Private Identity-based Bulletin Boards for Anonymous Messaging and Other Online Services (Full Version)

Karim Eldefrawy ⋆†, Stanislaw Jarecki ‡, Benjamin Terner † ⋆, Gene Tsudik ‡

⋆ SRI International, ‡ UC Irvine, † Confidencial.io

Abstract. Secure and anonymous messaging has many compelling use-cases and is becoming increasingly popular. In this paper, we consider it in the context of delay-and-disruption-prone networks, which are characterized by handicapped network access, disrupted operation, censorship, and intermittent network outages.

With such settings in mind, we define and design a Private Identity-Based Bulletin Board ($\mathcal{PIB}^3$) scheme, which allows users to anonymously post and retrieve messages to and from a distributed database, and supports communication between users without pre-established setup or pre-exchanged keys. Anyone can encrypt a message for an identity and public epoch, such that only the party with the decryption key for that identity can identify, retrieve, and decrypt the message. Against one corrupted non-colluding $\mathcal{PIB}^3$server, the server learns neither the recipient identity nor the retrieved record indices beyond the leakage explicitly modeled by the scheme: the public epoch, the database size, and the number of retrievals made by the receiver. If retrieval-count privacy is required, retrievals can be padded to a fixed bound. The multi-server construction extends this guarantee to larger server sets, and gives coalition privacy whenever the underlying multi-server PIR scheme is private against the corresponding coalition.

Contributions of this work are: (1) formally defining functionality and security requirements for $\mathcal{PIB}^3$-s, (2) defining and constructing a Hierarchical Identity-based Encryption (HIBE) scheme with searchable ciphertexts, which serves as a building block for the proposed $\mathcal{PIB}^3$ scheme and may be of independent interest, (3) designing an efficient $\mathcal{PIB}^3$ scheme that can be realized with $n \geq 2$ servers based on the HIBE scheme with searchable ciphertexts combined with additional primitives, and (4) implementing a functional $\mathcal{PIB}^3$ prototype which demonstrates practicality of the entire concept and allows us to assess its performance empirically.

1 Introduction

End-to-end secure messaging systems and other anonymity- and privacy-enhancing online services are increasingly popular, as evidenced by widespread adoption of

⋆ Work performed partially while at UC Irvine and at SRI International.

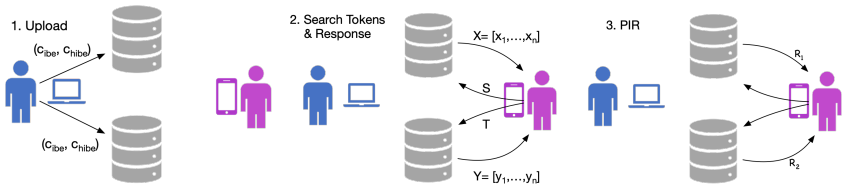
applications such as Signal [6, 18], Telegram [5], Openchat [1], and Tor [22]. Although secure messaging protects conversation *contents*, it does not typically protect conversation metadata, including the identities of senders and receivers and the fact that a message was sent. A recent line of work constructs scalable metadata-private messaging systems, at the expense of either continuous user participation to populate mix networks [32, 33] or weaker security guarantees [20, 17]. However, such techniques still either depend on reliable network connectivity, or do not provide anonymity against network adversaries or the servers that store and distribute messages to online/offline parties.

This paper presents a scheme for anonymous communication over unreliable and heavily censored networks, where access and backbone links may be censored, rate limited, or shut down by operators or governments [8, 39, 26, 27].This addresses shortcomings of prior secure messaging and metadata-private communication techniques, which can fail under poor network conditions or provide weaker security definitions for the identity-discovery setting considered here. The scheme is a *Private Identity-Based Bulletin Board* ($\mathcal{PIB}^3$). $\mathcal{PIB}^3$allows any sender, knowing only a receiver identity such as an email address or pseudonym, to post an encrypted record to non-colluding servers and then for the receiver to retrieve such record later. The receiver securely discovers records addressed to its identity and a public epoch, and retrieves the corresponding messages without revealing either its identity or the retrieved indices to any corrupted non-colluding bulletin-board server, except through explicitly modeled leakage. The leakage includes public epochs, database size, and retrieval counts in the unpadded version of the protocol.

