

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.

Contents

1	Introduction	3
1.1	Our Contributions	4
2	Technical Overview	5
3	Related Works	9
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	18
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	38
8.1	Implementation	38
8.2	Evaluation Results	39
A	Additional Details for Our Online Query Phase Protocol	47
A.1	Instantiating Vector Commitments	47
A.2	Instantiating Π_{wf} , Π_{cnt} , Π_{sub} , Π_{mem} and $\Pi_{\cap}$	48
A.3	Cost Analysis	59
B	Additional Details for Our Audit Phase Protocol	60
B.1	Instantiating Π_{state} and Π_{upd}	60
B.2	Cost Analysis	64

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 are the most common external group to 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 [53]. 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 [26].

The freedom of information ecosystem often fails in predictable ways. In addition to long processing delays and “normal” bureaucratic challenges, government agencies frequently 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 [52]. 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” [52]. While it is impossible to know if an omission is accidental or intentional, these omissions radically undermine the goals of freedom of information laws.

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 [61]. 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, 59, 60]. 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 [52]. Although 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 video of a lecture by Rear Adm. Grace Hopper, despite the existence of the footage being well-documented [44].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]

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

occur outside the public’s view” [52].

Towards more trustworthy freedom of information responses. Rectifying this situation 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—or no 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 select a subset of the “true” witness, and still satisfy the relation. Instantiating this template in a performant way, however, is challenging, particularly when one considers that any proposed solution must scale to a large number of documents and be runnable by agencies with small budgets (e.g., local police departments). We also require immense flexibility, as agencies each have their own way of digitizing and storing records.

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. In doing so, we make the following contributions:

- **Defining Sunshine Systems.** We begin by giving formal definitions for Sunshine Systems. While these systems are closely related to existing systems like key transparency and verifiable SQL, we observe that there are subtle differences, including the requirement that it should be impossible for a prover to *under-report* or *over-report* documents.
- **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. Each document is 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

(SRS), and we prove its security in the algebraic group model and the random oracle model.

- **Implementation and evaluation.** Finally, we implement and evaluate our construction on the Enron email dataset [39]. Our results demonstrate that even on commodity hardware, producing query results requires at most several minutes, and preprocessing can be completed within several hours—a task that can be run infrequently and overnight.

2 Technical Overview

Today, a freedom of information request is serviced almost entirely on the agency’s word—the very agency whose misdeeds may be at stake within the records. A requester submits a query; staff inside the agency search its own record systems (email archives, case-management databases, shared drives, or physical filing cabinets), determine which records are responsive, and return them. The response may also acknowledge the existence of relevant documents that are *not* being returned, possibly because making the documents public would constitute a security risk or because the documents are privileged for some reason. The requester sees only this output, but has no view into what was searched, what the agency actually holds, or what was set aside. An incomplete response is, in practice, indistinguishable from a complete one.

It is important to acknowledge that there are, of course, many other barriers to an effective information request ecosystem that are beyond the scope of our work. For example, fulfilling information requests often takes an extremely long time (multiple years). Alternatively, an agency might claim that certain documents are not accessible by information request laws, potentially due to national security concerns. In these cases, where the agency either fails to complete the information request process or acknowledges that documents have been withheld, requesters can appeal to the courts to intervene on their behalf. Our primary concern is cases in which the suppression of information is undetectable, leaving requesters without an indication that appealing to a court might be a fruitful option. Importantly, appealing to a court is an expensive process, often costing tens of thousands of dollars, making appealing every response an ineffective solution [52].

This points to a narrow but socially impactful intervention opportunity. We cannot force an agency to hand over a document and technology cannot repair genuine administrative failures (e.g., whether a real-world event was faithfully recorded, how redactions are applied, or how paper records are digitized). But the step at the center of every request that considers digital documents—searching a fixed body of records and returning the matches—is fundamentally *computation*, and computation can be made verifiable. The result is that the omission of documents becomes detectable, which allows the requester to seek court intervention, just as the other failure modes allow today.

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

- *Detects omitted documents:* As previously discussed, 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* additional, unrelated documents to those returned. This property is in response to critical scholarship on transparency that highlights the way in which institutions often disclose *too many* documents in an attempt to consume investigators’ resources [2]—an offline analog of a denial-of-service attack. While information request systems are not the primary place where this problem emerges, it naturally applies to this setting. 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 of the documents produced by government agencies may be sensitive and maybe cannot be immediately released to the public, but may later be subject to sunshine laws. As such, the verification system cannot proactively reveal anything unnecessary about documents, either as documents are created or when answering queries.
- *Interoperability*: Government agencies are, generally speaking, not required to store documents in a particular format or using a particular technology. As such, we require a system design that is interoperable with a variety of storage schemes. Looking ahead, we do this by making our systems independent of the documents themselves, reasoning over document identifiers computed as salted document hashes, rather than the documents themselves.
- *Low concrete costs*: We note that local agencies may not have a tremendous amount of computational power available to them, nor the budgetary resources to rent cloud resources in response to information requests. As such, we need a system design which can take advantage of existing, consumer-grade computers and can 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. Looking ahead, we do this by taking advantage of “offline” pre-processing time, which can be run overnight when government offices are closed, resulting in concretely efficient online prover times.

Our aim is to accomplish these design goals within the context of a very specific class of adversarial behavior. Namely, we consider an adversary that is semi-honest while initially producing documents, but then may later be malicious when attempting to answer information requests. This captures a natural, real-world behavior: a government actor who 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, 52, 59, 60]. Alternatively, this captures an automated system that processes documents, e.g., a mail server that accumulates all outgoing and incoming emails, including documents that government actors might be later tempted to suppress. In practice, this is the best adversarial model we can hope for, as there is no reasonable way to require a malicious actor to properly process documents—especially when some of them may initially be physical documents.

Problems with the strawman construction. When viewed through a cryptographic lens, this problem appears to be a direct application of the *commit and prove* paradigm. The government agency can simply produce a binding commitment to their documents, and then produce a proof that the returned subset of documents matches the query criteria. Indeed, generic commit and prove zero-knowledge proofs do offer a feasibility result, but this back-of-the-napkin sketch misses several important caveats that complicate instantiating this template in practice:

- *Linear online prover complexity*: Recall that our key goal is to ensure that no relevant documents are omitted from the response. Within the commit and prove paradigm, this would require that the prover iterate through all possible documents, showing that all documents that it did not return are not relevant to the query; this work must be repeated in response to *each* query. As the set of documents grows, this will become prohibitive on consumer-grade hardware.
- *Documents are added incrementally*: The naïve commit and prove approach fails to provide meaningful security in our threat model, as we are considering a malicious prover who might commit to modified data in response to a query. Instead, we need an approach that requires

the agency to incrementally process—and commit to—documents as they are created, before the query is made; recall that we consider a model in which the adversary is semi-honest during document creation. We note, however, that simply committing to each document separately results in a highly inefficient construction, as simply reading the statement would be linear in the number of documents.

A template for efficient Sunshine Systems. Looking at these two failures, it becomes clear that we need a different set of cryptographic techniques. Luckily, there are two naturally applicable families of techniques:

- (1) *Improving online performance with lookup arguments:* Lookup arguments are a class of zero-knowledge proofs that allow a prover to convince a verifier that a (set of) committed value(s) are entries in a larger (committed) table. The key advantage of lookup arguments is that they are succinct and can leverage *preprocessing* such that online prover complexity is sublinear in the table size. In the context of sunshine systems, this promises to allow the prover to reuse precomputation across queries and defer this “expensive” computation to overnight downtime. Unfortunately, lookup arguments provide no natural way to ensure that the subset of the table returned contains “all” the rows in which a verifier is interested. In other words, we require additional techniques to make lookup arguments compatible with our design goals.
- (2) *Aggregating documents using append-only commitments:* Rather than having the prover produce a fresh commitment in response to each query (which is insecure) or produce a per-document commitment (which is inefficient), we can use a succinct vector commitment that supports extensions to aggregate the documents as they are produced, e.g., using techniques similar to those employed in certificate transparency, key transparency, and other similar systems [1, 20, 31, 32, 40, 45]. Namely, we can ask agencies to commit to documents as they are created and aggregate them in an append-only fashion into the succinct commitment. Just as is currently done in key transparency, a third-party auditor can be leveraged to ensure that each update to the commitment is well-formed and append-only, allowing verifiers to consider only the latest version of the commitment. While finding auditors in key transparency has proved difficult in practice, government oversight offices and civil society are extraordinarily well-positioned to perform this role for sunshine systems—provided the auditing verification procedure is cheap.

Combining these two techniques provides a workable template: documents are aggregated into a succinct commitment as they are generated, and these updates are verified by an external auditor. At query time, a lookup argument is used to retrieve the relevant subset. In Section 5 we provide formal definitions for this template, separating out the online query phase from the auditing process. Operationalizing this template, however, requires care.

The *first challenge* is that lookup arguments only partially solve our efficiency problem, as they provide no native safeguards to prevent a prover from dropping rows of the table—the very threat against which sunshine systems are intended to protect. Looking at this more carefully, it is somewhat inherent that generic lookup arguments cannot guarantee the completeness of responses, as completeness is a function of the *query language*; there must be something in the commitment which binds a prover to returning the correct subset. Moreover, the query language itself must be something that the prover can efficiently process, as evaluating an arbitrary predicate over every document is inherently linear in the dataset and preprocessing this computation for every document implies a very limited query language. 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 that the preprocessing used in lookup arguments is not amenable to incremental computation. Each time a single element is added to the table, the entire precomputation must be repeated. Thankfully, existing work by Dutta et al. [17] has designed an “updatable” lookup argument construction in which the cost of recomputing the preprocessing can be gracefully amortized. In their approach, the prover maintains a base table with the majority of the data, and incrementally added entries are stored in a smaller cache. The prover can then generate proofs that reason over both the base and cache. These proofs, however, become more expensive as the cache grows, unlike traditional lookup arguments. To manage this cost, the prover periodically merges the cache into the base, which requires performing a fresh round of computation, but collapses the prover’s time back down to that of a traditional lookup argument. Notably, the precomputation is only recomputed periodically, not in response to each incrementally added document.

Finally, we note that there are minor mechanical inconsistencies we need to overcome in order to merge succinct, append-only commitment schemes with lookup arguments. Specifically, modern, state-of-the-art lookup arguments use polynomial commitment schemes to commit to the table, but most state-of-the-art append-only commitment schemes that support efficient auditing are Merkle tree variants. As such, we need to design techniques to attach these two primitives to each other gracefully.

Instantiating the template for keyword queries. Keywords are an expressive query language that captures a fair subset of natural queries that might be made in a sunshine system (e.g., “*give me all police records associated with keywords SEXUAL MISCONDUCT, CAPTIAN, and REPRIMAND.*”). Formally, we will write that each document, identified with an identifier id_i (computed as the salted hash of the document), is associated with a set of keywords $\mathcal{W}_i$. Keywords can be associated with documents as the document is produced, either manually, using a pre-set dictionary, or using some machine-learning-powered system.

The natural starting point for keyword queries is an *index*, in which each keyword k is associated with the full list of relevant documents $\text{id}_1, \text{id}_2, \dots$. This approach fits naturally with lookup arguments, as the index can trivially be stored in tabular form, and a single lookup to the entry k will retrieve the *entire* list of relevant documents. Unfortunately, making this approach compatible with incremental updates is challenging, as each update requires *changing* existing entries in the table, providing an opportunity for the prover to drop existing records or requiring complex bookkeeping to ensure the append-only nature of the log. On the other hand, approaches that are more amenable to updatability, e.g., using each document id as the label in the table, do not provide a natural way to ensure that all the relevant documents are returned.

We observe that some state across the full set of documents is necessary in order to prevent the prover from under-reporting, but that this state can be very lightweight: simply keeping counts of the number of entries for each keyword. Namely, we have the prover incrementally add tuples of the form (k, id) to the (unordered) table, in addition to keeping a running count of how many times each keyword k appears in the table. This approach prevents the prover from suppressing documents, as omitting documents will result in a visible under-count.

All that remains to do, therefore, is to ensure that updates to the table are well-formed. The public commitment to the document set consists of a vector commitment of (k, id) pairs² and a vector commitment to the keyword counts E_{cnt} . We leverage homomorphic vector commitments to make performing updates simple. Namely, each time the prover wants to perform an update, they generate a commitment to the new pairs, but shift their indices in the vector such that they appear directly after the existing entries. When homomorphically combined, the result is the new table.

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

Similarly, the prover computes a commitment to the counts of the newly added entries, proves that this vector is well-formed with respect to the new entries, and then homomorphically sums the counts. Importantly, subsequent audits avoid recomputing a proof that reasons over the whole state. They certify only the newly added elements and their consistency with the prior state.

We note that wiring these pieces together correctly requires some additional machinery and bookkeeping. We discuss the details of responding to queries in Section 6 and the details of the audits in Section 7.

Adding keyword intersections. Finally, we observe that operationalizing keyword queries requires enabling *conjunctions* of keywords, i.e., retrieving the intersection of the documents that contain each keyword in the query. Without conjunctions, information requests about even reasonably specific documents will inevitably return many documents that are not pertinent. For example, a query for *all documents about Officer X and sexual misconduct* would also return off-duty logs about Officer X and unrelated sexual misconduct reports about other officers. To expose this functionality, we additionally prove that this intersection is computed properly in zero-knowledge; details of our approach can be found in Section 6.

A final thought. We emphasize that our goal in this work is to scope this interesting social problem as a technical, cryptographic one and provide a set of formal definitions. In doing so, we identify a combination of properties that, to our knowledge, have not previously been studied: the combination of (1) lookup arguments that (2) gracefully handle updates and (3) are guaranteed to return all of the relevant table entries. While we are able to provide these properties by carefully weaving together known techniques, our result serves as a feasibility result that promises to be high impact—and hopefully inspire further work.

3 Related Works

Cryptography and governance. Several other works study the promise of using zero-knowledge as a tool for accountable governance. Goldwasser and Park [30] and Frankle et al. [25] 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. [33] and Fetzer et al. [22] 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. [21] 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” [63] considers adversarial data publishers who may want to suppress data provided from trusted data owners. Devanbu et al. [15] and Martel et al. [47] provide solutions to this problem based on Merkle trees and Papamanthou et al. [55] build succinct versions based on pairings. 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., [42, 54]. A line of work on verifiable SQL queries [35, 43, 67–69] has extended these approaches by supporting generic SQL operations and making the responses zero-knowledge. While these approaches provide our desired functionality, they provide the wrong performance profile,

as they do not naturally interoperate with proof systems that support preprocessing like lookup arguments. Additionally, others have studied guaranteeing that no query results are omitted using trusted execution environments, e.g., [5]. 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 [18, 28, 36, 57, 65, 66] use vector commitments based on univariate polynomial commitments, while Setty et al. [58] extend this approach to multilinear polynomial commitments. Subsequent works support richer lookups, including segment lookup [14] and matrix lookup [16]. A key drawback is that table updates typically require recomputing the entire preprocessing. Dutta et al. [17] 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, 13, 20, 31, 32, 37, 40, 41, 45, 46, 49, 64]. 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. Pappas et al. [56] introduce *Rogue*, an updatable matrix lookup argument that improves the worst-case update costs of Dutta et al. [17] by avoiding periodic full-table reprocessing, and use it to build *RogueDB*, a verifiable SQL system with authenticated indexes. One could potentially replace the updatable lookup argument of Dutta et al. [17] in our construction with *Rogue*. However, this does not appear to offer a clear advantage for two reasons. First, *Rogue* inherits the same limitations as other lookup arguments, including Dutta et al. [17], discussed in Section 2; in particular, we would still require our mechanisms for complete conjunctive keyword retrieval and append-only auditing with fresh document identifiers. Second, *Rogue* does not achieve the zero-knowledge guarantee for private-table lookups required by our system. We therefore combine the update framework of Dutta et al. [17] with the zero-knowledge lookup argument *zkcq+* [12], rather than using *Rogue*.

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 $[r] := \{1, \dots, r\}$, 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 $\text{VC} = (\text{VC.Setup}, \text{VC.Commit}, \text{VC.Open}, \text{VC.Verify})$ defined as follows:*

- $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$: takes as input the security parameter λ and an upper bound d on the vector length, and outputs public parameters pp_{VC} .
- $(\text{Com}_V, \text{op}_V) \leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, V)$: takes as input pp_{VC} and a vector $V \in \mathbb{F}^r$ of a public length $r \leq d$, and outputs a commitment Com_V and the opening information op_V .

