of $\mathcal{R}_{\mathsf{upd}}$ hold. Zero knowledge follows from zero knowledge of the component arguments together with hiding of VC. □

Efficiency. Recall that $B := T^+ - T$. The prover invokes Π_{state} once for the appended rows, with capacities $N_{\Delta} \ge B$, m, and h_{Δ}. This costs $O(N_{\Delta} \log N_{\Delta} + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + h_{\Delta} \log h_{\Delta})_{\mathbb{F}} + O(N_{\Delta} + L_{\mathsf{cnt}} + h_{\Delta})_{\mathbb{G}}$. The prover then invokes Π_{pad}, Π_{dist}, and Π_{gp} on the identifier-set commitments, adding $O(h_D \log h_D + h_{\Delta} \log h_{\Delta})_{\mathbb{F}} + O(h_D + h_{\Delta})_{\mathbb{G}}$. Finally, each of the two Π_{supp} invocations costs $O(N_{\Delta} \log N_{\Delta} + B)_{\mathbb{F}} + O(N_{\Delta} \log N_{\Delta} + B)_{\mathbb{G}}$. The total prover cost is $O(h_D \log h_D + h_{\Delta} \log h_{\Delta} + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + N_{\Delta} \log N_{\Delta} + B)_{\mathbb{F}} + O(h_D + h_{\Delta} + L_{\mathsf{cnt}} + N_{\Delta} \log N_{\Delta} + B)_{\mathbb{G}}$.

The two Π_{supp} invocations contribute $O(B)_{\mathbb{F}} + O(B)_{\mathbb{G}}$ verification cost; the remaining subproofs contribute the logarithmic terms. Thus the verifier cost is $O(B + \log N_{\Delta} + \log m + \log n + \log h_D + \log h_{\Delta})_{\mathbb{F}} + O(B)_{\mathbb{G}} + O(1)_{\mathsf{multi\text{-}pairing}}$. Only the interval-support proofs contribute a B-dependent proof length, so the total proof size is $O(B)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

Zero knowledge under the common SRS. The simulator of Section A.2 also handles the honest Audit experiment. It retains com and com^+ and derives $\mathsf{Com}_{V^+} - \mathsf{Com}_V$ for each $V \in \{E_{\mathsf{key}}, E_{\mathsf{doc}}, E_{\mathsf{cnt}}\}$. An evaluation of a difference is computed from the recorded evaluations of the two polynomials and counted toward both masking budgets. Auxiliary commitments and proofs use these shared inputs and the same random oracle.

Honest updates use fresh randomness. The masking bounds include the Π_{state} and Π_{upd} invocations involving each encoding. After an Audit, the simulator continues with the same evaluation records and random-oracle history for at most q_{max} adaptive queries under com^+, which may depend on the Audit transcript. The hybrids of Section A.2 therefore apply to the combined view.

B.2 Cost Analysis

Recall that N is the capacity of the keyword and identifier vectors, $m := |\mathcal{K}|$, and $B := T^+ - T$ is the public number of keyword–identifier pairs appended by one update. Let N_Δ denote the public encoding capacity of the appended pair vectors. Let h_D be the public capacity of the identifier encoding associated with a complete document set, and let h_Δ be the public capacity of the identifier encoding associated with the appended rows. The bounds below exclude one-time setup and reusable preprocessing of evaluation domains. Proof size includes all auxiliary commitments, but excludes the public commitments com and com^+.

Initial Π_{state} cost. Π_{state} invokes Π_{wf}, Π_{cnt}, and Π_{id}. The prover cost of Π_{wf} is $O(N \log N)_{\mathbb{F}} + O(N)_{\mathbb{G}}$, the prover cost of Π_{cnt} is $O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}})_{\mathbb{F}} + O(N + L_{\mathsf{cnt}})_{\mathbb{G}}$, and the prover cost of Π_{id} is $O(N \log N + h_D \log h_D)_{\mathbb{F}} + O(N + h_D)_{\mathbb{G}}$. Summing these costs gives

$$\begin{aligned} P_{\mathsf{pub}} = O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + h_D \log h_D)_{\mathbb{F}} \\ + O(N + L_{\mathsf{cnt}} + h_D)_{\mathbb{G}}. \end{aligned}$$

The three field-operation terms account for the keyword–identifier table, the randomized count polynomial, and the document-identifier encoding, respectively.

Combining the three component verifiers gives $V_{\mathsf{pub}} = O(\log N + \log m + \log n + \log h_D)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\mathsf{multi-pairing}}$.

The identifier-set commitment and all three component proofs have constant size. Therefore, the proof size is $O(1)_{\mathbb{F},\mathbb{G}}$.

Per-update cost. π_{upd} contains $\pi^{\Delta}_{\mathsf{state}}$, π_{union}, and the two interval-support proofs linking the appended keyword and identifier vectors to com and com^+.

The Π_{state} invocation on $x^{\Delta}_{\mathsf{state}}$ processes the capacity-N_Δ encodings of the B appended pairs, the count polynomial with at most L_{cnt} coefficients, and the appended identifier encoding of capacity h_Δ. Generating π_{union} additionally processes the identifier encodings of capacities h_D and h_Δ. Finally, the interval representation checks are linear in B. The prover also performs domain reconstruction and basis derivation, contributing the $N_\Delta \log N_\Delta$ field and group terms. Summing these component costs gives

$$\begin{aligned} P_{\mathsf{upd}}(B) = O(h_D \log h_D + h_\Delta \log h_\Delta + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + N_\Delta \log N_\Delta + B)_{\mathbb{F}} \\ + O(h_D + h_\Delta + L_{\mathsf{cnt}} + N_\Delta \log N_\Delta + B)_{\mathbb{G}}. \end{aligned}$$

Combining the component verifiers gives $V_{\mathsf{upd}}(B) = O(B + \log N_\Delta + \log m + \log n + \log h_D + \log h_\Delta)_{\mathbb{F}} + O(B)_{\mathbb{G}} + O(1)_{\mathsf{multi-pairing}}$. The linear terms in B come from the two interval-support checks, while the logarithmic terms come from verifying $\pi^{\Delta}_{\mathsf{state}}$ and π_{union}.