Our construction, denoted PIB, uses two identity-based cryptographic components [40]. The first is a ciphertext-anonymous identity-based encryption (IBE) scheme, used to protect payload messages. The second is a two-level Hierarchical Identity-Based Encryption (HIBE) scheme with searchable ciphertexts, derived from Abdalla et al. [3], used to discover records addressed to an identity and epoch. We show how to compose these two building blocks with private information retrieval (PIR) to implement a bulletin board that supports asynchronous communication in disruption-prone networks. Our prototype shows the practicality of the bulletin board for databases with tens of thousands of records; for example, for 100kb payloads and 100,000 stored records, retrieval latency is just over 80 seconds.

1.1 Bulletin Boards for Communication in Adversarial Networks

A *bulletin board* allows parties to communicate without peer-to-peer, reliable channels. In our setting, parties may not have pairwise end-to-end secure channels and may not be online at the same time. Instead, they indirectly communicate by posting messages to a public service called a bulletin board. The bulletin board temporarily stores messages posted to it; it serves as privacy-preserving ephemeral "in-network storage" for users subject to network disruption, cen-



(a) The left client uploads a message to the PIB servers.

(b) The right client sends search tokens to the PIB servers and receives their responses.

(c) The right client computes a set of indices on which to execute PIR.

Fig. 1: Two clients, who have never met and have no pairwise secure channels but do know pseudonyms for each other, communicate via bulletin board servers. The clients can asynchronously interact with the servers, when connectivity allows.

sorship, or intermittent access.[1] Receivers asynchronously query bulletin boards to retrieve messages that have been posted for them. By providing "in-network storage," this construction provides resilience to communicating parties who may be subject to network disruptions or shut-downs.

Figure 1 shows the basic operation. A sender C wishing to send message m to receiver R with identity id uploads an encrypted record for a public epoch t. Later, R uses its identity-based secret key to discover and retrieve the records addressed to id in epoch t. We assume that clients communicate with the bulletin-board servers over anonymous channels, such as Tor [22], so that the servers do not learn the network identity of the uploading or querying client.

1.2 Design of an Identity-Based Bulletin Board

A private bulletin board should hide message contents and communication metadata from the host servers, beyond explicit modeled intended leakage such as public epochs, database size, and unpadded retrieval counts. An *identity-based* bulletin board allows senders and receivers to communicate even if they only know each other's identities, as in identity-based encryption [3, 10, 14, 15, 23, 40].

The main challenge is that receivers do not know which database records belong to them. This distinguishes our setting from standard PIR. In PIR, a client starts with an index and retrieves it privately. In our setting, the receiver must first discover which records were encrypted to its identity and epoch, and then retrieve those records privately. A naive solution is for the receiver to download the entire database and attempt to decrypt every record, which is private but inefficient, similarly to symmetric PIR-style approaches [25, 30].

The heart of the scheme is how the sender C uploads a message that can only be identified and decrypted by the reciever R. Our construction stores

[1] The storage is implemented at the application layer, but from the user's perspective it behaves like storage available somewhere in the network.

each record as two ciphertexts. The first ciphertext, c_{ibe}, is an IBE encryption of the payload under the public key derived from the receiver identity id. The second ciphertext, c_{hibe}, is a searchable HIBE ciphertext for the identity vector (id, t). The receiver uses its secret key to create a search token for epoch t, identifies matching records, and then uses PIR to retrieve the corresponding IBE ciphertexts. The PIR calls can be batched using standard techniques [4, 7, 29, 35]. The retrieval protocol requires n bilinear pairings per server, n equality checks by the receiver, and one decryption by the receiver, where n is the number of records held by the bulletin board, in addition to the cost of PIR for the identified records. We discuss this design in Section 2.2 and present the HIBE searchable-ciphertext construction formally in Section 6.