- $(V|_{\mathcal{S}}, \pi_{\mathcal{S}}) \leftarrow \text{VC.Open}(\text{pp}_{\text{VC}}, \text{Com}_V, \mathcal{S}, \text{op}_V)$: takes as input pp_{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 $\pi_{\mathcal{S}}$.
- $b \leftarrow \text{VC.Verify}(\text{pp}_{\text{VC}}, \text{Com}_V, \mathcal{S}, V|_{\mathcal{S}}, \pi_{\mathcal{S}})$: takes as input 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 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 $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$. For every public length $r \leq d$, every $V \in \mathbb{F}^r$ and every $\mathcal{S} \subseteq [r]$,

$$\Pr \left[\text{VC.Verify}(\text{pp}_{\text{VC}}, \text{Com}_V, \mathcal{S}, V|_{\mathcal{S}}, \pi_{\mathcal{S}}) = 1 \mid \begin{array}{l} (\text{Com}_V, \text{op}_V) \leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, V), \\ (V|_{\mathcal{S}}, \pi_{\mathcal{S}}) \leftarrow \text{VC.Open}(\text{pp}_{\text{VC}}, \text{Com}_V, \mathcal{S}, \text{op}_V) \end{array} \right] = 1.$$

- **Subvector Binding:** Let $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$. For every public length $r \leq d$ 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,

$$\Pr \left[\begin{array}{l} V_0 \neq V_1 \\ \wedge \text{VC.Verify}(\text{pp}_{\text{VC}}, \text{Com}^*, \mathcal{S}, V_0, \pi_0) = 1 \\ \wedge \text{VC.Verify}(\text{pp}_{\text{VC}}, \text{Com}^*, \mathcal{S}, V_1, \pi_1) = 1 \end{array} \mid \begin{array}{l} (\text{Com}^*, \mathcal{S}, V_0, \pi_0, V_1, \pi_1) \\ \leftarrow \text{Adv}(\text{pp}_{\text{VC}}) \end{array} \right] \leq \text{negl}(\lambda),$$

- **Hiding:** Let $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$. For every supported public length $r \leq d$, 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 $b \in \{0, 1\}$, $(\text{Com}_b, \text{op}_b) \leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, V_b)$ and $(V_b|_{\mathcal{S}}, \pi_b^{\mathcal{S}}) \leftarrow \text{VC.Open}(\text{pp}_{\text{VC}}, \text{Com}_b, \mathcal{S}, \text{op}_b)$. Then

$$\{(\text{pp}_{\text{VC}}, \text{Com}_0, \mathcal{S}, V_0|_{\mathcal{S}}, \pi_{\mathcal{S}}^0)\} \approx_c \{(\text{pp}_{\text{VC}}, \text{Com}_1, \mathcal{S}, V_1|_{\mathcal{S}}, \pi_{\mathcal{S}}^1)\}.$$

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

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. Our construction uses an SRS induced by the hiding-KZG polynomial commitment scheme [38]. In our concrete instantiation, the public parameters of VC and all component arguments are derived from this same SRS.

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 $\mathcal{H} : \{0, 1\}^* \rightarrow \mathcal{Y}$. 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 [50], 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 [23] 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) [27]. 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. The AGM is used in the knowledge-soundness analysis of the arguments.

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 $\Pi = (\Pi.\text{Setup}, \Pi.\text{Prove}, \Pi.\text{Verify})$ defined as follows:*

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

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

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

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

- **Zero Knowledge:** There exist stateful PPT simulators Sim_1 and Sim_2 defined as follows:

- $(\widetilde{\text{pp}}_\Pi, \tau_\Pi) \leftarrow \text{Sim}_1(1^\lambda)$, where $\widetilde{\text{pp}}_\Pi$ are the simulated public parameters and τ_Π contains the simulation trapdoor.
- $\tilde{\pi} \leftarrow \text{Sim}_2(\widetilde{\text{pp}}_\Pi, \tau_\Pi, x)$, 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 $(x, w) \in \mathcal{R}$,

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

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

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 [58]). *Given commitments $\text{Com}_\mathbf{v}$ and $\text{Com}_\mathbf{t}$ 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 $\mathbf{v} = (v_1, \dots, v_m)$ of $\text{Com}_\mathbf{v}$ 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** [36]. We treat **LogUp** as a black-box subprotocol $\Pi_{\text{LogUp}} = (\Pi_{\text{LogUp}}.\text{Prove}, \Pi_{\text{LogUp}}.\text{Verify})$ 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 $\mathbf{mult} \in \mathbb{F}^N$ be a multiplicity vector, where $\mathbf{mult}[j]$ records how many times $\mathbf{t}[j]$ occurs in the query. For a verifier challenge $\beta \xleftarrow{\$} \mathbb{F}$, multiset consistency is characterized by the identity

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

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 [19] efficiently realizes the same underlying logarithmic-derivative semantics while reducing 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 $(\mathbf{mult}[i])_{i=1}^N$, $\mathbf{t}$, and $(\mathbf{mult}[i]/(\beta - \mathbf{t}[i]))_{i=1}^N$, respectively. The core consistency constraint enforced by **cq** is

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

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 commitments $\{[Q_i(X)]_1\}_{i \in [N]}$.

Our construction makes extensive use of lookup arguments for proving that committed values occur in a committed table. The **LogUp** protocol [36] provides the basic logarithmic-derivative primitive for expressing lookup and multiplicity constraints. The **cq** protocol [19] uses cached quotients and table-dependent preprocessing so that the online prover cost depends primarily on the lookup size rather than the full table size. The **zkcq+** construction [12] further provides a fully zero-knowledge lookup argument for private tables and lookup values.

To support updates efficiently, we use the Base+Cache framework of Dutta et al. [17]. 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

In this section, we formally define Sunshine Systems. As discussed previously, a Sunshine System operates 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 universe (or 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 universe (or set) of possible keywords. We will represent each document as a pair $(\text{id}, \mathcal{W})$, where $\text{id} \in \mathcal{ID}$ is its identifier and $\mathcal{W} \subseteq \mathcal{K}$ is the set of keywords associated with the document. We assume that the universes $\mathcal{ID}$ 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*

$$\mathcal{D} = \{(\text{id}_1, \mathcal{W}_1), \dots, (\text{id}_M, \mathcal{W}_M)\},$$

where each document identifier $\text{id}_i \in \mathcal{ID}$ 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 $i \neq j$, $\text{id}_i \neq \text{id}_j$.

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,

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

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,

$$\text{Res}_{\mathcal{D}}(\mathcal{Q}) := \left\{ \text{id}_i \mid \{q_1, \dots, q_\ell\} \subseteq \mathcal{W}_i \wedge (\text{id}_i, \mathcal{W}_i) \in \mathcal{D} \right\}.$$

Raw Document Files $\leftrightarrow$ Identifiers. We assume that each raw document file is hashed using a salted cryptographic hash in order to produce the identifiers in $\mathcal{ID}$. As this commitment serves as a hiding and binding commitment to the document itself, we can safely reveal the identifiers publicly. 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 some digitally-native representation. When completing a query response, agencies can separately return the document identifiers along with their salt and the verifier can check for consistency. 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 universe of keywords and $\mathcal{ID}$ be a universe of identifiers. The online query phase in a Sunshine System comprises a tuple of PPT algorithms (Setup, Commit, Response, Verify) defined as follows:*

- **Setup**(1^λ) $\rightarrow$ **pp**: *This algorithm takes the security parameter 1^λ as input and outputs public parameters **pp**.*

³We 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.

- $\text{Commit}(\text{pp}, \mathcal{D}) \rightarrow (\text{com}, \text{op})$: 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 .
- $\text{Response}(\text{pp}, \text{com}, \text{op}, \mathcal{D}, \mathcal{Q}) \rightarrow (\mathcal{J}, \pi)$: 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 .
- $\text{Verify}(\text{pp}, \text{com}, \mathcal{Q}, \mathcal{J}, \pi) \rightarrow b$: 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 $\text{pp} \leftarrow \text{Setup}(1^\lambda)$. 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

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

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

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

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

3. **Hiding Commitment:** Let $\text{pp} \leftarrow \text{Setup}(1^\lambda)$. 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

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

4. **Zero Knowledge:** There exist PPT simulators $\text{Sim}_{\text{online},1}$ and $\text{Sim}_{\text{online},2}$ such that, for any well-formed set of documents $\mathcal{D}$, every query $\mathcal{Q} \subseteq \mathcal{K}$ with $1 \leq |\mathcal{Q}| \leq c$, and every n.u. PPT adversary Adv , it holds that

$$\left| \Pr \left[\begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda), \\ (\text{com}, \text{op}) \leftarrow \text{Commit}(\text{pp}, \mathcal{D}), \\ (\mathcal{J}, \pi_{\text{online}}) \leftarrow \text{Response}(\text{pp}, \text{com}, \text{op}, \mathcal{D}, \mathcal{Q}), \\ b \leftarrow \text{Adv}(\text{pp}, \text{com}, \mathcal{Q}, \mathcal{J}, \pi_{\text{online}}) \end{array} \right] \right. \\ \left. - \Pr \left[\begin{array}{l} (\widetilde{\text{pp}}, \tau_{\text{online}}) \leftarrow \text{Sim}_{\text{online},1}(1^\lambda), \\ (\widetilde{\text{com}}, \cdot) \leftarrow \text{Commit}(\widetilde{\text{pp}}, \mathcal{D}), \\ \widetilde{\pi_{\text{online}}} \leftarrow \text{Sim}_{\text{online},2}(\widetilde{\text{pp}}, \tau_{\text{online}}, \widetilde{\text{com}}, \mathcal{Q}, \text{Res}_{\mathcal{D}}(\mathcal{Q})), \\ \widetilde{b} \leftarrow \text{Adv}(\widetilde{\text{pp}}, \widetilde{\text{com}}, \mathcal{Q}, \text{Res}_{\mathcal{D}}(\mathcal{Q}), \widetilde{\pi_{\text{online}}}) \end{array} \right] \right| \leq \text{negl}(\lambda).$$

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

⁴This 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.

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. Since audits are intended to verify integrity of this evolving system over time, we expect them to occur less frequently than query issuance.

Definition 6 (Sunshine System: Audit Phase). *Let $\lambda \in \mathbb{N}$ be the security parameter, let $\mathcal{K}$ be a universe of keywords, and let $\mathcal{ID}$ be a universe 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:*

- **Update**($\text{pp}, \mathcal{D}, \text{com}, \text{op}, \Delta, \text{com}^+, \text{op}^+$) $\rightarrow \pi_{\text{audit}}$: *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*
- **Audit**($\text{pp}, \text{com}, \text{com}^+, \pi_{\text{audit}}$) $\rightarrow b$: *This algorithm takes the public parameters pp , initial and new commitments com, 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 $\text{pp} \leftarrow \text{Setup}(1^\lambda)$. For any well-formed document sets $\mathcal{D}$ and Δ such that $\mathcal{D} \cup \Delta$ is well-formed and $\nexists \text{id}, \mathcal{W}, \mathcal{W}' : (\text{id}, \mathcal{W}) \in \mathcal{D} \wedge (\text{id}, \mathcal{W}') \in \Delta$, let $(\text{com}, \text{op}) \leftarrow \text{Commit}(\text{pp}, \mathcal{D})$ and $(\text{com}^+, \text{op}^+) \leftarrow \text{Commit}(\text{pp}, \mathcal{D} \cup \Delta)$.⁶ Then it holds that*

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

2. **Knowledge Soundness:** *Let $\text{pp} \leftarrow \text{Setup}(1^\lambda)$. For every n.u. stateful PPT adversary Adv , there exists a PPT extractor Ext , such that if $(\text{com}^*, \text{com}^{+*}, \pi_{\text{audit}}^*) \leftarrow \text{Adv}(\text{pp})$, then it holds that*

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

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

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

⁵We 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.

⁶We leave the private choice of representation implicit in the notation **Commit**($\text{pp}, \mathcal{D}$). 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.

$\mathcal{D} \wedge (\text{id}, \mathcal{W}') \in \Delta$, and every n.u. PPT adversary Adv , it holds that

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

Remark 1 (Incremental efficiency). *The initial certification of a public commitment may require processing the entire committed state. 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 certify both the previously accepted Audit record and the transition to the updated commitment. In a sequential execution, the auditor may instead retain the most recently accepted Audit record and use it as the predecessor state for the next transition. A subsequent invocation then needs to certify only the transition from this previously accepted Audit record to the proposed updated commitment. Our construction makes this state reuse explicit in Section 7.*

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.

6.1 Committing to a Document Set

For simplicity, let the universe of document identifiers be $\mathcal{ID} = [n]$ and the keyword universe be $\mathcal{K} = [m]$. We 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 $(\text{id}, \mathcal{W})$, where $\text{id} \in \mathcal{ID}$ is its identifier and $\mathcal{W} \subseteq \mathcal{K}$ is its set of keywords. For our protocol, we represent $(\text{id}, \mathcal{W})$ by the set of keyword–identifier pairs $\{(k, \text{id}) : k \in \mathcal{W}\}$. 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 $E_{\text{key}}, E_{\text{doc}} \in \mathbb{F}^N$ of fixed public capacity N . The entries $(E_{\text{key}}[j], E_{\text{doc}}[j])$ 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 $1, \dots, T$ contain exactly these pairs,

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

Document set $\mathcal{D} = \{(\text{id}_i, \mathcal{W}_i)\}_{i \in [3]}$

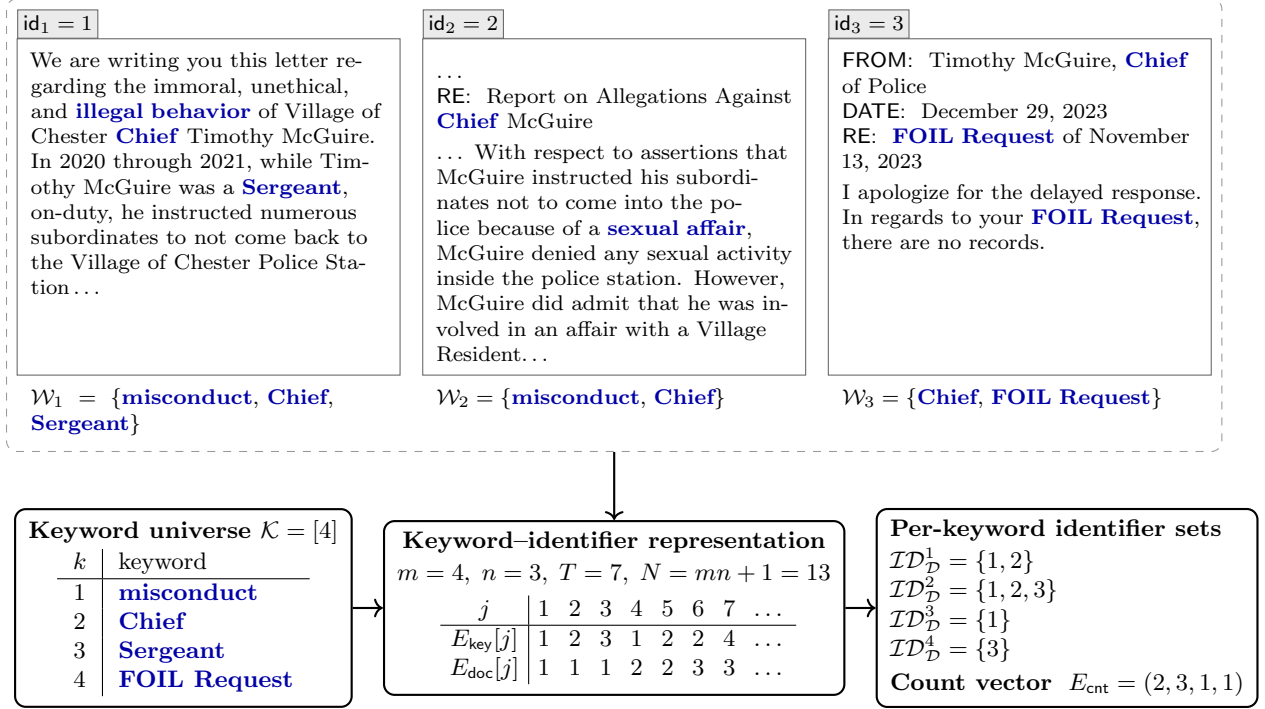


Figure 1: Example of the document-set representation, built from records of the Village of Chester, NY police department investigation discussed in Section 1 [61]. The first two documents contain verbatim excerpts from relevant documents [3, 51], while the third is reconstructed from *Sussman v. Village of Chester* [62]. Each document is assigned an identifier 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., $\text{Res}_{\mathcal{D}}(\{1, 2\}) = \mathcal{ID}_{\mathcal{D}}^1 \cap \mathcal{ID}_{\mathcal{D}}^2 = \{1, 2\}$. The document set is represented by its keyword-identifier pairs, stored in the first T positions of the fixed-capacity vectors ($E_{\text{key}}, E_{\text{doc}}$); the remaining positions are padded with $(0, 0)$. The count vector E_{cnt} records the number of documents associated with each keyword, and the sets $\mathcal{ID}_{\mathcal{D}}^k$ are derived from $(E_{\text{key}}, E_{\text{doc}})$.

while positions $T + 1, \dots, N$ 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.