The proof contains a constant number of compact commitments and constant-size component proofs, together with the two interval-support proofs. Hence, the proof size is $O(B)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

Full standalone Audit cost. A standalone Audit proof contains $\mathsf{Com}^{\mathsf{audit}}_{\mathsf{aux}}$, π_{state}, and π_{upd}. Therefore, its prover cost is the sum of P_{pub} and $P_{\mathsf{upd}}(B)$:

$$\begin{aligned} P^{\mathsf{stand}}_{\mathsf{audit}} = O(N \log N + L_{\mathsf{cnt}} \log L_{\mathsf{cnt}} + h_D \log h_D + h_\Delta \log h_\Delta + N_\Delta \log N_\Delta + B)_{\mathbb{F}} \\ + O(N + L_{\mathsf{cnt}} + h_D + h_\Delta + N_\Delta \log N_\Delta + B)_{\mathbb{G}}. \end{aligned}$$

Combining the two component verifiers gives $V^{\mathsf{stand}}_{\mathsf{audit}} = O(B + \log N + \log N_\Delta + \log m + \log n + \log h_D + \log h_\Delta)_{\mathbb{F}} + O(B)_{\mathbb{G}} + O(1)_{\mathsf{multi-pairing}}$.

The two identifier-set commitments and π_{state} contribute only a constant number of group elements. The update proof contributes $O(B)_{\mathbb{F}} + O(1)_{\mathbb{G}}$. Therefore, the proof size is $O(B)_{\mathbb{F}} + O(1)_{\mathbb{G}}$.

Sequential Audit cost. Consider a sequence of s accepted updates with public batch sizes $B_1, \ldots, B_s$. The initial π_{state} and $\mathsf{Com}_{\mathcal{ID}_{\mathcal{D}}}$ are generated and verified once. Each later update contains one fresh identifier-set commitment and one fresh update proof.

The total prover cost of the sequence is

$$P^{\mathsf{seq}}_{\mathsf{audit}} = P_{\mathsf{pub}} + \sum_{r=1}^{s} P_{\mathsf{upd}}(B_r).$$

The total verifier cost is $V^{\mathsf{seq}}_{\mathsf{audit}} = V_{\mathsf{pub}} + \sum_{r=1}^{s} V_{\mathsf{upd}}(B_r)$. Their total proof size is $O(\sum_{r=1}^{s} B_r)_{\mathbb{F}} + O(s)_{\mathbb{G}}$.

The $O(s)_{\mathbb{G}}$ term comes from a constant number of fresh commitments and constant-size proof components for every accepted update.

Averaged over the s updates, the prover and verifier costs are

$$\overline{P}^{\mathsf{seq}}_{\mathsf{audit}} = \frac{P_{\mathsf{pub}}}{s} + \frac{1}{s}\sum_{r=1}^{s} P_{\mathsf{upd}}(B_r)$$

and

$$\overline{V}^{\mathsf{seq}}_{\mathsf{audit}} = \frac{V_{\mathsf{pub}}}{s} + \frac{1}{s}\sum_{r=1}^{s} V_{\mathsf{upd}}(B_r).$$

The average proof size per accepted update is

$$O\left(\frac{1}{s}\sum_{r=1}^{s} B_r\right)_{\mathbb{F}} + O(1)_{\mathbb{G}}.$$

Lookup maintenance and rebasing. Rebasing is a preprocessing operation rather than an Audit proof. Its main cost is materializing the complete current keyword and identifier vectors and rebuilding the table-dependent lookup preprocessing for the new Base. Let h be the public capacity of each query-specific padded identifier vector. This costs $M_{\mathsf{rebase}} = O(N \log N + mh \log h)_{\mathbb{F}} + O(N \log N + mh)_{\mathbb{G}}$.

Suppose that the system rebases after every s accepted updates and, for simplicity, that every update appends B keyword–identifier pairs. Immediately before rebasing, the Cache contains $\delta = Bs$ pairs.

Assuming a constant number of queries per update, with ℓ and h fixed, the Cache-dependent lookup work averaged over one rebase interval is $O(Bs \log^2(Bs))_{\mathbb{F}}$. The Base refresh contributes the amortized cost $O(\frac{N \log N + mh \log h}{s})_{\mathbb{F}}$ per update. Hence the amortized lookup-maintenance cost is $O(Bs \log^2(Bs) + \frac{N \log N + mh \log h}{s})_{\mathbb{F}}$. The corresponding group cost is $O(Bs + \frac{N \log N + mh}{s})_{\mathbb{G}}$.

When $N \log N$ dominates $mh \log h$ and $B \log N \leq N$, choosing $s = \Theta(\sqrt{\frac{N}{B \log N}})$ balances the Cache-dependent and Base-refresh terms. The resulting amortized lookup-maintenance cost is $O(\sqrt{BN} \log^{3/2} N)_{\mathbb{F}}$.