The bulletin board is temporal, each record is associated with a public epoch. The epoch duration is a system parameter, for example an hour or a day, and the board may purge records older than a public retention window. The searchable ciphertext does not hide the epoch from the bulletin-board servers. Accordingly, our privacy guarantee hides the receiver identity within a fixed public epoch, while treating the epoch, database size, and unpadded retrieval count as explicit leakage. If an application must also hide the retrieval count, the receiver can pad retrievals to a fixed bound and specify overflow behavior.

1.3 Contributions

This work makes four contributions.
(1) We define the syntax, correctness, and privacy requirements for a Private Identity-Based Bulletin Board ($\mathcal{PIB}^3$), and show how it supports secure anonymous messaging over delay- and disruption-prone networks.
(2) We define and construct a two-level HIBE scheme with searchable ciphertexts. The construction supports public-key record discovery, hides the recipient identity within a fixed public epoch, and may be of independent interest.
(3) We construct PIB, an efficient two-server $\mathcal{PIB}^3$ scheme, from ciphertext-anonymous IBE, HIBE searchable ciphertexts, and PIR. The construction generalizes to n servers, as shown in Appendix I; in the two-server case, the optimized retrieval communication is $O((n+\lambda)\log(n)+C+G)$, where n is database size, λ is the security parameter, C is ciphertext size, and G is the group element size. Communication optimizations are discussed in Appendix A.
(4) We implement a prototype of the two-server $\mathcal{PIB}^3$ construction and evaluate its performance in Appendix B. For small messages or small databases, retrieval latency is measured in seconds; for 100kb messages and a database of $100,000$ records, retrieval latency is just over 80 seconds.

1.4 Outline

Section 2 gives a technical overview. Section 3 and appendix C discuss related work. Section 4 and appendix D reviews preliminaries. Sections 5 and 6 and appendix E define and construct HIBE with searchable ciphertexts. Sections 7 and 8

define private bulletin boards and instantiate $\mathcal{PIB}^3$. Appendix A presents communication optimizations. Appendix B presents experiments and performance results. Appendix F discusses a single-server leakage attack that motivates the two-server design. Appendices G and H contain proofs of the cryptographic schemes. Appendix I presents the generalization of the protocol to n servers.

2 Technical Overview

In this section we outline the $\mathcal{PIB}^3$. We describe the operational flow of a bulletin board and identify the technical tools needed to realize the scheme.

2.1 System and Adversary Model

We consider a delay-prone and disruption-prone network under surveillance by a capable adversary (read state-level capabilities). Senders and receivers know pseudonyms for one another and have downloaded a messaging client containing public cryptographic material, such as IBE public parameters. The network may be censored or intermittently unavailable; users may rely on peer-to-peer store-and-forward applications such as FireChat or Bridgefy [8], or on delay-tolerant networking infrastructure, until they regain connectivity. The bulletin board acts as temporary identity-based storage, senders upload records for receivers, and receivers later retrieve their records when connectivity permits.

Clients communicate with the bulletin-board servers over anonymous channels, so the servers do not learn the network identity of the client issuing an upload or query. The network adversary may eavesdrop on links and disrupt connectivity, but it is polynomial time and cannot break the cryptographic assumptions used by the scheme. At the bulletin-board layer, the adversary may corrupt one server, control malicious uploading clients, adaptively generate records, and participate in retrieval protocols as the corrupted server.

The corrupted server learns its local protocol view, the public epoch, the database size, and the number of PIR retrievals made during a query; these values are explicit leakage. Active probing attacks that change the number of matches for a target identity and epoch are modeled up to this retrieval-count leakage. If an application requires hiding this count, the protocol can be padded to a fixed bound B, using dummy retrievals when $|I| < B$ and specifying an overflow policy for $|I| > B$.