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

See Figure 1 for an example.

Commitment. After constructing $E_{\text{key}}, E_{\text{doc}}$, 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

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

which binds the prover to the above representation of $\mathcal{D}$.

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

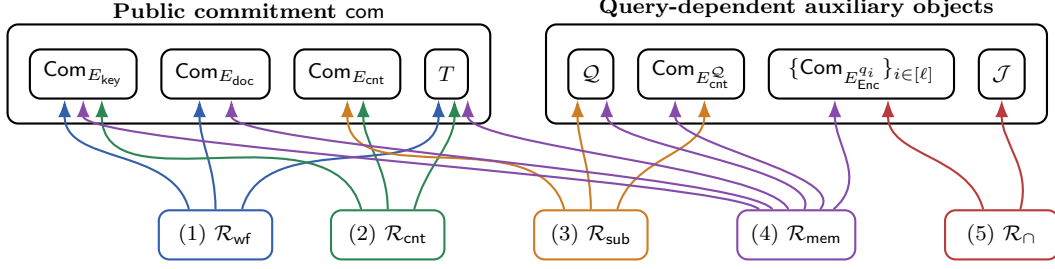


Figure 2: Overview of the five relations underlying the online query protocol. The query-independent relations $\mathcal{R}_{\text{wf}}$ and $\mathcal{R}_{\text{cnt}}$ certify the well-formedness of the committed document-set representation and the consistency of its count vector. The query-dependent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

6.2 Query Response and Verification

Recall from Section 5.1 that, on a query $Q \subseteq \mathcal{K}$, the agency returns a candidate response $\mathcal{J} \subseteq \mathcal{ID}$ together with a proof that $\mathcal{J} = \text{Res}_{\mathcal{D}}(Q)$, where $\mathcal{D}$ is a well-formed document set represented by $(E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, T)$ 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-knowledge arguments 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 $\mathcal{R}_{\text{wf}}$. The relation $\mathcal{R}_{\text{wf}}$ checks that the committed pair vectors encode a well-formed document set (see Definition 4):

- **Statement:** $x_{\text{wf}} := (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, T)$.
- **Witness:** $w_{\text{wf}} := (E_{\text{key}}, E_{\text{doc}}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{doc}}})$.
- **Relation:** We define $\mathcal{R}_{\text{wf}}(x_{\text{wf}}, w_{\text{wf}}) = 1$ if:
 1. *Valid commitments:* $(E_{\text{key}}, \text{op}_{E_{\text{key}}})$ and $(E_{\text{doc}}, \text{op}_{E_{\text{doc}}})$ are valid openings of $\text{Com}_{E_{\text{key}}}$ and $\text{Com}_{E_{\text{doc}}}$, respectively.
 2. *Valid nonzero entries:* For every $j \in [T]$, $E_{\text{key}}[j] \in \mathcal{K}$ and $E_{\text{doc}}[j] \in \mathcal{ID}$.
 3. *Zero padding:* For every $j \in \{T+1, \dots, N\}$, $E_{\text{key}}[j] = E_{\text{doc}}[j] = 0$.
 4. *No duplicate keyword-identifier pairs:* For every distinct $j, j' \in [T]$, $E_{\text{doc}}[j] = E_{\text{doc}}[j'] \implies E_{\text{key}}[j] \neq E_{\text{key}}[j']$.

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

- **Statement:** $x_{\text{cnt}} := (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{cnt}}}, T)$.
- **Witness:** $w_{\text{cnt}} := (E_{\text{key}}, E_{\text{cnt}}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{cnt}}})$.
- **Relation:** We define $\mathcal{R}_{\text{cnt}}(x_{\text{cnt}}, w_{\text{cnt}}) = 1$ if:
 1. *Valid commitments:* $(E_{\text{key}}, \text{op}_{E_{\text{key}}})$ and $(E_{\text{cnt}}, \text{op}_{E_{\text{cnt}}})$ are valid openings of $\text{Com}_{E_{\text{key}}}$ and $\text{Com}_{E_{\text{cnt}}}$, respectively.
 2. *Consistent counts:* For every $k \in \mathcal{K}$, $E_{\text{cnt}}[k] = |\{j \in [T] : E_{\text{key}}[j] = k\}|$.

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 $\mathcal{ID}_{\mathcal{D}}^k := \{\text{id} \in \mathcal{ID} \mid \exists j \in [T] : E_{\text{key}}[j] = k \wedge E_{\text{doc}}[j] = \text{id}\}$ denote the set of document identifiers associated with k . Write $\mathcal{Q} = \{q_1, \dots, q_\ell\}$ in the canonical order inherited from $\mathcal{K} = [m]$.

The prover constructs the query-specific count vector $E_{\text{cnt}}^{\mathcal{Q}} := (E_{\text{cnt}}[q_1], \dots, E_{\text{cnt}}[q_\ell])$, which records the number of documents associated with each queried keyword.

We fix the public capacity $h := n + 1$ 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. 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 $c_i := E_{\text{cnt}}^{\mathcal{Q}}[i]$ and define $E_{\text{Enc}}^{q_i} := (\text{id}_{i,1}, \dots, \text{id}_{i,c_i}, 0, \dots, 0) \in \mathbb{F}^h$, where the first c_i entries are intended to contain the distinct identifiers associated with q_i . Let $\mathcal{E}_{\text{doc}}^{q_i} := \{E_{\text{Enc}}^{q_i}[j] \mid j \in [h] \wedge E_{\text{Enc}}^{q_i}[j] \neq 0\}$ denote the set represented by these entries. The prover commits to $E_{\text{cnt}}^{\mathcal{Q}}$ and each $E_{\text{Enc}}^{q_i}$ and sets $\text{Com}_{\text{aux}} := (\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}, \{\text{Com}_{E_{\text{Enc}}^{q_i}}\}_{i \in [\ell]})$.

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

- **Statement:** $x_{\text{sub}} := (\text{Com}_{E_{\text{cnt}}}, \text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}, \mathcal{Q})$.
- **Witness:** $w_{\text{sub}} := (E_{\text{cnt}}, E_{\text{cnt}}^{\mathcal{Q}}, \text{op}_{E_{\text{cnt}}}, \text{op}_{E_{\text{cnt}}^{\mathcal{Q}}})$.
- **Relation:** We define $\mathcal{R}_{\text{sub}}(x_{\text{sub}}, w_{\text{sub}}) = 1$ if:
 1. *Valid commitments:* $(E_{\text{cnt}}, \text{op}_{E_{\text{cnt}}})$ and $(E_{\text{cnt}}^{\mathcal{Q}}, \text{op}_{E_{\text{cnt}}^{\mathcal{Q}}})$ are valid openings of $\text{Com}_{E_{\text{cnt}}}$ and $\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}$, respectively.
 2. *Consistent query counts:* For every $i \in [\ell]$, $E_{\text{cnt}}^{\mathcal{Q}}[i] = E_{\text{cnt}}[q_i]$.

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

- **Statement:** $x_{\text{mem}} := (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, T, \mathcal{Q}, \text{Com}_{\text{aux}})$.
- **Witness:** $w_{\text{mem}} := (E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}^{\mathcal{Q}}, \{E_{\text{Enc}}^{q_i}\}_{i \in [\ell]}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{doc}}}, \text{op}_{E_{\text{cnt}}^{\mathcal{Q}}}, \{\text{op}_{E_{\text{Enc}}^{q_i}}\}_{i \in [\ell]})$.

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

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

Together with $\mathcal{R}_{\text{wf}}$, $\mathcal{R}_{\text{cnt}}$, and $\mathcal{R}_{\text{sub}}$, these conditions imply $\mathcal{E}_{\text{doc}}^{q_i} = \mathcal{ID}_{\mathcal{D}}^{q_i}$ 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:** $x_{\cap} := (\mathcal{J}, \{\text{Com}_{E_{\text{Enc}}^{q_i}}\}_{i \in [\ell]})$.

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

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

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

Combining the five relations yields

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

6.2.2 Online Query Phase Protocol

Fix public parameters $N, m, n, c, h, d \in \mathbb{N}$, where N is the capacity of the keyword–identifier representation, $\mathcal{K} = [m]$, $\mathcal{ID} = [n]$, c bounds the query size, 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 $\mathcal{R}_{\text{wf}}$, $\mathcal{R}_{\text{cnt}}$, $\mathcal{R}_{\text{sub}}$, $\mathcal{R}_{\text{mem}}$, 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.

Theorem 1 (Online Query Phase). *Assume that VC satisfies Definition 1 and has commitments of size $\text{poly}(\lambda)$. Assume that Π_{wf} , Π_{cnt} , Π_{sub} , Π_{mem} , and $\Pi_{\cap}$ satisfy Definition 2 for $\mathcal{R}_{\text{wf}}$, $\mathcal{R}_{\text{cnt}}$, $\mathcal{R}_{\text{sub}}$, $\mathcal{R}_{\text{mem}}$, and $\mathcal{R}_{\cap}$, respectively.⁹ Then the algorithms (Setup, Commit, Response, Verify) described in*

⁹For 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, $(x_{\rho}, \text{Ext}_{\rho}(\text{pp}_{\rho}, x_{\rho}, \pi_{\rho})) \in \mathcal{R}_{\rho}$ 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).

Protocol Π_{online} : Online Query Phase

$\text{pp} \leftarrow \text{Setup}(1^\lambda)$.

Compute $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$ and

$$\begin{aligned} \text{pp}_{\text{wf}} &\leftarrow \Pi_{\text{wf}}.\text{Setup}(1^\lambda), & \text{pp}_{\text{cnt}} &\leftarrow \Pi_{\text{cnt}}.\text{Setup}(1^\lambda), \\ \text{pp}_{\text{sub}} &\leftarrow \Pi_{\text{sub}}.\text{Setup}(1^\lambda), & \text{pp}_{\text{mem}} &\leftarrow \Pi_{\text{mem}}.\text{Setup}(1^\lambda), \\ \text{pp}_{\cap} &\leftarrow \Pi_{\cap}.\text{Setup}(1^\lambda). \end{aligned}$$

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

$(\text{com}, \text{op}) \leftarrow \text{Commit}(\text{pp}, \mathcal{D})$.

Construct $(E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, T)$ from $\mathcal{D}$ and compute

$$\begin{aligned} (\text{Com}_{E_{\text{key}}}, \text{op}_{E_{\text{key}}}) &\leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, E_{\text{key}}), \\ (\text{Com}_{E_{\text{doc}}}, \text{op}_{E_{\text{doc}}}) &\leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, E_{\text{doc}}), \\ (\text{Com}_{E_{\text{cnt}}}, \text{op}_{E_{\text{cnt}}}) &\leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, E_{\text{cnt}}). \end{aligned}$$

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

$$\text{op} := (E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{doc}}}, \text{op}_{E_{\text{cnt}}}).$$

$(\mathcal{J}, \pi_{\text{online}}) \leftarrow \text{Response}(\text{pp}, \text{com}, \text{op}, \mathcal{D}, \mathcal{Q})$.

Write $\mathcal{Q} = \{q_1, \dots, q_\ell\}$ in canonical order. Compute $\mathcal{J} := \text{Res}_{\mathcal{D}}(\mathcal{Q})$ and $E_{\text{cnt}}^{\mathcal{Q}} := (E_{\text{cnt}}[q_1], \dots, E_{\text{cnt}}[q_\ell])$. For each $i \in [\ell]$, construct $E_{\text{Enc}}^{q_i} \in \mathbb{F}^h$ whose nonzero entries are exactly the elements of $\mathcal{ID}_{\mathcal{D}}^{q_i}$ (see Section 6.2.1).

Compute

$$(\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}, \text{op}_{E_{\text{cnt}}^{\mathcal{Q}}}) \leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, E_{\text{cnt}}^{\mathcal{Q}})$$

and, for each $i \in [\ell]$,

$$(\text{Com}_{E_{\text{Enc}}^{q_i}}, \text{op}_{E_{\text{Enc}}^{q_i}}) \leftarrow \text{VC.Commit}(\text{pp}_{\text{VC}}, E_{\text{Enc}}^{q_i}).$$

Set $\text{Com}_{\text{aux}} := (\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}, \{\text{Com}_{E_{\text{Enc}}^{q_i}}\}_{i \in [\ell]})$.

Compute

$$\begin{aligned} \pi_{\text{wf}} &\leftarrow \Pi_{\text{wf}}.\text{Prove}(\text{pp}_{\text{wf}}, x_{\text{wf}}, w_{\text{wf}}), \\ \pi_{\text{cnt}} &\leftarrow \Pi_{\text{cnt}}.\text{Prove}(\text{pp}_{\text{cnt}}, x_{\text{cnt}}, w_{\text{cnt}}), \\ \pi_{\text{sub}} &\leftarrow \Pi_{\text{sub}}.\text{Prove}(\text{pp}_{\text{sub}}, x_{\text{sub}}, w_{\text{sub}}), \\ \pi_{\text{mem}} &\leftarrow \Pi_{\text{mem}}.\text{Prove}(\text{pp}_{\text{mem}}, x_{\text{mem}}, w_{\text{mem}}), \\ \pi_{\cap} &\leftarrow \Pi_{\cap}.\text{Prove}(\text{pp}_{\cap}, x_{\cap}, w_{\cap}). \end{aligned}$$

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

$b \leftarrow \text{Verify}(\text{pp}, \text{com}, \mathcal{Q}, \mathcal{J}, \pi_{\text{online}})$.

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

Compute

$$\begin{aligned} b_{\text{wf}} &\leftarrow \Pi_{\text{wf}}.\text{Verify}(\text{pp}_{\text{wf}}, x_{\text{wf}}, \pi_{\text{wf}}), \\ b_{\text{cnt}} &\leftarrow \Pi_{\text{cnt}}.\text{Verify}(\text{pp}_{\text{cnt}}, x_{\text{cnt}}, \pi_{\text{cnt}}), \\ b_{\text{sub}} &\leftarrow \Pi_{\text{sub}}.\text{Verify}(\text{pp}_{\text{sub}}, x_{\text{sub}}, \pi_{\text{sub}}), \\ b_{\text{mem}} &\leftarrow \Pi_{\text{mem}}.\text{Verify}(\text{pp}_{\text{mem}}, x_{\text{mem}}, \pi_{\text{mem}}), \\ b_{\cap} &\leftarrow \Pi_{\cap}.\text{Verify}(\text{pp}_{\cap}, x_{\cap}, \pi_{\cap}). \end{aligned}$$

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

Figure 3: Online query phase of the Sunshine System.

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 5.¹⁰

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

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

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

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

For each $\rho \in \{\text{wf}, \text{cnt}, \text{sub}, \text{mem}, \cap\}$, 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 $w_\rho^* \leftarrow \text{Ext}_\rho(\text{pp}_\rho, x_\rho^*, \pi_\rho^*)$. The witness w_ρ^* for $\mathcal{R}_\rho$ contains all vectors and corresponding commitment openings required by that relation.

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

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

$$w_\rho^* \leftarrow \text{Ext}_\rho(\text{pp}_\rho, x_\rho^*, \pi_\rho^*).$$

2. For every ρ , $\text{Ext}_{\text{online}}$ checks whether $(x_\rho^*, w_\rho^*) \in \mathcal{R}_\rho$. Let Bad_{ext} denote the event that Acc occurs and $(x_\rho^*, w_\rho^*) \notin \mathcal{R}_\rho$ for some ρ . If Bad_{ext} occurs, $\text{Ext}_{\text{online}}$ outputs $\perp$ and terminates.
3. $\text{Ext}_{\text{online}}$ compares the vector values assigned to every commitment shared by different relations. In particular, $\text{Ext}_{\text{online}}$ checks consistency of

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

¹⁰For this claim, we additionally require the proof size of $\Pi_{\cap}$ be $\text{poly}(\lambda, |\mathcal{Q}|)$, independently of $|\mathcal{J}|$.