2.2 $\mathcal{PIB}^3$ Design

We assume that arbitrary clients upload encrypted messages to the bulletin board through anonymous channels. Each stored record contains two ciphertext components:

1. an IBE ciphertext c_{ibe} that protects the payload message, and
2. a HIBE searchable ciphertext c_{hibe} generated for an identity vector (id, t), where id is the receiver identity and t is a public epoch.

The HIBE searchable ciphertext is used only for discovery, not for message confidentiality. It is obtained by extending the usual HIBE syntax with three searchable-ciphertext algorithms. The algorithm $\mathsf{CipherGen}$ creates a searchable ciphertext for an identity vector (id, t); Token uses the receiver's first-level secret key $\mathsf{sk}_{\mathsf{id}}$ to create a search token for (id, t); and Check tests whether a searchable ciphertext and search token correspond to the same identity vector. Thus, a receiver holding the secret key for identity id can generate a token for a public epoch t and test whether a record was generated for (id, t).

This is the primitive that lets senders post using only a receiver identity and epoch, while letting receivers later discover their records without pre-established pairwise keys. The construction is public-key like, anyone can create a searchable ciphertext for (id, t), while only the receiver holding the secret key for id can generate the corresponding search token. The searchable ciphertext hides id only within a fixed public epoch; it does not hide t itself. We define this primitive in Section 5 and construct it in Section 6.

The payload is protected by the IBE ciphertext. To bind the two ciphertext components, the IBE plaintext contains a hash of c_{hibe} together with the message. Thus, a stored record has the form $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$, where c_{hibe} enables discovery and c_{ibe} protects the message contents. The binding ensures consistency between the IBE ciphertext and the searchable ciphertext, without claiming that the binding by itself gives a CCA transformation. The concrete encoding is in Section 8.

Retrieving records. A naive single-server retrieval protocol would have a receiver with identity id generate a search token for (id, t), send the token to the server, and ask the server to return every record whose searchable ciphertext accepts under Check. This leaks too much; a malicious server can create probe records for a target identity and epoch, and then test whether a later query matches those records. We therefore distribute the retrieval protocol across two (or more) non-colluding servers.

At a high level, the receiver splits its search token into two randomized shares, one for each server. Each server partially evaluates Check over the searchable ciphertexts in its database, and the receiver combines the responses to recover the matching index set I. It then invokes PIR to retrieve the IBE ciphertexts at the indices in I.

The PIR invocations hide I from each corrupted non-colluding server, but the unpadded protocol reveals $|I|$. Our privacy theorem is therefore stated with retrieval-count leakage. Beyond the public epoch, database size, and retrieval count, a corrupted non-colluding server learns neither the receiver identity nor the retrieved index set. If an application requires hiding $|I|$, the receiver can pad retrievals to a fixed bound B, using dummy retrievals when $|I| < B$ and specifying an overflow policy for $|I| > B$.

3 Related Work

An extended discussion of related work appears in Appendix C.

Anonymous messaging. We compare against recent large-scale anonymous messaging systems with cryptographic guarantees. Riposte [20] hides the link between a posting party and a message, but leaks the identities of parties who posted. Atom [32] and XRD [33] use mix-network-style techniques and scalable architectures to improve throughput and anonymity. These systems are optimized for settings in which many servers and clients remain online, and often rely on inter-server communication or dummy traffic.

Talek [17] requires at least one honest non-colluding server and provides access-sequence indistinguishability for clients that already know which logical records they wish to retrieve. In contrast, $\mathcal{PIB}^3$ addresses an identity-discovery problem: a receiver must discover records encrypted to its identity and epoch without revealing that identity to a corrupted non-colluding bulletin-board server. Thus, Talek and $\mathcal{PIB}^3$ protect different leakage profiles. Talek hides access sequences; $\mathcal{PIB}^3$ hides retrieved indices and recipient identity.

4 Preliminaries

Notation. A function $f(\lambda)$ is *negligible* if it is asymptotically smaller than the inverse of every polynomial in λ. We write $\mathcal{M}$ for the message space, $\mathcal{ID}$ for the identity space, and $\mathcal{T}$ for the epoch space used as the second level of our HIBE identities. We denote by ϵ the empty string and by [] the empty list, and write $a||b$ for concatenation. We use $\approx_c$ to indicate computational indistinguishability. The notation $a \leftarrow b$ denotes assignment, while $a \leftarrow B$ denotes sampling from a set or distribution B. The notation $a = (b_0, b_1)$ denotes that a is parsed as (b_0, b_1). All adversaries are probabilistic polynomial time unless otherwise stated.

Network Model. We assume that clients communicate with the bulletin-board servers over anonymous channels, so the servers do not learn the network identity of the client on the other end of the channel. Such channels can be implemented using TOR or similar anonymous communication systems.

Decisional Bilinear Diffie-Hellman. Our HIBE construction relies on the decisional bilinear Diffie-Hellman assumption. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ be a bilinear pairing and let P generate $\mathbb{G}_1$. The *decisional bilinear Diffie-Hellman* problem is to distinguish $(P, aP, bP, cP, e(P,P)^{abc})$ from (P, aP, bP, cP, r), where a, b, c are sampled uniformly from $\mathbb{Z}_p^*$ and r is sampled uniformly from $\mathbb{G}_T$. We write $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dbdh}}$ for the advantage of an adversary $\mathcal{A}$ in this game.

Private Information Retrieval. We use a standard multi-server private information retrieval protocol in which a client retrieves record x_i from a replicated database $X = [x_1, \ldots, x_n]$ without revealing i to any corrupted non-colluding server. In our bulletin-board protocol, the receiver invokes PIR once for each matching record. Thus, the PIR calls hide the retrieved indices, but the number of PIR invocations is treated as explicit leakage unless the implementation pads retrievals to a fixed bound. The full PIR syntax and definition are in Appendix D.

Identity-Based Encryption. An Identity Based Encryption scheme IBE consists of algorithms $(\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Encrypt}, \mathsf{Decrypt})$. We require correctness, IND-CPA security, and CPA anonymity, also known as key privacy: given a ciphertext, an adversary without the corresponding secret keys cannot distinguish which of two challenge identities was used. The Boneh–Franklin IBE scheme [10] satisfies the anonymity property we use in the random-oracle model, as shown by Abdalla et al. [3]. The formal IND-CPA and CPA-anonymity experiments used in this work are given in Appendix D.

5 Hierarchical IBE with Searchable Ciphertexts ($\mathsf{HIBE}^{\mathsf{SC}}$)

We adapt Hierarchical IBE [9, 24, 28] to a setting in which ciphertexts are not used to carry messages, but instead to support private discovery. In a general HIBE scheme an identity is a vector $\vec{id} = [\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_\ell]$. In our application $\ell = 2$, and we write $\vec{id} = (\mathsf{id}, t)$, where $\mathsf{id} \in \mathcal{ID}$ is a party identifier and $t \in \mathcal{T}$ is a public epoch. We write $\mathsf{id} \subset \vec{id}$ when id is the first component of $\vec{id}$.

A Hierarchical Identity Based Encryption with Searchable Ciphertexts scheme $\mathsf{HIBE}^{\mathsf{SC}}$ is a tuple of algorithms $(\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{CipherGen}, \mathsf{Token}, \mathsf{Check})$. The algorithms Setup and KeyDer are as in an IBE scheme. The additional searchable-ciphertext algorithms are:

- $c \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{CipherGen}(\mathsf{pp}, \vec{id})$ computes a searchable ciphertext c with respect to identity vector $\vec{id}$.
- $\tau \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Token}(\mathsf{pp}, \mathsf{sk}, \vec{id})$ returns a search token τ with respect to identity vector $\vec{id}$.
- $b \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(\mathsf{pp}, \tau, c)$ returns either $b = 0$ or $b = 1$.