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

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

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

Probability that event Bad_{ext} happens. Fix ρ . Acc implies $\Pi_{\rho}.\text{Verify}(\text{pp}_{\rho}, x_{\rho}^*, \pi_{\rho}^*) = 1$. By knowledge soundness of Π_{ρ} , $\Pr[\text{Acc} \wedge (x_{\rho}^*, w_{\rho}^*) \notin \mathcal{R}_{\rho}] \leq \text{negl}_{\rho}(\lambda)$. Since there are only five sub-arguments,

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

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

$$\Pr[\text{Bad}_{\text{bind}}] \leq \text{negl}(\lambda).$$

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

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

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

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

Suppose, for the sake of contradiction, that $\Pr[\text{Acc} \wedge \text{Fail}]$ is non-negligible. Under $\text{Acc} \wedge \text{Fail}$, both checks of $\text{Ext}_{\text{online}}$ 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: $\mathcal{D}^*$ is not well formed. The extracted witness w_{wf}^* satisfies $\mathcal{R}_{\text{wf}}$. Hence, for every $j \in [T^*]$, $E_{\text{key}}[j] \in \mathcal{K}$ and $E_{\text{doc}}[j] \in \mathcal{ID}$, every position after T^* contains $(0, 0)$, and no keyword-identifier pair occurs twice among the first T^* positions. By construction, $\mathcal{D}^*$ contains one document for each distinct identifier appearing in the first T^* entries of E_{doc} , and every such document has a nonempty keyword set contained in $\mathcal{K}$. Therefore $\mathcal{D}^*$ is well-formed, a contradiction.

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

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

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

which contradicts this case.

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

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

Together with the previous bound on the probability that Acc occurs and $\text{Ext}_{\text{online}}$ 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

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

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

Zero knowledge. For every $\rho \in \{\text{wf}, \text{cnt}, \text{sub}, \text{mem}, \cap\}$, let $(\text{Sim}_{\rho,1}, \text{Sim}_{\rho,2})$ denote the setup and proof simulators guaranteed by Definition 2 for Π_{ρ} . The setup simulator produces $(\widetilde{\text{pp}}_{\rho}, \tau_{\rho})$, and the proof simulator uses these outputs and a statement to generate a proof without its witness. We construct the simulators $\text{Sim}_{\text{online},1}$ and $\text{Sim}_{\text{online},2}$ required by Definition 5.

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

Proof simulation. Let $\mathcal{Q} = \{q_1, \dots, q_{\ell}\}$. On input $(\widetilde{\text{pp}}, \tau_{\text{online}}, \widetilde{\text{com}}, \mathcal{Q}, \mathcal{J})$, $\text{Sim}_{\text{online},2}$ proceeds as follows.

1. Parse $\widetilde{\text{com}} = (\widetilde{\text{Com}}_{E_{\text{key}}}, \widetilde{\text{Com}}_{E_{\text{doc}}}, \widetilde{\text{Com}}_{E_{\text{cnt}}}, T)$.
2. Generate the shared auxiliary commitments with fresh independent randomness by computing $(\widetilde{\text{Com}}_{E_{\text{cnt}}^{\mathcal{Q}}}, \cdot) \leftarrow \text{VC.Commit}(\widetilde{\text{pp}}_{\text{VC}}, 0^{\ell})$ and, for every $i \in [\ell]$, $(\widetilde{\text{Com}}_{E_{\text{enc}}^{q_i}}, \cdot) \leftarrow \text{VC.Commit}(\widetilde{\text{pp}}_{\text{VC}}, 0^h)$. Define $\widetilde{\text{Com}}_{\text{aux}} := (\widetilde{\text{Com}}_{E_{\text{cnt}}^{\mathcal{Q}}}, \{\widetilde{\text{Com}}_{E_{\text{enc}}^{q_i}}\}_{i \in [\ell]}).$

3. Form the five sub-statements

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

Invoke the five proof simulators in protocol order:

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

4. Output

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

Hybrid distributions. Fix a well-formed document set $\mathcal{D}$ and a valid query $\mathcal{Q}$, and set $\mathcal{J} := \text{Res}_{\mathcal{D}}(\mathcal{Q})$. Each hybrid is a distribution of the complete view $(\text{pp}, \text{com}, \mathcal{Q}, \mathcal{J}, \pi_{\text{online}})$, generated according to the rules below.

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

Indistinguishability.

$\text{H}_0 \approx_c \text{H}_1$. For every $k \in \{0, \dots, 5\}$, let G_k replace the setup and proof of the first k arguments, in the order $\Pi_{\text{wf}}, \Pi_{\text{cnt}}, \Pi_{\text{sub}}, \Pi_{\text{mem}}, \Pi_{\cap}$, 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 $\text{G}_0 = \text{H}_0$ and $\text{G}_5 = \text{H}_1$.

Fix $k \in [5]$, 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 subproofs 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 $G_{k-1} \approx_c G_k$. By transitivity,

$$H_0 = G_0 \approx_c G_1 \approx_c \dots \approx_c G_5 = H_1.$$

$H_1 \approx_c H_2$. The two hybrids differ only in whether the commitment in the $E_{\text{cnt}}^{\mathcal{Q}}$ position is generated from $E_{\text{cnt}}^{\mathcal{Q}} \in \mathbb{F}^\ell$ or from 0^ℓ . By hiding of VC, these commitments are computationally indistinguishable. Once either commitment is fixed, the corresponding statements for Π_{sub} and Π_{mem} are formed, and their proofs are generated by the corresponding simulators as prescribed by $\text{Sim}_{\text{online},2}$. The other commitments are unchanged, and all other subproofs follow the same generation procedures. Therefore, $H_1 \approx_c H_2$.

$H_{3,i-1} \approx_c H_{3,i}$ **for every** $i \in [\ell]$. Fix $i \in [\ell]$. The two hybrids differ only in whether the i -th query-specific commitment is generated from $E_{\text{enc}}^{q_i} \in \mathbb{F}^h$ or from 0^h . By hiding of VC, these commitments are computationally indistinguishable. Once either commitment is fixed, the corresponding statements for Π_{mem} and $\Pi_\cap$ are formed, and their proofs are generated by their simulators. The other commitments are unchanged, and all other subproofs follow the same generation procedures. Thus $H_{3,i-1} \approx_c H_{3,i}$. Since $H_{3,0} = H_2$, transitivity gives

$$H_2 = H_{3,0} \approx_c H_{3,1} \approx_c \dots \approx_c H_{3,\ell}.$$

In $H_{3,\ell}$, all query-dependent commitments are to zero vectors of the prescribed public lengths, and all subproofs are simulated. Its proof component has exactly the output distribution of $\text{Sim}_{\text{online},2}$, so the full view is the simulated experiment in Definition 5. Combining the preceding transitions,

$$H_0 \approx_c H_1 \approx_c H_2 = H_{3,0} \approx_c \dots \approx_c H_{3,\ell}.$$

This establishes zero knowledge.

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

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

Thus Π_{online} satisfies succinctness as defined in Definition 5. $\square$

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. Combining the stated security assumptions of these instantiations, their security guarantees, Theorem 1, 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 3 (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 public commitment to the current document set has already been certified to satisfy $\mathcal{R}_{\text{wf}}$ and $\mathcal{R}_{\text{cnt}}$. Hence, an online response need only include $\pi_{\text{query}} := (\text{Com}_{\text{aux}}, \pi_{\text{sub}}, \pi_{\text{mem}}, \pi_{\cap})$. The verifier first checks that the commitment is the current audit-certified commitment and then verifies Π_{sub} , Π_{mem} , and $\Pi_{\cap}$, avoiding repeated proofs of $\mathcal{R}_{\text{wf}}$ and $\mathcal{R}_{\text{cnt}}$. Section 7 formalizes how the Audit Phase establishes and preserves these reusable guarantees.

Prover runtime. In the standalone setting of Remark 3, generating π_{wf} and π_{cnt} requires quasilinear work in the keyword–identifier capacity N and the keyword-domain size m . After table-dependent lookup preprocessing, the Base-only prover cost in the audit-assisted setting is quasilinear in m and in the public capacity h of each identifier vector $E_{\text{Enc}}^{q_i}$, and polynomial in the query size $\ell = |\mathcal{Q}|$. If the Cache contains δ recently appended keyword–identifier pairs, Π_{mem} incurs additional lookup work depending on ℓ , h , and δ . For fixed m, ℓ, h, δ , the resulting per-query prover bound is independent of N . The detailed bounds are given in Section A.3.

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 3, 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 process begins when the agency publishes an initial commitment. The auditor (e.g., a government accountability office or civil society organization) checks that it binds a well-formed document set with correct keyword counts; once accepted, it becomes the current certified Audit record. Query issuers either verify this record themselves or obtain it through the system’s authenticated Audit-state interface. Audit-assisted queries are accepted only with respect to this current commitment.

To update the document set, the agency publishes a new commitment to the updated document set together with a zero-knowledge update proof. The proof establishes that the appended document set is well-formed, that its document identifiers are fresh, that the new commitment preserves all previously committed keyword–identifier pairs while appending exactly the new pairs, and that the keyword counts are updated consistently. If any check fails, the current commitment remains the verified Audit record.

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 $(E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, T)$ denote the representation of $\mathcal{D}$, and let $(E_{\text{key}}^+, E_{\text{doc}}^+, E_{\text{cnt}}^+, T^+)$ denote the representation of $\mathcal{D}^+$. The corresponding public commitments are

$$\begin{aligned} \text{com} &:= (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, \text{Com}_{E_{\text{cnt}}}, T), \\ \text{com}^+ &:= (\text{Com}_{E_{\text{key}}^+}, \text{Com}_{E_{\text{doc}}^+}, \text{Com}_{E_{\text{cnt}}^+}, T^+). \end{aligned}$$

For any document set $\mathcal{X}$, define its document-identifier set as $\mathcal{ID}_{\mathcal{X}} := \{\text{id} \mid (\text{id}, \mathcal{W}) \in \mathcal{X}\}$. 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 $\text{Com}_{\mathcal{ID}_{\mathcal{D}}}$ and $\text{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$, respectively.

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

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

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

$$\mathcal{ID}_{\mathcal{D}} = \{E_{\text{doc}}[j] \mid j \in [T]\}.$$

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

- **Statement:** $x_{\text{upd}} := (\text{com}, \text{com}^+, \text{Com}_{\text{aux}}^{\text{audit}})$, where $\text{com} = (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, \text{Com}_{E_{\text{cnt}}}, T)$ and $\text{com}^+ = (\text{Com}_{E_{\text{key}}^+}, \text{Com}_{E_{\text{doc}}^+}, \text{Com}_{E_{\text{cnt}}^+}, T^+)$.
- **Witness:**

$$w_{\text{upd}} := (E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, \mathcal{ID}_{\mathcal{D}}, E_{\text{key}}^+, E_{\text{doc}}^+, E_{\text{cnt}}^+, \mathcal{ID}_{\mathcal{D}^+}, \\ \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{doc}}}, \text{op}_{E_{\text{cnt}}}, \text{op}_{\mathcal{ID}_{\mathcal{D}}}, \text{op}_{E_{\text{key}}^+}, \text{op}_{E_{\text{doc}}^+}, \\ \text{op}_{E_{\text{cnt}}^+}, \text{op}_{\mathcal{ID}_{\mathcal{D}^+}}).$$

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

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

1. *Valid commitments.* The commitments in com , com^+ , $\text{Com}_{\mathcal{ID}_{\mathcal{D}}}$, and $\text{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ open to their corresponding witness values under the openings in w_{upd} .
2. *Well-formed appended rows.* For every $j \in [B]$, $E_{\text{key}}^{\Delta}[j] \in [m]$ and $E_{\text{doc}}^{\Delta}[j] \in [n]$, 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:

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

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

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

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

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

where $\text{Com}_{\text{aux}}^{\text{audit}} := (\text{Com}_{\mathcal{ID}_{\mathcal{D}}}, \text{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$. Suppose $(x_{\text{state}}, w_{\text{state}}) \in \mathcal{R}_{\text{state}}$ and $(x_{\text{upd}}, w_{\text{upd}}) \in \mathcal{R}_{\text{upd}}$, 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 $\mathcal{R}_{\text{state}}$ at x_{state}^+ .

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

- *Well-formedness.* First, since x_{state} satisfies $\mathcal{R}_{\text{state}}$, the first T rows opened from com satisfy $\mathcal{R}_{\text{wf}}$. By the well-formedness and append-only conditions of $\mathcal{R}_{\text{upd}}$, 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, $(\text{Com}_{E_{\text{key}}^+}, \text{Com}_{E_{\text{doc}}^+}, T^+)$ and its corresponding witness satisfy $\mathcal{R}_{\text{wf}}$.
- *Count consistency.* Second, for every $k \in [m]$, $\mathcal{R}_{\text{state}}$ for x_{state} and the definition of E_{cnt}^{Δ} give

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

and

$$E_{\text{cnt}}^{\Delta}[k] = |\{j \in [B] : E_{\text{key}}^{\Delta}[j] = k\}|.$$

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

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

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

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

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

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

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

Hence all three conditions of $\mathcal{R}_{\text{state}}$ hold for x_{state}^+ , and therefore $(x_{\text{state}}^+, w_{\text{state}}^+) \in \mathcal{R}_{\text{state}}$. $\square$

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 $B = T^+ - T \leq N_\Delta$ (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 $\mathcal{R}_{\text{state}}$ and $\mathcal{R}_{\text{upd}}$, 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.

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 $\mathcal{R}_{\text{state}}$ and $\mathcal{R}_{\text{upd}}$, respectively.¹¹ Then the algorithms (Setup, Commit, Update, 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 $\text{pp} \leftarrow \text{Setup}(1^\lambda)$. Let $\mathcal{D}$ and Δ satisfy the conditions in Definition 6. Suppose (com, op) and $(\text{com}^+, \text{op}^+)$ 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 the first T entries and append the pairs induced by Δ . The statements and witnesses formed by the honest Update algorithm therefore satisfy $\mathcal{R}_{\text{state}}$ and $\mathcal{R}_{\text{upd}}$. Completeness of Π_{state} and Π_{upd} therefore implies $\text{Audit}(\text{pp}, \text{com}, \text{com}^+, \pi_{\text{audit}}) = 1$.

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

$$\begin{aligned} \text{com}^* &:= (\text{Com}_{E_{\text{key}}}^*, \text{Com}_{E_{\text{doc}}}^*, \text{Com}_{E_{\text{cnt}}}^*, T^*), \\ \text{com}^{++} &:= (\text{Com}_{E_{\text{key}}}^{++}, \text{Com}_{E_{\text{doc}}}^{++}, \text{Com}_{E_{\text{cnt}}}^{++}, T^{++}), \\ \pi_{\text{audit}}^* &:= (\text{Com}_{\text{aux}}^{\text{audit},*}, \pi_{\text{state}}^*, \pi_{\text{upd}}^*), \end{aligned}$$

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

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

For each $\rho \in \{\text{state}, \text{upd}\}$, 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 $w_\rho^* \leftarrow \text{Ext}_\rho(\text{pp}_\rho, x_\rho^*, \pi_\rho^*)$. The witness w_ρ^* for $\mathcal{R}_\rho$ contains all vectors and corresponding commitment openings required by that relation.

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

¹¹As in Theorem 1, it suffices that Π_{state} and Π_{upd} are *jointly* knowledge sound, i.e., that extraction of witnesses for $\mathcal{R}_{\text{state}}$ and $\mathcal{R}_{\text{upd}}$ 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

$\text{pp} \leftarrow \text{Setup}(1^\lambda)$.

Compute $\text{pp}_{\text{VC}} \leftarrow \text{VC.Setup}(1^\lambda, d)$, $\text{pp}_{\text{state}} \leftarrow \Pi_{\text{state}}.\text{Setup}(1^\lambda)$, and $\text{pp}_{\text{upd}} \leftarrow \Pi_{\text{upd}}.\text{Setup}(1^\lambda)$.

Output $\text{pp} := (N, m, n, N_\Delta, h_D, h_\Delta, d, \text{pp}_{\text{VC}}, \text{pp}_{\text{state}}, \text{pp}_{\text{upd}})$.

$(\text{com}, \text{op}) \leftarrow \text{Commit}(\text{pp}, \mathcal{D})$.

Construct $(E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, T)$ from $\mathcal{D}$ and compute

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

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

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

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

$$\text{op} := (E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{doc}}}, \text{op}_{E_{\text{cnt}}}).$$

$\pi_{\text{audit}} \leftarrow \text{Update}(\text{pp}, \mathcal{D}, \text{com}, \text{op}, \Delta, \text{com}^+, \text{op}^+)$.

Set $\mathcal{D}^+ := \mathcal{D} \cup \Delta$. Parse

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

and

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

together with their openings. Then, compute

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

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

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

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

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

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

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

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

$b \leftarrow \text{Audit}(\text{pp}, \text{com}, \text{com}^+, \pi_{\text{audit}})$.

Parse $\pi_{\text{audit}} = (\text{Com}_{\text{aux}}^{\text{audit}}, \pi_{\text{state}}, \pi_{\text{upd}})$, where $\text{Com}_{\text{aux}}^{\text{audit}} = (\text{Com}_{\mathcal{ID}_{\mathcal{D}}}, \text{Com}_{\mathcal{ID}_{\mathcal{D}^+}})$.

Reject unless $0 \leq T \leq T^+ < N$, $T^+ - T \leq N_\Delta$, 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

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

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

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

Figure 4: Sunshine System: Audit Phase.

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

$$w_\rho^* \leftarrow \text{Ext}_\rho(\text{pp}_\rho, x_\rho^*, \pi_\rho^*).$$

2. For every ρ , $\text{Ext}_{\text{audit}}$ checks whether $(x_\rho^*, w_\rho^*) \in \mathcal{R}_\rho$. Let Bad_{ext} denote the event that Acc occurs and $(x_\rho^*, w_\rho^*) \notin \mathcal{R}_\rho$ for some ρ . If Bad_{ext} occurs, $\text{Ext}_{\text{audit}}$ outputs $\perp$ and terminates.

3. $\text{Ext}_{\text{audit}}$ compares the vector values assigned to every commitment shared by $\mathcal{R}_{\text{state}}$ and $\mathcal{R}_{\text{upd}}$. In particular, $\text{Ext}_{\text{audit}}$ checks consistency at $\text{Com}_{E_{\text{key}}}^*$, $\text{Com}_{E_{\text{doc}}}^*$, $\text{Com}_{E_{\text{cnt}}}^*$, and $\text{Com}_{\mathcal{ID}_{\mathcal{D}}}^*$. Let Bad_{bind} denote the event that $\text{Acc} \wedge \neg \text{Bad}_{\text{ext}}$ occurs and two extracted witnesses assign different vectors to the same shared commitment. If Bad_{bind} occurs, $\text{Ext}_{\text{audit}}$ outputs $\perp$ and terminates.
4. Denote the consistent vectors by $E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, E_{\text{key}}^+, E_{\text{doc}}^+, E_{\text{cnt}}^+$, and the encoded identifier sets by $\mathcal{ID}_{\mathcal{D}}, \mathcal{ID}_{\mathcal{D}^+}$. For each identifier appearing in the first T^* positions of E_{doc} , define $\mathcal{W}_{\text{id}} := \{E_{\text{key}}[j] \mid j \in [T^*] \wedge E_{\text{doc}}[j] = \text{id}\}$, and set $\mathcal{D} := \{(\text{id}, \mathcal{W}_{\text{id}}) \mid \exists j \in [T^*] : E_{\text{doc}}[j] = \text{id}\}$. Let $B^* := T^{++} - T^*$ and, for every $j \in [B^*]$, define $E_{\text{key}}^\Delta[j] := E_{\text{key}}^+[T^* + j]$ and $E_{\text{doc}}^\Delta[j] := E_{\text{doc}}^+[T^* + j]$. Let E_{cnt}^Δ and $\mathcal{ID}_\Delta$ be the count vector and identifier set induced by these appended rows, as specified by $\mathcal{R}_{\text{upd}}$. Define

$$\Delta := \{(\text{id}, \{E_{\text{key}}^\Delta[j] \mid j \in [B^*] \wedge E_{\text{doc}}^\Delta[j] = \text{id}\}) \mid \text{id} \in \mathcal{ID}_\Delta\}.$$

Take the openings of the three commitments in com^* from w_{state}^* , and those in com^{++} from w_{upd}^* . Assemble

$$\begin{aligned} \text{op}^* &:= (E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, \text{op}_{E_{\text{key}}}^*, \text{op}_{E_{\text{doc}}}^*, \text{op}_{E_{\text{cnt}}}^*), \\ \text{op}^{++} &:= (E_{\text{key}}^+, E_{\text{doc}}^+, E_{\text{cnt}}^+, \text{op}_{E_{\text{key}}^+}^*, \text{op}_{E_{\text{doc}}^+}^*, \text{op}_{E_{\text{cnt}}^+}^*) \end{aligned}$$

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

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

Probability that event Bad_{ext} happens. Fix ρ . Acc implies $\Pi_\rho.\text{Verify}(\text{pp}_\rho, x_\rho^*, \pi_\rho^*) = 1$. By knowledge soundness of Π_ρ , $\Pr[\text{Acc} \wedge (x_\rho^*, w_\rho^*) \notin \mathcal{R}_\rho] \leq \text{negl}_\rho(\lambda)$. Since there are only two sub-arguments,

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

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

$$\Pr[\text{Bad}_{\text{bind}}] \leq \text{negl}(\lambda).$$

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

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

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

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

Suppose, for the sake of contradiction, that $\Pr[\text{Acc} \wedge \text{Fail}]$ is non-negligible. Under $\text{Acc} \wedge \text{Fail}$, both checks of $\text{Ext}_{\text{audit}}$ 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_{state}^* satisfies $\mathcal{R}_{\text{state}}$, so the first T^* positions of $(E_{\text{key}}, E_{\text{doc}})$ 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 $\mathcal{R}_{\text{upd}}$ similarly establishes that Δ is well-formed.

Identifier consistency of $\mathcal{R}_{\text{state}}$ and freshness in $\mathcal{R}_{\text{upd}}$ give $\mathcal{ID}_{\mathcal{D}} = \{E_{\text{doc}}[j] \mid j \in [T^*]\}$ 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 $(\text{id}, \mathcal{W}) \in \mathcal{D}$ and $(\text{id}, \mathcal{W}') \in \Delta$ for some $\text{id}, \mathcal{W}, \mathcal{W}'$. Identifier consistency of $\mathcal{R}_{\text{state}}$ and the definition of $\mathcal{ID}_{\Delta}$ imply $\text{id} \in \mathcal{ID}_{\mathcal{D}} \cap \mathcal{ID}_{\Delta}$. However, the fresh-identifier condition of $\mathcal{R}_{\text{upd}}$ 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_{state}^* provides valid openings of all three commitments in com^* . Its well-formedness condition identifies the pair vectors with the representation used to construct $\mathcal{D}$, and its count-consistency condition gives $E_{\text{cnt}}[k] = |\{j \in [T^*] : E_{\text{key}}[j] = k\}|$ for every $k \in \mathcal{K}$. Thus $(E_{\text{key}}, E_{\text{doc}}, E_{\text{cnt}}, T^*)$ is a valid committed representation of $\mathcal{D}$, and op^* opens com^* to this document set.

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

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

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

$$\begin{aligned} E_{\text{cnt}}^+[k] &= E_{\text{cnt}}[k] + E_{\text{cnt}}^{\Delta}[k] \\ &= |\{j \in [T^*] : E_{\text{key}}[j] = k\}| + |\{j \in [B^*] : E_{\text{key}}^{\Delta}[j] = k\}| \\ &= |\{j \in [T^{++}] : E_{\text{key}}^+[j] = k\}|. \end{aligned}$$

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

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

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

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

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

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

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

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

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

Invoke the two proof simulators in protocol order:

$$\begin{aligned} \widetilde{\pi}_{\text{state}} &\leftarrow \text{Sim}_{\text{state},2}(\widetilde{\text{pp}}_{\text{state}}, \tau_{\text{state}}, \widetilde{x}_{\text{state}}), \\ \widetilde{\pi}_{\text{upd}} &\leftarrow \text{Sim}_{\text{upd},2}(\widetilde{\text{pp}}_{\text{upd}}, \tau_{\text{upd}}, \widetilde{x}_{\text{upd}}). \end{aligned}$$

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

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

- H_0 is the real Audit transcript.
- H_1 uses $\text{Sim}_{\text{Audit},1}$ 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 $\mathcal{ID}_D$ with $\widetilde{\text{Com}}_{\mathcal{ID}_D}$. 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 $\mathcal{ID}_{D^+}$ with $\widetilde{\text{Com}}_{\mathcal{ID}_{D^+}}$. The statement and its simulated proof for Π_{upd} are regenerated using the new commitments.

Indistinguishability.

$H_0 \approx_c H_1$. For every $k \in \{0, 1, 2\}$, let G_k replace the setup and proof of the first k arguments, in the order $\Pi_{\text{state}}, \Pi_{\text{upd}}$, 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 $G_0 = H_0$ and $G_2 = H_1$.

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 $G_{k-1} \approx_c G_k$. By transitivity,

$$H_0 = G_0 \approx_c G_1 \approx_c G_2 = H_1.$$

$H_1 \approx_c H_2$. The two hybrids differ only in whether $\text{Com}_{\mathcal{ID}_{\mathcal{D}}}$ 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 $\text{Sim}_{\text{Audit},2}$. The other commitments remain unchanged. Therefore, $H_1 \approx_c H_2$.

$H_2 \approx_c H_3$. The two hybrids differ in whether the new $\text{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ 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 $\text{Sim}_{\text{upd},2}$. The other commitments remain unchanged, and all other subproofs follow the same generation procedures. Therefore, $H_2 \approx_c H_3$.

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 $\text{Sim}_{\text{Audit},2}$, so the full view is the simulated experiment in Definition 6. Combining the preceding transitions,

$$H_0 \approx_c H_1 \approx_c H_2 \approx_c H_3.$$

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. Combining the stated security assumptions of these instantiations, their security guarantees and Theorem 2, we obtain 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 4 (Response monotonicity). *Every accepted update preserves all previously committed documents and appends only documents with fresh identifiers. Hence, for every query $\mathcal{Q}$, $\text{Res}_{\mathcal{D}}(\mathcal{Q}) \subseteq \text{Res}_{\mathcal{D} \cup \Delta}(\mathcal{Q})$. This follows directly from the append-only semantics of $\mathcal{R}_{\text{upd}}$.*

Remark 5 (Standalone and subsequent audits). *Figure 4 describes a standalone Audit invocation, with proof $\pi_{\text{audit}} = (\text{Com}_{\text{aux}}^{\text{audit}}, \pi_{\text{state}}, \pi_{\text{upd}})$. In a sequential execution, the auditor first accepts $x_{\text{state}}^{(0)} = (\text{com}^{(0)}, \text{Com}_{\mathcal{ID}_{\mathcal{D}(0)}})$ together with $\pi_{\text{state}}^{(0)}$, and records $x_{\text{state}}^{(0)}$ as the current verified Audit record.*

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

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

It accepts only if $\text{com}^{(r-1)}$ and $\text{Com}_{\mathcal{ID}_{\mathcal{D}(r-1)}}$ match its current verified Audit record and

$$\Pi_{\text{upd}}.\text{Verify}(\text{pp}_{\text{upd}}, x_{\text{upd}}^{(r)}, \pi_{\text{upd}}^{(r)}) = 1.$$

By Lemma 1, $x_{\text{state}}^{(r)} = (\text{com}^{(r)}, \text{Com}_{\mathcal{ID}_{\mathcal{D}(r)}})$ then satisfies $\mathcal{R}_{\text{state}}$ and becomes the new verified Audit record.

Remark 6 (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. [17] (see Section 4.3) to maintain this preprocessing. A previously verified commitment is selected as 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 latest verified commitment becomes the new Base, the Cache is cleared, and the table-dependent preprocessing is recomputed. Rebasing does not change the public commitment and therefore does not generate new proofs.

Prover runtime. Generating the initial π_{state} requires quasilinear work in the keyword–identifier capacity N , the keyword-domain size m , and the document-identifier encoding capacity h_D . For subsequent audits, generating π_{upd} for $B = T^+ - T$ appended keyword–identifier pairs requires quasilinear work in m , the appended-pair encoding capacity N_Δ , and the identifier-encoding capacities h_D and h_Δ for the complete and appended document sets, respectively.

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 now present our end-to-end implementation of the Sunshine System and a concrete evaluation of its performance. We provide all artifacts necessary to replicate our experiments 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 2025 MacBook Air with an Apple M4 processor and 16 GB of RAM, a standard commodity laptop available refurbished for $\approx$ \$800. Unless explicitly labeled otherwise, all benchmarks use strict single-thread execution, and the numbers we report are the average of five runs.

We conduct experiments using the Enron email corpus [39]; while this dataset was not generated by government agencies, it has become a standard information processing dataset. We choose $m = 2^{10}$ keywords to associate with the emails by scraping the most common non-stopword, non-numeric terms in the corpus. For the purposes of our experiment, we upper-bound the number of keywords associated with each document at 32, but opportunistically split documents if they had more than 32 keywords; the resulting artifact contains 50,520 documents and 1,048,570 keyword–identifier pairs. Queries are evaluated over these records; splitting need not preserve an original email’s conjunction matches when its keywords fall into different records. We used a maximum

Query Keywords	Prover	Π_{sub}	Π_{mem}	$\Pi_{\cap}$	Verifier	Size (KiB)
2	18.16	0.78	3.70	13.67	0.87	38.6
4	49.36	0.77	9.10	39.48	1.08	95.3
8	110.16	0.78	16.05	93.28	1.54	208.8
16	246.9	0.78	52.96	193.08	2.42	435.8

Table 1: Audit-assisted Online query performance for a query with a fixed number of keywords. Query were chosen such that they all return 5 documents. The “Prover” column reports total time, whereas Π_{sub} , Π_{mem} , and $\Pi_{\cap}$ report the part of overall prover time devoted to these subroutines. All times in seconds.

log capacity of $N = 2^{20}$ and initialized the base with 838,846 keyword–identifier pairs (about 80% of the log capacity). Each batch appended to the log had a maximum size of $B = 2^8 = 256$ keyword–identifier pairs, and the system was configured to rebase after 64 batches.

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 prover recomputes the preprocessing.

Online query phase. We measure the Audit-assisted Online mode of Remark 3. These measurements cover Π_{sub} , Π_{mem} , and $\Pi_{\cap}$, but exclude initial certification, setup, preprocessing, artifact loading, and Audit operations. Reported proof sizes exclude the public statement.

Our benchmarks consider a system with an initial base of 838,846 keyword–identifier pairs and a half-full cache, with 32 processed updates and 7,834 keyword–identifier pairs. In Table 1, we show the proof generation times, proof verification times, and proof sizes of answering queries with ℓ keywords for $\ell \in \{2, 4, 8, 16\}$. Each query was selected such that there were 5 identifiers in the corpus that would be returned by the query. Recall that this application does not require sub-second prover times, as agencies are not responding to these requests at scale. We find that proofs can be generated sufficiently quickly to fit within a reasonable workflow for a government agency worker.

In Table 1, we also provide a breakdown of the prover times for the different subprotocols within our query response. These results clearly highlight that the main performance bottleneck is the intersection subprotocol, the cost of which is driven by the number of keywords in the query. Thus, as long as the number of keywords in queries remains reasonably bounded, performance will continue to be good; we observe that a conjunction of 16 keywords is already quite specific, and queries with substantially more keywords seem unlikely.

Finally, we note that the prover time for π_{mem} is dependent on the cache size. To demonstrate this dependency, we consider a query with $\ell = 4$ keywords, but vary the cache size, capturing the proof generation times after 1, 16, 32, 48, and 64 update batches (each containing at most 256 keyword–identifier pairs, without splitting a document across batches) are added to the cache; the one document that is returned from the cache is introduced in the first update batch. The resulting proof times are captured in Figure 5. Our results indicate that there is a difference of about 12 seconds in π_{mem} and total prover time 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 cache sizes.

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

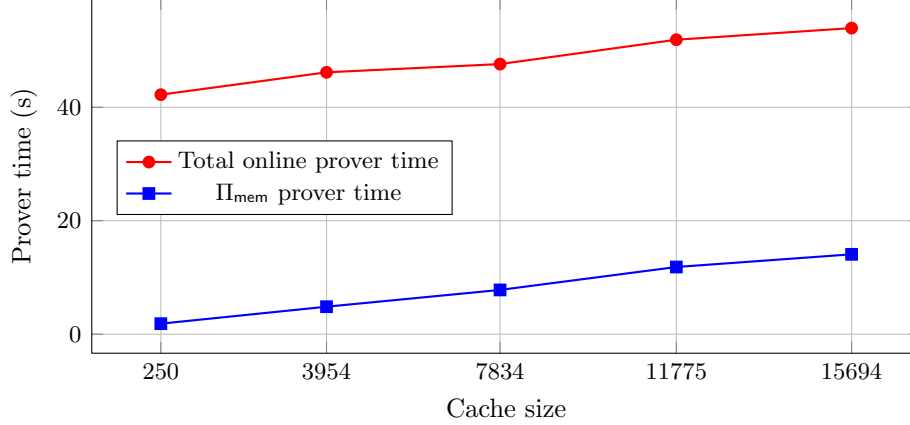


Figure 5: Prover times as a function of the cache size.