The intended behavior is that $\mathsf{Check}(\mathsf{pp}, \tau, c)$ returns 1 exactly when c was generated with respect to $\vec{id} = (\mathsf{id}, t)$ and τ was generated for the same vector using a secret key derived for the first-level identity id. If either the identity or epoch differs, then Check should return 0, except with negligible probability.

Security. Unlike an IBE scheme, $\mathsf{HIBE}^{\mathsf{SC}}$ does not require semantic security because it does not encrypt messages. Instead, we require correctness, soundness, and fixed-epoch anonymity. Correctness requires that honestly generated tokens identify honestly generated ciphertexts for the same identity vector. Soundness requires that no adversary without access to the first-level secret key for id can produce a token that accepts an honestly generated ciphertext for an identity vector (id, t). Fixed-epoch anonymity requires that ciphertexts hide the first-level identity within a fixed public epoch: an adversary should not distinguish a ciphertext generated for (id_0, t) from one generated for (id_1, t), provided it has not obtained either first-level secret key. The epoch t is fixed across the challenge identities and is treated as public leakage.

We write $\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ for the advantage of adversary $\mathcal{A}$ in the soundness game, and $\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ for its advantage in the fixed-epoch anonymity game. A

$\mathsf{HIBE}^{\mathsf{SC}}$ scheme is *sound* if $\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ is negligible for every PPT adversary $\mathcal{A}$. It is *fixed-epoch anonymous* if $\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ is negligible for every PPT adversary $\mathcal{A}$. The formal correctness definition and security experiments are given in Appendix E.

6 Two-Level HIBE With Searchable Ciphertexts

We present a two-level HIBE scheme with searchable ciphertexts, obtained by modifying the HIBE scheme of Abdalla et al. [3]. The scheme does not encrypt plaintext messages; instead, a ciphertext can be tested, using an appropriate search token, to determine whether it was generated for an identity vector (id, t). We prove soundness and fixed-epoch identity anonymity for the construction in Appendix G. This construction may be of independent interest. Removing the plaintext also removes one random-oracle call from Abdalla's scheme, and our anonymity proof reduces to Decisional Bilinear Diffie-Hellman (DBDH), rather than Computational Bilinear Diffie-Hellman (CBDH).

Construction intuition. In Abdalla's scheme, the decryptor uses its first-level secret key and the remaining ciphertext components to recover a pairing-based mask. We remove the encrypted message and use the mask itself as the searchable component. A ciphertext for (id, t) contains enough information for a holder of the secret key for id to recompute the same mask for epoch t, but not enough for a party without that key to do so. The token and check algorithms in Figure 2 implement this mask-comparison test.

In the $\mathcal{PIB}^3$ application, the HIBE identity vector is (id, t). The construction hides id only when t is fixed across the challenge ciphertexts. It does not hide t itself, because the epoch component is publicly testable from the searchable ciphertext. Accordingly, the security notion proved for the construction is fixed-epoch identity anonymity.

6.1 Construction

Figure 2 gives the construction. The first level of the identity vector is a party identifier $\mathsf{id} \in \mathcal{ID}$, and the second level is an epoch $t \in \mathcal{T}$. The KeyDer algorithm derives a key only for the first-level identity id; the epoch-specific token material is derived later using the public hash function H_2.

The setup algorithm samples groups $\mathbb{G}_1$ and $\mathbb{G}_T$, a symmetric bilinear pairing $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, and public hash functions $\mathsf{H}_1\colon \mathcal{ID} \to \mathbb{G}_1$ and $\mathsf{H}_2\colon \mathcal{T} \to \mathbb{G}_1$. For a ciphertext generated for $\vec{id} = (\mathsf{id}, t)$, the values $c_0 = r \cdot Q_0$ and $c_1 = r \cdot P_t$, where $P_t = \mathsf{H}_2(t)$, make the epoch publicly testable: for a candidate epoch τ, one can check whether $e(c_1, Q_0) = e(\mathsf{H}_2(\tau), c_0)$. Thus the construction hides the first-level identity only for a public epoch.