Threads	Rebase (hours)	Speedup	Peak Memory
1	3.5	1.00×	3.3GB
4	1.14	3.08×	4.2GB
8	0.88	4.00×	4.5GB

Table 2: Rebase performance with different numbers of threads.

note that the asymptotics of our protocol are actually quite good—online prover time is independent of the base size, allowing prover times to stay reasonably constant even as the number of documents continues to increase. Moreover, our other proof components are dependent on cache size and the number of keywords, both of which remain bounded as the total number of documents grows. Instead, the key determinant of long-term scalability will be rebasing, discussed below.

Audit phase. To evaluate the audit phase, we benchmark the runtime of adding $\delta = 7834$ keyword–identifier pairs into the cache over the course of 32 batches, using our initial base as the starting point. We measure recurring Π_{upd} proofs after initial Π_{state} . The reported update costs exclude the initial Π_{state} proof and are not the total cost of a standalone Audit invocation. We found that computing π_{upd} for each batch requires an average of 182.87 seconds, producing a 249.3 KB proof that could be verified in 2.04s. These figures can be significantly improved with parallelism; with the maximum parallelism available on our prover, prover time dropped to 41.0s and verification to 0.54s. This indicates that the overhead of regularly generating these update proofs can reasonably fit into everyday workflows. For example, documents might be accumulated over the course of the day or the week, and the agency might generate a single π_{upd} 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 the cost of this computation, we begin with a base of size $T_{\text{base}} = 838,846$, add 64 batches with a total of 15,694 keyword–identifier pairs, and then rebase. Table 2 shows rebase performance as a function of the number of threads used to recompute the preprocessing.

Strict single-thread rebasing takes 12,621.93s, or 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; the runtime of rebasing is quasilinear in the table size. This also makes clear that our approach can scale to larger sets of documents without interfering with everyday workflows. We note that if the rebasing costs do become too large, a prover can always maintain a small number of separate tables and simply generate proofs independently with respect to each table. While this induces a linear asymptotic factor, the linear term will grow exceedingly slowly and remain well within real-world performance requirements.

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] 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.
- [14] 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.
 - [15] 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.
 - [16] 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>.
 - [17] 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.
 - [18] 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>.
 - [19] Liam Eagen, Dario Fiore, and Ariel Gabizon. cq: Cached quotients for fast lookups. *Cryptology ePrint Archive*, 2022.
 - [20] 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.
 - [21] Andrés Fábrega, Jack Cable, Michael A Specter, and Sunoo Park. Cryptographic verifiability for voter registration systems. *arXiv preprint arXiv:2503.03974*, 2025.
 - [22] 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.
 - [23] 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.
 - [24] 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.
 - [25] 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>.
 - [26] 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.

- [27] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Annual International Cryptology Conference*, pages 33–62. Springer, 2018.
- [28] 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>.
- [29] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *Cryptology ePrint Archive*, 2019.
- [30] 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.
- [31] Google. Binary transparency. Available at <https://binary.transparency.dev/>.
- [32] Google. Trillian: an open-source verifiable log. Available at <https://transparency.dev/>.
- [33] 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.
- [34] 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.
- [35] 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.
- [36] Ulrich Haböck. Multivariate lookups based on logarithmic derivatives. *Cryptology ePrint Archive*, 2022.
- [37] 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>.
- [38] 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.
- [39] 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.
- [40] 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.

- [41] 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>.
- [42] 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.
- [43] 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.
- [44] 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.
- [45] Harjasleen Malvai, Francesca Falzon, Andrew Zitek-Estrada, Sarah Meiklejohn, and Joseph Bonneau. SoK: Cryptographic authenticated dictionaries. In *NDSS 2026*. The Internet Society, February 2026.
- [46] 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.
- [47] 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.
- [48] Ueli Maurer. Zero-knowledge proofs of knowledge for group homomorphisms. *Designs, Codes and Cryptography*, 77(2):663–676, 2015.
- [49] 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>.
- [50] 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.
- [51] 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.
- [52] 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.

- [53] 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.
- [54] 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.
- [55] 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.
- [56] 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>.
- [57] 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>.
- [58] 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.
- [59] ACLU SoCal. Court rules inglewood police department systematically violated california public records act. Available [here](#), November 2025.
- [60] ACLU SoCal. Aclu social v. inglewood. Available at <https://www.aclusocal.org/cases/aclu-social-v-ingewood/>, April 2026.
- [61] Sammy Sussman. Can police departments be trusted to release their own misconduct records? apparently not. Available [here.](#), October 2024.
- [62] 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=9xTDB33nszpxFqS5D70mBg==&system=prod>, September 2024.
- [63] 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.
- [64] 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.
- [65] 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.

- [66] 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>.
- [67] 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.
- [68] 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>.
- [69] 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 [38]. 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 [34]. For the sake of completeness, we recall how a polynomial commitment can be used as a vector commitment.

Evaluation domain. To commit to vectors of 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 $\mathbb{H}_d := \{1, \omega, \omega^2, \dots, \omega^{d-1}\}$. Thus, $\mathbb{H}_d$ is precisely the set of d -th roots of unity in $\mathbb{F}$, and its vanishing polynomial is $Z_{\mathbb{H}_d}(X) = X^d - 1$.

Encoding and commitment. Let $V = (V[1], \dots, V[d]) \in \mathbb{F}^d$. 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 .

Committing directly to f_V would not, in general, hide V . We therefore randomize it without changing its evaluations on $\mathbb{H}_d$. The committer samples $\rho_V \xleftarrow{\$} \mathbb{F}_{<b_{\text{hide}}}[X]$ uniformly at random and defines

$$\hat{f}_V(X) := f_V(X) + (X^d - 1)\rho_V(X).$$

Because $X^d - 1$ vanishes at every point in $\mathbb{H}_d$, for every $i \in [d]$ we have $\hat{f}_V(\omega^{i-1}) = f_V(\omega^{i-1}) = V[i]$. Thus, $\hat{f}_V$ is a randomized polynomial encoding of V that preserves all evaluations corresponding to vector entries. The parameter b_{hide} controls the amount of randomness and is chosen sufficiently large to preserve hiding even given any evaluations of $\hat{f}_V$ outside $\mathbb{H}_d$ that are revealed by the surrounding protocol.

The commitment algorithm outputs $\text{Com}_V := \text{KZG.Commit}(\hat{f}_V)$ and $\text{op}_V := (V, \rho_V)$. Given (V, ρ_V) , one can reconstruct f_V and $\hat{f}_V$, check the prescribed degree bounds, and recompute the commitment.

For a polynomial $\widehat{f}_V$ extracted by an argument, the vector is recovered by evaluation on $\mathbb{H}_d$. Its interpolation polynomial f_V then determines $\rho_V = \frac{\widehat{f}_V - f_V}{X^d - 1}$. The encoding degree bound ensures $\deg \rho_V < b_{\text{hide}}$, giving a valid opening.

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.

Hiding follows from the random polynomial ρ_V : adding the multiple $(X^d - 1)\rho_V(X)$ randomizes the committed polynomial while leaving all vector entries unchanged. For the choice of b_{hide} used by our protocols, this randomness is sufficient to hide V given the information revealed by the surrounding protocol.

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 valid randomized encoding of $aV_1 + bV_2$. Together with the linear homomorphism of KZG commitments, this gives the linear homomorphism required by Definition 1.

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

Our concrete instantiations of $\Pi_{\text{wf}}, \Pi_{\text{cnt}}, \Pi_{\text{sub}}, \Pi_{\text{mem}}$, and $\Pi_{\cap}$ build on zero-knowledge arguments for several common NP relations. In Section A.2.1, we first define these auxiliary relations and describe efficient proof systems for them. Then, in Sections A.2.2 to A.2.6, we show how to compose these building blocks into the five required arguments. These instantiations use the SRS and programmable ROM, with knowledge soundness analyzed against algebraic adversaries; see Section 4.2. All component parameters are derived from the same SRS as VC. Commitments shared by several arguments are generated once and used unchanged in their statements. The concrete simulators use a common simulation trapdoor for these commitments, their evaluations, and random oracle answers.

A.2.1 Auxiliary Relations and Arguments

We start by introducing some notation.

PLONK-style algebraic arguments. Because our vector commitments are instantiated using KZG *polynomial commitments* [38], several auxiliary relations introduced below reduce to proving that a committed constraint polynomial $G(X)$ vanishes over a public evaluation domain $\mathbb{H}_d$. We prove such statements using PLONK-style quotient-polynomial arguments [29]. Concretely, the prover commits to a quotient polynomial $Q(X)$ satisfying $G(X) = Q(X)Z_{\mathbb{H}_d}(X)$, where $Z_{\mathbb{H}_d}$ is the vanishing polynomial of $\mathbb{H}_d$, and the verifier checks this identity at a random challenge point derived via Fiat–Shamir [24].

Throughout this section, we use the term *PLONK-style* to refer to this quotient-polynomial methodology. For each application, we specify the corresponding relation and constraint polynomial explicitly.

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 $V = (v_1, \dots, v_r, 0, \dots, 0) \in \mathbb{F}^h$, 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.

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 $x_{\text{look}} := (\text{Com}_U, \text{Com}_V, d, r)$, and the witness is $w_{\text{look}} := (U, V, \vec{\sigma}, \text{op}_U, \text{op}_V)$, where $\vec{\sigma} = (\sigma_1, \dots, \sigma_r) \in [d]^r$ specifies the private lookup positions. The relation $\mathcal{R}_{\text{look}}$ 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 $\mathcal{R}_{\text{look}}$ using the exact zero-knowledge lookup argument **zkcq+** [12]. Its parameters are derived from our common SRS, following the parameter requirements of that construction. The prover commits to a polynomial encoding of V and internally derives the lookup multiplicities and blinded rational functions required by **zkcq+**. 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 $O(\log r)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $\text{priv} \subseteq \{U, V, a, b\}$ denote the vectors that are private. We define the statement as $x_{\text{logup}} := (\{\text{Com}_W\}_{W \in \text{priv}}, \{W\}_{W \notin \text{priv}}, r, s)$, and the witness as $w_{\text{logup}} := (\{W, \text{op}_W\}_{W \in \text{priv}})$.

The relation $\mathcal{R}_{\text{logup}}$ 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 $\mathcal{R}_{\text{logup}}$ using the LogUp argument **LogUp** [36], realizing its constraints with zero-knowledge polynomial arguments over our hiding KZG commitments. After all statement commitments are fixed, the prover and verifier derive a Fiat–Shamir challenge β outside the represented value domains and prove the corresponding rational identity at $X = \beta$.
- **Cost:** The prover cost is $O(r \log r + s \log s)_{\mathbb{F}} + O(r + s)_{\mathbb{G}}$, the verifier cost is $O(\log r + \log s)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $0 \leq T \leq d$ be public. The statement is $x_{\text{supp}}^{\text{pre}} := (\text{Com}_V, T, d)$, and the witness is $w_{\text{supp}}^{\text{pre}} := (V, \text{op}_V)$. The relation $\mathcal{R}_{\text{supp}}^{\text{pre}}$ requires $V[j] = 0$ for every $T < j \leq d$. 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 $T + B \leq d$. The statement is $x_{\text{supp}}^{\text{int}} := (\text{Com}_W, \text{Com}_U, T, B, d)$, and the witness is $w_{\text{supp}}^{\text{int}} := (W, U, \text{op}_W, \text{op}_U)$. The relation $\mathcal{R}_{\text{supp}}^{\text{int}}$ requires $W = 0^T \| U \| 0^{d-T-B}$. The vector $U \in \mathbb{F}^B$ may use a public encoding capacity $N_{\Delta} \geq B$, 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 [29] and the interval form using a generalized Schnorr representation argument [48], both over our hiding KZG commitments.

For the prefix form, let $L_j^{(d)}$ be the Lagrange basis polynomial for position j in $\mathbb{H}_d$, and define $M_T(X) := \sum_{j=T+1}^d L_j^{(d)}(X)$. 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

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

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. The opening of M_T authenticates its evaluation against this fixed public commitment.

For the interval form, we use a generalized Schnorr representation argument [48] for the two commitments. It proves that their randomized encodings have the form

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

using the same hidden values $u_1, \dots, u_B$ 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 domain and the profile for (d, T) have been preprocessed. Given the public domain material, constructing the profile for (d, T) costs $O(d \log d)_{\mathbb{F}} + O(d)_{\mathbb{G}}$. It can be reused while (d, T) is unchanged. The profile is derived from the SRS, rather than chosen by the prover. Given this preprocessing, the prover cost is $O(d \log d)_{\mathbb{F}} + O(d)_{\mathbb{G}}$, the verifier cost is $O(\log d)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 additionally reconstructs the masking data and re-derives the domain bases. 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 the masking data depend on the current commitment opening.

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 $x_{\text{pref}} := (\text{Com}_S, h)$, and the witness is $w_{\text{pref}} := (S, \text{op}_S)$. The relation $\mathcal{R}_{\text{pref}}$ requires that there exists $r \in \{0, \dots, h-1\}$ such that $S = (1^r, 0^{h-r})$.
- **Instantiation:** Let $\hat{f}_S$ denote the polynomial underlying the hiding KZG commitment to S . Define

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

The prover commits to quotient polynomials Q_{bool} and Q_{step} satisfying

$$\hat{f}_S(X)(\hat{f}_S(X) - 1) = Q_{\text{bool}}(X)Z_{<h-1}(X)$$

and

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

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

$$(1 - \hat{f}_S(X))\hat{f}_X(X) = Q_{\text{zpad}}(X)Z_{\mathbb{H}_h}(X)$$

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 $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $x_{\text{pad}} := (\text{Com}_X, \text{Com}_{S_X}, h)$, and the witness is $w_{\text{pad}} := (X, S_X, \text{op}_X, \text{op}_{S_X})$. The relation $\mathcal{R}_{\text{pad}}$ requires both $\mathcal{R}_{\text{pref}}$ and $\mathcal{R}_{\text{zpad}}$ to hold. Equivalently, there exists $r < h$ such that $S_X = (1^r, 0^{h-r})$ and $X[i] = 0$ for every $r < i \leq h$.
- **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 $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $a \leq b$ be public integers whose canonical embeddings in $\mathbb{F}$ are understood. The statement is $x_{\text{range}} := (\text{Com}_V, a, b, h)$, and the witness is $w_{\text{range}} := (V, \text{op}_V)$. The relation $\mathcal{R}_{\text{range}}$ requires $V[i] \in \{a, a+1, \dots, b\}$ for every $i \in [h]$.
- **Instantiation:** We instantiate $\mathcal{R}_{\text{range}}$ using the lookup argument $\text{zkcq} + [12]$, with the public table $(a, a+1, \dots, b)$.
- **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 $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $x_{\text{card}} := (\text{Com}_S, \text{Com}_V, t, h, d)$, and the witness is $w_{\text{card}} := (S, V, \text{op}_S, \text{op}_V)$. The relation $\mathcal{R}_{\text{card}}$ 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 proves $A[1] = 0$ and $A[j+1] - A[j] = S[j]$ for $j \in [h-1]$ using zero-knowledge PLONK-style polynomial checks [29] over hiding KZG commitments [38]. It then commits to a private singleton (c) and reuses the indexed-equality construction of Section A.2.4 at the public positions h and t to prove $(A + S)[h] = c = V[t]$, using the same singleton commitment in both checks and the derived commitment $\text{Com}_{A+S} := \text{Com}_A + \text{Com}_S$. The common value is kept private.

- **Cost:** For its use in an ℓ -keyword query, the cardinality encodings use $O(\ell)$ masking coefficients. Given prepared public domain bases, the prover cost is $O(h \log h + d \log d + \ell)_{\mathbb{F}} + O(h + d + \ell)_{\mathbb{G}}$, the verifier cost is $O(\log h + \log d + \ell)_{\mathbb{F}} + O(\ell)_{\mathbb{G}} + O(1)_{\text{pairing}}$, and the proof size is $O(\ell)_{\mathbb{F}} + O(1)_{\mathbb{G}}$. The d -dependent work includes constructing and binding the restriction of the fixed commitment Com_V .

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 $x_{\text{gp}}^{\text{eq}} := (\text{Com}_X, \text{Com}_Y, r)$, and the witness is $w_{\text{gp}}^{\text{eq}} := (X, Y, \text{op}_X, \text{op}_Y)$. The relation $\mathcal{R}_{\text{gp}}^{\text{eq}}$ 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 $x_{\text{gp}}^{\text{dec}} := (\text{Com}_X, \text{Com}_Y, \text{Com}_Z, r, s, t)$, and the witness is $w_{\text{gp}}^{\text{dec}} := (X, Y, Z, \text{op}_X, \text{op}_Y, \text{op}_Z)$. The relation $\mathcal{R}_{\text{gp}}^{\text{dec}}$ 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 [29]. After all relevant 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 corresponding product identities are proved using the standard PLONK grand-product construction.

- **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 $O(\log D)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, 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 $x_{\text{dist}} := (\text{Com}_V, h)$, and the witness is $w_{\text{dist}} := (V, \text{op}_V)$. The relation $\mathcal{R}_{\text{dist}}$ requires $V[i] \neq V[j]$ for every distinct $i, j \in [h]$.
- **Instantiation:** In the Online and Audit protocols, the entries of V lie in a certified integer range $[0, U]$, canonically embedded in $\mathbb{F}$, where $2U < |\mathbb{F}|$. We reuse the value bounds established by the surrounding arguments and instantiate $\mathcal{R}_{\text{dist}}$ using a sort-and-difference argument. The prover first commits to a vector V^{sort} containing the entries of V in increasing order and invokes Π_{gp} to prove $V \equiv V^{\text{sort}}$. It then commits to the adjacent-difference vector $D[i] := V^{\text{sort}}[i+1] - V^{\text{sort}}[i]$ for $1 \leq i < h$. A PLONK-style quotient argument proves these difference constraints, and Π_{range} proves that every entry of D lies in $[1, U]$. Under the stated bounds, the field differences agree with positive integer differences, so V^{sort} is strictly increasing and contains no repeated value.
- **Cost:** 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 $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, and the proof size is $O(1)_{\mathbb{F}, \mathbb{G}}$.

A.2.2 Protocol Π_{wf}

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

Prover. The prover proceeds as follows:

- Invokes Π_{supp} on $(\text{Com}_{E_{\text{key}}}, T)$ and $(\text{Com}_{E_{\text{doc}}}, T)$ to prove $E_{\text{key}}[j] = E_{\text{doc}}[j] = 0$ for every $T < j \leq N$.
- Invokes Π_{range} to prove $E_{\text{key}}[j] \in [m]$ and $E_{\text{doc}}[j] \in [n]$ for every $j \in [T]$.
- Sets $\eta := m + 1$ and defines $P := E_{\text{key}} + \eta E_{\text{doc}}$. Using linear homomorphism, computes $\text{Com}_P := \text{Com}_{E_{\text{key}}} + \eta \text{Com}_{E_{\text{doc}}}$. Invokes Π_{dist} on Com_P to prove that $P[j]$ for $j \in [T]$ are pairwise distinct.¹²
- Outputs the two support proofs, the two range proofs, and the distinctness proof.

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

Lemma 2 (Protocol Π_{wf}). *Assume that $2(m + (m + 1)n) < |\mathbb{F}|$. 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 $\mathcal{R}_{\text{wf}}$ 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 $P := E_{\text{key}} + (m + 1)E_{\text{doc}}$. Under the assumption $m + (m + 1)n < |\mathbb{F}|$, the map $(k, \text{id}) \mapsto k + (m + 1)\text{id}$ 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 $\mathcal{R}_{\text{wf}}$.

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 . $\square$

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 $O(\log N + \log m + \log n)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$. 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 $x_{\text{cnt}} := (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{cnt}}}, T)$, and the witness is $w_{\text{cnt}} := (E_{\text{key}}, E_{\text{cnt}}, \text{op}_{E_{\text{key}}}, \text{op}_{E_{\text{cnt}}})$. The protocol proves that $E_{\text{cnt}}[k]$ equals the number of occurrences of keyword k among the first T entries of E_{key} . In the concrete protocols, Π_{wf} certifies that the first T entries of E_{key} lie in $\mathcal{K}$. Π_{cnt} uses this certification. It proceeds as follows.

Prover. The prover proceeds as follows:

¹²The 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** [36] to prove

$$\sum_{j=1}^T \frac{1}{\beta - E_{\text{key}}[j]} = \sum_{k=1}^m \frac{E_{\text{cnt}}[k]}{\beta - k}.$$

- 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** [36] satisfies Definition 2 for $\mathcal{R}_{\text{logup}}$. For the certified invocations described above, Π_{cnt} is complete and zero knowledge. Its joint execution with Π_{wf} provides knowledge soundness for $\mathcal{R}_{\text{cnt}}$ against algebraic adversaries in the SRS and programmable ROM.*

Proof sketch. The well-formedness certification ensures that $E_{\text{key}}[j] \in \mathcal{K}$ 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, $E_{\text{cnt}}[k] = |\{j \in [T] : E_{\text{key}}[j] = k\}|$ for every $k \in \mathcal{K}$, which is the condition of $\mathcal{R}_{\text{cnt}}$.

Completeness in this certified context follows from the honest counts. Zero knowledge follows from the corresponding properties of the component arguments and hiding of **VC**. $\square$

Efficiency. The protocol invokes **LogUp** once, with public input capacities N and m , which gives $O(N \log N + m \log m)_{\mathbb{F}} + O(N + m)_{\mathbb{G}}$. The verifier cost is $O(\log N + \log m)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{pairing}}$, and the proof size is $O(1)_{\mathbb{F}, \mathbb{G}}$, which are the same as one **LogUp** invocation.

A.2.4 Protocol Π_{sub}

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

Prover. The prover proceeds as follows:

- Writes $\mathcal{Q} = \{q_1, \dots, q_{\ell}\}$ in canonical order.
- Invokes a PLONK-style argument [29] to prove $E_{\text{cnt}}^{\mathcal{Q}}[i] - E_{\text{cnt}}[q_i] = 0$ for every $i \in [\ell]$. The public query $\mathcal{Q}$ determines the wiring between position i of $E_{\text{cnt}}^{\mathcal{Q}}$ and position q_i of E_{cnt} , and the ℓ constraints are batched using a random linear combination. These equalities are proved with respect to the fixed commitments $\text{Com}_{E_{\text{cnt}}}$ and $\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}$, with the counts kept as private witnesses.
- 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 $\mathcal{R}_{\text{sub}}$ in the SRS and programmable ROM, with knowledge soundness against algebraic adversaries.*

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

Efficiency. Assuming the public evaluation domains and commitment bases have been preprocessed, the interpolation and quotient computations involving the count polynomial cost $O(m \log m)_{\mathbb{F}}$ using FFTs and IFFT. Let z_i be the public count-domain point corresponding to position q_i . The product tree constructs $Z_{\mathcal{Q}}(X) := \prod_{i=1}^{\ell} (X - z_i)$ and the interpolation data for the batched polynomial checks in $O(\ell \log^2 \ell)_{\mathbb{F}}$ operations, while the additional logarithmic factor comes from the $O(\log \ell)$ tree levels. Together with $O(m + \ell)_{\mathbb{G}}$ commitment generation cost, the prover cost is $O(m \log m + \ell \log^2 \ell)_{\mathbb{F}} + O(m + \ell)_{\mathbb{G}}$.

The verifier prepares only the public query-dependent polynomials and checks the batched proof, costing $O(\ell \log^2 \ell)_{\mathbb{F}} + O(\ell)_{\mathbb{G}} + O(1)_{\text{pairing}}$. 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 $x_{\text{mem}} := (\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, T, \mathcal{Q}, \text{Com}_{\text{aux}})$, where $\text{Com}_{\text{aux}} := (\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}, \{\text{Com}_{E_{\text{Enc}}^{q_i}}\}_{i \in [\ell]})$, and the witness is

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

The protocol proves that each query-specific vector $E_{\text{Enc}}^{q_i}$ contains distinct valid identifiers associated with q_i , with exactly $E_{\text{cnt}}^{\mathcal{Q}}[i]$ selected entries and zero padding elsewhere. In the complete online protocol, Π_{wf} certifies the same table commitments and public value T . The membership argument uses this certification and does not repeat the table well-formedness 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 $c_i := E_{\text{cnt}}^{\mathcal{Q}}[i]$, constructs $S_i = (1^{c_i}, 0^{h-c_i})$, commits to S_i , and invokes Π_{pad} on $(\text{Com}_{E_{\text{Enc}}^{q_i}}, \text{Com}_{S_i}, h)$ to prove that S_i identifies a prefix of $E_{\text{Enc}}^{q_i}$ and that every unselected entry is zero.
- Invokes Π_{card} to prove $\sum_{j=1}^h S_i[j] = E_{\text{cnt}}^{\mathcal{Q}}[i]$ for every $i \in [\ell]$. The prover collects these checks into one batched cardinality proof.
- For every $i \in [\ell]$, invokes Π_{range} to prove that every entry of $E_{\text{Enc}}^{q_i}$ selected by S_i lies in $[n]$, and invokes Π_{dist} to prove that these selected entries are pairwise distinct.
- After all relevant commitments are fixed, derives $\alpha \xleftarrow{\$} \mathbb{F} \setminus \{0\}$ via Fiat–Shamir. Defines $E^{(\alpha)} := E_{\text{key}} + \alpha E_{\text{doc}}$ and, for every $i \in [\ell]$, $A_i := q_i S_i + \alpha E_{\text{Enc}}^{q_i}$. Using linear homomorphism, derives

$$\text{Com}_{E^{(\alpha)}} := \text{Com}_{E_{\text{key}}} + \alpha \text{Com}_{E_{\text{doc}}}$$

and

$$\text{Com}_{A_i} := q_i \text{Com}_{S_i} + \alpha \text{Com}_{E_{\text{Enc}}^{q_i}}.$$

- For every $i \in [\ell]$, invokes **zkcq+** [12] for $\mathcal{R}_{\text{look}}$ to prove that every entry of A_i appears in the table $E^{(\alpha)}$.

At a selected position j , the lookup value is $q_i + \alpha E_{\text{Enc}}^{q_i}[j]$. At an unselected position, $S_i[j] = E_{\text{Enc}}^{q_i}[j] = 0$, 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 $\text{Com}_{E^{(\alpha)}}$ 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 $(\text{Com}_{E_{\text{key}}}, \text{Com}_{E_{\text{doc}}}, T)$ 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 $\mathcal{R}_{\text{look}}$. Then Π_{mem} described above satisfies Definition 2 for $\mathcal{R}_{\text{mem}}$ 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 $E_{\text{Enc}}^{q_i}$ has exactly $E_{\text{cnt}}^Q[i]$ 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 $\mathcal{R}_{\text{look}}$, every selected position j has a matching table position $k \in [N]$ such that $q_i + \alpha E_{\text{Enc}}^{q_i}[j] = E_{\text{key}}[k] + \alpha E_{\text{doc}}[k]$. Since all relevant commitments are fixed before α is derived, except with negligible probability this equality implies $E_{\text{key}}[k] = q_i$ and $E_{\text{doc}}[k] = E_{\text{Enc}}^{q_i}[j]$. 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 $\mathcal{R}_{\text{mem}}$.

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

Remark 7. *We apply the Base+Cache framework of Dutta et al. [17] to the table-dependent preprocessing used by Π_{mem} . A previously audited public commitment is selected as the Base, while the keyword–identifier pairs appended after that commitment 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 $\mathcal{R}_{\text{mem}}$ 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 vector E_{cnt}^Q is encoded over a public domain of size $\Theta(\ell)$. The masking polynomials used by Π_{card} have $O(\ell)$ coefficients. The batched cardinality proof links all ℓ selectors to the corresponding entries of E_{cnt}^Q , with respect to the fixed commitment $\text{Com}_{E_{\text{cnt}}^Q}$. Summing these costs gives the Base-only prover bound $O(\ell h \log h + \ell^2 \log \ell)_{\mathbb{F}} + O(\ell h + \ell^2)_{\mathbb{G}}$. 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 $O(\delta \log^2 \delta + \ell(\delta + h) \log^2(\delta + h))_{\mathbb{F}} + O(\ell(\delta + h))_{\mathbb{G}}$. Summing their verification costs gives $O(\ell \log h + \ell^2)_{\mathbb{F}} + O(\ell^2)_{\mathbb{G}}$ together with $O(\ell)$ pairing terms, accumulated into one final multi-pairing check. The proof size is $O(\ell^2)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$. The Cache does not change the verifier cost or proof size.

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

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

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. A standalone invocation includes these checks as prerequisites.

Prover. The prover proceeds as follows:

- Sets $\mathcal{I}_1 := \mathcal{E}_{\text{doc}}^{q_1}$.
- For every $i \in \{2, \dots, \ell\}$, constructs fixed-capacity padded encodings of the following hidden sets:
 1. $\mathcal{I}_i$, the accumulated intersection;
 2. $\mathcal{X}_i$, the entries of $\mathcal{I}_{i-1}$ omitted from $\mathcal{I}_i$;
 3. $\mathcal{Y}_i$, the entries of $\mathcal{E}_{\text{doc}}^{q_i}$ omitted from $\mathcal{I}_i$;
 4. $\mathcal{U}_i$, the multiset union of $\mathcal{X}_i$ and $\mathcal{Y}_i$.¹³

Commits to these vectors and their auxiliary selectors and invokes Π_{pad} on all four encodings.

- Invokes Π_{gp} to prove

$$\mathcal{I}_{i-1} \equiv \mathcal{I}_i \uplus \mathcal{X}_i, \quad \mathcal{E}_{\text{doc}}^{q_i} \equiv \mathcal{I}_i \uplus \mathcal{Y}_i, \quad \mathcal{U}_i \equiv \mathcal{X}_i \uplus \mathcal{Y}_i.$$

- Invokes Π_{dist} on the encoding of $\mathcal{U}_i$. Since $\mathcal{U}_i$ represents the multiset union of $\mathcal{X}_i$ and $\mathcal{Y}_i$, distinctness of $\mathcal{U}_i$ implies $\mathcal{X}_i \cap \mathcal{Y}_i = \emptyset$.
- After the final round, invokes Π_{gp} to prove $\mathcal{I}_{\ell} \equiv \mathcal{J}$. When $\ell = 1$, directly proves $\mathcal{E}_{\text{doc}}^{q_1} \equiv \mathcal{J}$.
- Outputs all intermediate commitments and subproofs.

Verifier. The verifier checks that commitments are reused consistently across successive rounds, verifies all subproofs, and accepts if and only if all of them accept.

Lemma 6 (Protocol $\Pi_{\cap}$). *Assume that VC satisfies Definition 1, that the input query-specific vectors have been certified by Π_{mem} , and that Π_{pad} , Π_{gp} , and Π_{dist} satisfy Definition 2 for their respective relations. 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. Completeness follows from the honest intersections and residual sets.

For knowledge soundness, the first two decomposition relations in round i imply $\mathcal{I}_i \subseteq \mathcal{I}_{i-1} \cap \mathcal{E}_{\text{doc}}^{q_i}$. The third decomposition relation and distinctness of $\mathcal{U}_i$ imply $\mathcal{X}_i \cap \mathcal{Y}_i = \emptyset$. Suppose an identifier

¹³The encodings of $\mathcal{I}_i$, $\mathcal{X}_i$, and $\mathcal{Y}_i$ have capacity h , while the encoding of $\mathcal{U}_i$ has capacity $2h$.

belongs to both $\mathcal{I}_{i-1}$ and $\mathcal{E}_{\text{doc}}^{q_i}$ but is omitted from $\mathcal{I}_i$. The first two decomposition relations then place it in both $\mathcal{X}_i$ and $\mathcal{Y}_i$, contradicting their disjointness. Hence $\mathcal{I}_i = \mathcal{I}_{i-1} \cap \mathcal{E}_{\text{doc}}^{q_i}$. Inducting over the rounds and applying the final multiset equality gives $\mathcal{J} = \bigcap_{i=1}^{\ell} \mathcal{E}_{\text{doc}}^{q_i}$, which is exactly $\mathcal{R}_{\cap}$.

Zero knowledge follows from zero knowledge of Π_{pad} , Π_{gp} , and Π_{dist} , together with hiding of VC. $\square$

Efficiency. Each of the $\ell - 1$ intersection rounds invokes Π_{pad} four times, Π_{gp} three times, and Π_{dist} once. The vectors have capacities h or $2h$, so each invocation costs $O(h \log h)_{\mathbb{F}} + O(h)_{\mathbb{G}}$. Including the final Π_{gp} against $\mathcal{J}$, the total prover cost is $O(\ell h \log h)_{\mathbb{F}} + O(\ell h)_{\mathbb{G}}$.