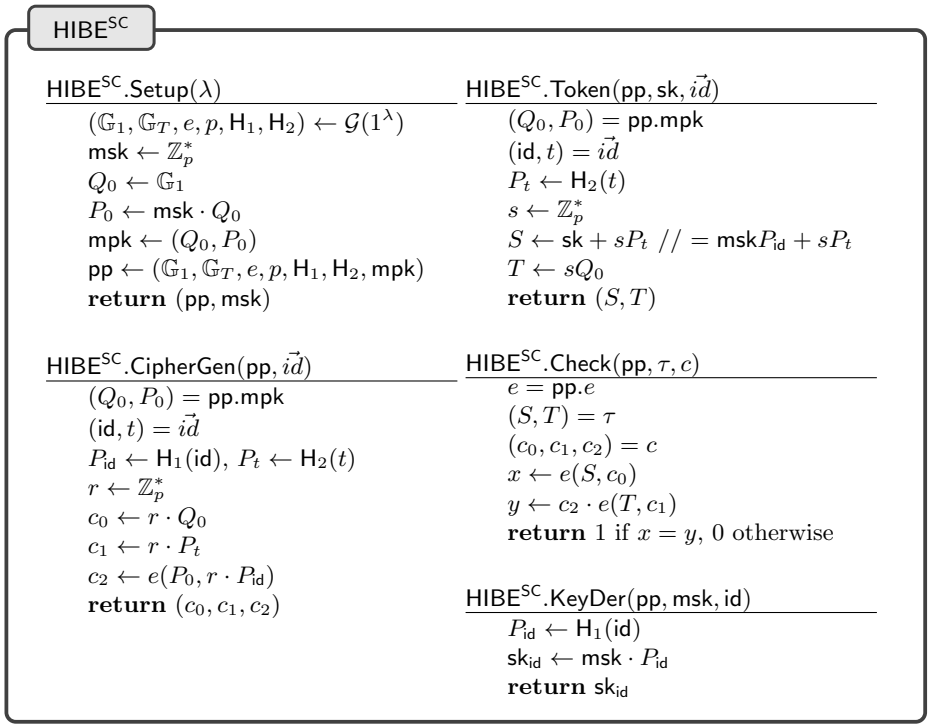


Fig. 2: Two-Level $\mathsf{HIBE}^{\mathsf{SC}}$ with searchable ciphertexts

7 Private Bulletin Board

Definition 1 (Bulletin Board Scheme). *An s-server* bulletin board scheme *is a tuple* (Setup, KeyDer, Encode, Reconstruct, Query) *where*

- $\mathsf{pp}, \mathsf{msk} \leftarrow \mathsf{Setup}(\lambda)$ *initializes the scheme and returns the public parameters*
- $k \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathsf{id})$ *samples a key* $k \in \mathcal{K}$ *corresponding to an identity* $\mathsf{id} \in \mathcal{ID}$
- $c \leftarrow \mathsf{Encode}(\mathsf{pp}, m, \mathsf{id})$ *on input of a message* m *and identity* id*, generates a confidentiality preserving encoding.*
- $m \leftarrow \mathsf{Reconstruct}(\mathsf{pp}, \mathsf{sk}, c)$ *reconstructs a message* m *given its encoding* c *and a secret key* sk*.*
- Query *is a* protocol *between a receiver* R *and* s *servers* $S_1, \ldots, S_s$ *in which the servers hold (identical) databases containing* n *records* $D = [r_1, r_2, \ldots, r_n]$ *and output nothing, and the receiver inputs its secret key and outputs some subset of the records held by the servers.*

Decodability. We require that parties can decode (or decrypt) the messages that have been posted for them.