For $\ell = 1$, only the final Π_{gp} is needed, and the total prover cost is also $O(\ell h \log h)_{\mathbb{F}} + O(\ell h)_{\mathbb{G}}$.

Each round contributes $O(\log h)_{\mathbb{F}} + O(1)_{\mathbb{G}}$ verification cost and a constant number of pairing terms. Including the final check against $\mathcal{J}$ gives $O(\ell \log h + |\mathcal{J}|)_{\mathbb{F}} + O(\ell + |\mathcal{J}|)_{\mathbb{G}}$, 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 $O(\ell)_{\mathbb{F}, \mathbb{G}}$.

A.3 Cost Analysis

Recall that N is the capacity of the keyword-identifier table, and $m := |\mathcal{K}|$, $\ell := |\mathcal{Q}|$, and $a := |\mathcal{J}|$. Let h be the public capacity of each query-specific padded identifier vector. The vector $E_{\text{cnt}}^{\mathcal{Q}}$ is encoded over a public domain of size $\Theta(\ell)$, and the masking polynomials used by Π_{card} have $O(\ell)$ coefficients. We substitute these bounds directly into the costs below. The bounds exclude one-time setup and reusable preprocessing of public evaluation domains and lookup tables. Proof size includes all proof-generated commitments but excludes com , $\mathcal{Q}$, and $\mathcal{J}$.

Reusable query-independent cost. The reusable query-independent proof consists of 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 $O(N \log N + m \log m)_{\mathbb{F}} + O(N + m)_{\mathbb{G}}$. Summing these costs gives

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

The $N \log N$ term comes from processing the committed length- N vectors, while the $m \log m$ term comes from the public keyword domain used by Π_{cnt} . Combining the two component verifiers gives $V_{\text{query-ind}} = O(\log N + \log m + \log n)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{multi-pairing}}$. 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 consists of Π_{sub} , Π_{mem} , and $\Pi_{\cap}$, together with the query-dependent commitments.

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

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

The $m \log m$ and $\ell \log^2 \ell$ terms come from proving $E_{\text{cnt}}^{\mathcal{Q}}[i] = E_{\text{cnt}}[q_i]$ for every $i \in [\ell]$, with respect to $\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}$ and $\text{Com}_{E_{\text{cnt}}}$. The $\ell h \log h$ term comes from the padded identifier vectors, their cardinality checks, and their intersection. The $\ell^2 \log \ell$ term comes from the ℓ cardinality checks against $\text{Com}_{E_{\text{cnt}}^{\mathcal{Q}}}$, each processing the encoding of $E_{\text{cnt}}^{\mathcal{Q}}$ over a public domain of size $\Theta(\ell)$.

Combining the three sub-verifiers gives $V_{\text{query}} = O(\ell \log^2 \ell + \ell \log h + \ell^2 + a)_{\mathbb{F}} + O(\ell^2 + a)_{\mathbb{G}} + O(1)_{\text{multi-pairing}}$.

Including the query-dependent commitments, the per-query proof contains $O(\ell)$ group elements and $O(\ell^2)$ field elements. Its size is therefore $O(\ell^2)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$.

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

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

The complete per-query prover cost is therefore

$$P_{\text{query}}^{\text{cache}} = P_{\text{query}} + P_{\text{cache}}.$$

The Cache-dependent overhead is described in Remark 7. 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

$$P_{\text{online}}^{\text{stand}} = O(N \log N + m \log m + \ell \log^2 \ell + \ell h \log h + \ell^2 \log \ell)_{\mathbb{F}} + O(N + m + \ell h + \ell^2)_{\mathbb{G}}.$$

Combining all five component verifiers gives

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

The reusable query-independent proof has constant size. Including the per-query proof, the total proof size is $O(\ell^2)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$.

For a nonempty Cache of public size δ , the complete standalone prover cost is $P_{\text{online}}^{\text{stand,cache}} = P_{\text{online}}^{\text{stand}} + P_{\text{cache}}$. The Cache does not change the standalone verifier cost or proof size.

Including the public query and response, total standalone communication is $O(\ell^2 + a)_{\mathbb{F}} + O(\ell)_{\mathbb{G}}$.

B Additional Details for Our Audit Phase Protocol

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

We now present our concrete instantiations of Π_{state} and Π_{upd} . We reuse the vector commitment of Section A.1 and the auxiliary arguments of Section A.2.1. In particular, protocols for Π_{state} and Π_{upd} will use Π_{pad} , Π_{supp} , Π_{zpad} , Π_{range} , Π_{dist} , Π_{gp} , and LogUp [36]. Section B.1.1 defines the auxiliary argument Π_{id} used by Π_{state} . Then, in Sections B.1.2 and B.1.3, we show how to compose these building blocks to design Π_{state} and Π_{upd} .

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.

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 $\text{Com}_{\mathcal{ID}_{\mathcal{D}}} := (\text{Com}_I, \text{Com}_{S_I})$. 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 $E_{\text{doc}} \in \mathbb{F}^N$ and $I, S_I \in \mathbb{F}^{h_D}$. The statement is $x_{\text{id}} := (\text{Com}_{E_{\text{doc}}}, T, \text{Com}_I, \text{Com}_{S_I})$, and the witness is $w_{\text{id}} := (E_{\text{doc}}, I, S_I, \text{op}_{E_{\text{doc}}}, \text{op}_I, \text{op}_{S_I})$. The relation $\mathcal{R}_{\text{id}}$ requires that (I, S_I) encode exactly the set $\{E_{\text{doc}}[j] \mid j \in [T]\}$.
- **Instantiation:** This instantiation is used together with a Π_{wf} certification of the keyword-identifier representation, sharing the same document commitment 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 , $\mu[i] := |\{j \in [T] : E_{\text{doc}}[j] = I[i]\}|$, and $\mu[i] = 0$ at every unselected position. For the honest identifier encoding, every selected multiplicity is positive. The well-formedness certification ensures that no identifier occurs with the same keyword twice, 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 $\gamma \xleftarrow{\$} \mathbb{F} \setminus (\{0\} \cup [n])$ via Fiat-Shamir and invoke **LogUp** [36] to prove

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

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 $O(\log N + \log h_D + \log n + \log m)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{multi-pairing}}$, 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** [36] satisfies Definition 2 for $\mathcal{R}_{\text{logup}}$. Assume also the guarantees of Π_{wf} in Lemma 2. For the certified invocations described above, Π_{id} described above is complete and zero knowledge, and its joint execution with Π_{wf} provides knowledge soundness for $\mathcal{R}_{\text{id}}$ 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** [36] 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 $\{E_{\text{doc}}[j] \mid j \in [T]\}$. Thus, $\mathcal{R}_{\text{id}}$ holds.

Zero knowledge follows from zero knowledge of the component arguments together with hiding of VC. $\square$

B.1.2 Protocol Π_{state}

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

Prover. The prover proceeds as follows:

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

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 $\mathcal{R}_{\text{state}}$ 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 $\mathcal{R}_{\text{state}}$, 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 $\mathcal{R}_{\text{state}}$.

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 $O(N \log N + m \log m)_{\mathbb{F}} + O(N + m)_{\mathbb{G}}$, 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 $O(N \log N + m \log m + h_D \log h_D)_{\mathbb{F}} + O(N + m + h_D)_{\mathbb{G}}$. Summing the three component verifier costs gives $O(\log N + \log m + \log n + \log h_D)_{\mathbb{F}} + O(1)_{\mathbb{G}} + O(1)_{\text{multi-pairing}}$. 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 $x_{\text{upd}} := (\text{com}, \text{com}^+, \text{Com}_{\text{aux}}^{\text{audit}})$, where

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

The witness is w_{upd} specified by $\mathcal{R}_{\text{upd}}$. 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 $x_{\text{state}} := (\text{com}, \text{Com}_{\mathcal{ID}_{\mathcal{D}}})$. Let N_{Δ} be the supported encoding capacity fixed by the public parameters, with $B \leq N_{\Delta}$. 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 $E_{\text{key}}^{\Delta}, E_{\text{doc}}^{\Delta} \in \mathbb{F}^B$, their count vector $E_{\text{cnt}}^{\Delta} \in \mathbb{F}^m$, 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_{key}^{Δ} and E_{doc}^{Δ} , represents $\mathcal{ID}_{\Delta}$ by a padded identifier vector I^{Δ} and selector S_I^{Δ} , and commits to both. Defines $\text{Com}_{\mathcal{ID}_{\Delta}} := (\text{Com}_{I^{\Delta}}, \text{Com}_{S_I^{\Delta}})$.
- Using linear homomorphism, derives $\text{Com}_{E_{\text{cnt}}^{\Delta}} := \text{Com}_{E_{\text{cnt}}^+} - \text{Com}_{E_{\text{cnt}}}$ and the corresponding opening. Defines $\text{com}_{\Delta} := (\text{Com}_{E_{\text{key}}^{\Delta}}, \text{Com}_{E_{\text{doc}}^{\Delta}}, \text{Com}_{E_{\text{cnt}}^{\Delta}}, B)$ and $x_{\text{state}}^{\Delta} := (\text{com}_{\Delta}, \text{Com}_{\mathcal{ID}_{\Delta}})$. Invokes Π_{state} on $x_{\text{state}}^{\Delta}$ to obtain $\pi_{\text{state}}^{\Delta}$.
- Invokes Π_{pad} and Π_{dist} on $\text{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$, and invokes Π_{gp} on $\text{Com}_{\mathcal{ID}_{\mathcal{D}}}$, $\text{Com}_{\mathcal{ID}_{\Delta}}$, and $\text{Com}_{\mathcal{ID}_{\mathcal{D}^+}}$ to prove $\mathcal{ID}_{\mathcal{D}^+} \equiv \mathcal{ID}_{\mathcal{D}} \uplus \mathcal{ID}_{\Delta}$. Denote these proofs collectively by π_{union} .
- For each $V \in \{E_{\text{key}}, E_{\text{doc}}\}$, let V^+ denote the corresponding vector committed by com^+ and V^{Δ} the corresponding appended vector. Using linear homomorphism of VC, derives $\text{Com}_{V^+ - V} := \text{Com}_{V^+} - \text{Com}_V$. Invokes the interval form of Π_{supp} on $(\text{Com}_{V^+ - V}, \text{Com}_{V^{\Delta}}, T, B, N)$ to prove $V^+ - V = 0^T \| V^{\Delta} \| 0^{N-T^+}$. Denotes the resulting proofs by $\pi_{\text{supp}}^{\text{key}, \Delta}$ and $\pi_{\text{supp}}^{\text{doc}, \Delta}$.
- Outputs $\pi_{\text{upd}} := (\text{Com}_{E_{\text{key}}^{\Delta}}, \text{Com}_{E_{\text{doc}}^{\Delta}}, \text{Com}_{\mathcal{ID}_{\Delta}}, \pi_{\text{state}}^{\Delta}, \pi_{\text{union}}, \pi_{\text{supp}}^{\text{key}, \Delta}, \pi_{\text{supp}}^{\text{doc}, \Delta})$.

Verifier. The verifier derives $\text{Com}_{E_{\text{cnt}}^{\Delta}} := \text{Com}_{E_{\text{cnt}}^+} - \text{Com}_{E_{\text{cnt}}}$ and, for each $V \in \{E_{\text{key}}, E_{\text{doc}}\}$, $\text{Com}_{V^+ - V} := \text{Com}_{V^+} - \text{Com}_V$. It then reconstructs the sub-statements above, verifies $\pi_{\text{state}}^{\Delta}$, 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 $x_{\text{state}} = (\text{com}, \text{Com}_{\mathcal{ID}_{\mathcal{D}}})$ has been certified by an accepted Π_{state} proof. Assume further that Π_{state} satisfies Definition 2 for $\mathcal{R}_{\text{state}}$, 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 for these certified invocations. Its joint execution with Π_{state} provides knowledge soundness for $\mathcal{R}_{\text{upd}}$ against algebraic adversaries in the SRS and programmable ROM.*

Proof sketch. Completeness follows from the honest append construction.

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

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 $V \in \{E_{\text{key}}, E_{\text{doc}}\}$, 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 $T < j \leq N$. Conse-

quently,

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

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

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

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

Efficiency. Recall that $B := T^+ - T$. The prover invokes Π_{state} once for the appended rows, with capacities $N_\Delta \geq B$, m , and h_Δ , which costs $O(N_\Delta \log N_\Delta + m \log m + h_\Delta \log h_\Delta)_\mathbb{F} + O(N_\Delta + m + h_\Delta)_\mathbb{G}$. The prover then invokes Π_{pad} , Π_{dist} , and Π_{gp} on the identifier set commitment. Their total input capacity is $O(h_D + h_\Delta)$, giving an additional $O(h_D \log h_D + h_\Delta \log h_\Delta)_\mathbb{F} + O(h_D + h_\Delta)_\mathbb{G}$. Finally, the two Π_{supp} invocations each cost $O(N_\Delta \log N_\Delta + B)_\mathbb{F} + O(N_\Delta \log N_\Delta + B)_\mathbb{G}$. Adding these contributions gives $O(h_D \log h_D + h_\Delta \log h_\Delta + m \log m + N_\Delta \log N_\Delta + B)_\mathbb{F} + O(h_D + h_\Delta + m + N_\Delta \log N_\Delta + B)_\mathbb{G}$.

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 $O(B + \log N_\Delta + \log m + \log n + \log h_D + \log h_\Delta)_\mathbb{F} + O(B)_\mathbb{G} + O(1)_{\text{multi-pairing}}$. Only the interval-support proofs contribute a B -dependent proof length, so the total proof size is $O(B)_\mathbb{F} + O(1)_\mathbb{G}$.

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 proof-generated commitments, but excludes the public commitments com and com^+ .

Initial public-commitment cost. The initial public-commitment proof consists of 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 $O(N \log N + m \log m)_\mathbb{F} + O(N + m)_\mathbb{G}$, 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

$$P_{\text{pub}} = O(N \log N + m \log m + h_D \log h_D)_\mathbb{F} + O(N + m + h_D)_\mathbb{G}.$$

The $N \log N$ term comes from processing the committed length- N keyword and identifier vectors, the $m \log m$ term comes from the public keyword domain used by Π_{cnt} , and the $h_D \log h_D$ term comes from validating the padded identifier encoding and its multiplicity vector.

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

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. The update proof contains an internal public-commitment proof for the appended rows, an identifier-set transition proof, and two interval-support proofs binding the appended keyword and identifier vectors to com and com^+ .

The internal public-commitment proof processes the capacity- N_Δ encodings of the B appended pairs, the public keyword domain of length m , and the appended identifier encoding of capacity h_Δ . The identifier-set transition 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

$$P_{\text{upd}}(B) = O(h_D \log h_D + h_\Delta \log h_\Delta + m \log m + N_\Delta \log N_\Delta + B)_{\mathbb{F}} + O(h_D + h_\Delta + m + N_\Delta \log N_\Delta + B)_{\mathbb{G}}.$$

Combining the component verifiers gives $V_{\text{upd}}(B) = O(B + \log N_\Delta + \log m + \log n + \log h_D + \log h_\Delta)_{\mathbb{F}} + O(B)_{\mathbb{G}} + O(1)_{\text{multi-pairing}}$. The linear terms in B come from the two interval-support checks, while the logarithmic terms come from the internally certified count and identifier encodings.

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 $\text{Com}_{\text{aux}}^{\text{audit}}$, the initial public-commitment proof π_{state} , and the update proof π_{upd} . Therefore, its prover cost is the sum of P_{pub} and $P_{\text{upd}}(B)$:

$$\begin{aligned} P_{\text{audit}}^{\text{stand}} &= O(N \log N + m \log m + h_D \log h_D + h_\Delta \log h_\Delta + N_\Delta \log N_\Delta + B)_{\mathbb{F}} \\ &\quad + O(N + m + h_D + h_\Delta + N_\Delta \log N_\Delta + B)_{\mathbb{G}}. \end{aligned}$$

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

The two identifier-set commitments and the initial public-commitment proof 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 $B_1, \dots, B_s$. The initial public-commitment proof and its identifier-set commitment 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

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

The total verifier cost is $V_{\text{audit}}^{\text{seq}} = V_{\text{pub}} + \sum_{r=1}^s V_{\text{upd}}(B_r)$. 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

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

and

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

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 $M_{\text{rebase}} = O(N \log N + mh \log h)_{\mathbb{F}} + O(N \log N + mh)_{\mathbb{G}}$.

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 fixed-shape queries per update, 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 $O(\sqrt{BN} \log^{3/2} N)_{\mathbb{F}}$.