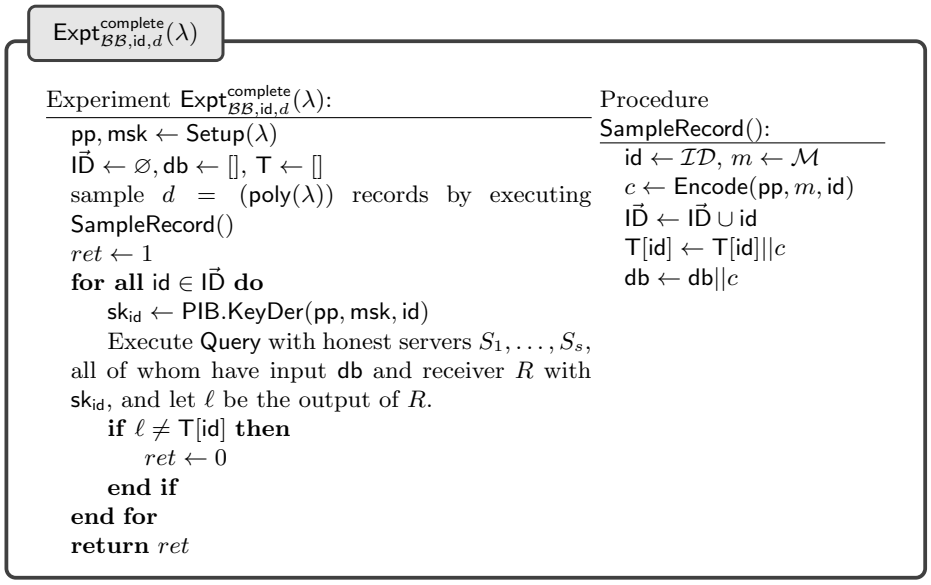


Fig. 3: Bulletin Board Completeness Experiment

Definition 2 (Bulletin Board Decodability). *A bulletin board scheme is* decodable *if for all λ and all $m \in \mathcal{M}$:*

$$\Pr\left[m = m' \; : \; \begin{array}{c} \mathsf{pp}, \mathsf{msk} \leftarrow \mathsf{Setup}(\lambda) \\ \mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathsf{id}) \\ c \leftarrow \mathsf{Encode}(\mathsf{pp}, m, \mathsf{id}) \\ m' \leftarrow \mathsf{Reconstruct}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}}, c) \end{array}\right] = 1$$

Completeness. A scheme is *complete* if whenever a receiver R runs the Query protocol with the bulletin board servers, it receives *all* of the messages that are generated with respect to its identity. This property is formalized via the experiment in Figure 3.

Definition 3 (Bulletin Board Completeness). *A bulletin board scheme is* complete *if for all λ,* $\Pr[\mathsf{Expt}^{\mathsf{complete}}_{\mathcal{BB},\mathsf{id},d}(\lambda) = 1] = 1$.

Privacy. A bulletin board is *private* if each corrupted non-colluding server learns no information from ciphertexts or from its interactions in the Query protocol beyond explicitly modeled leakage.

For the construction in this paper, the leakage function includes the public epoch associated with each stored record, the database size, and the number of PIR retrievals made by the receiver in each query. The leakage does not include the receiver identity id, the retrieved index set I, or the message contents. The experiment permits adaptive uploads and active probing up to this leakage

function: an adversary may upload records whose presence changes the number of ciphertexts matching a later query, but in the unpadded construction this information is revealed only through the retrieval count $|I|$.

We formalize this property via the security experiment in Figure 4. The game is designed to capture a full execution of a bulletin board with an eavesdropping adversary that corrupts one of the servers and therefore learns the server's view during PIR. The adversary learns the public parameters and is given oracle access to oracles for KeyDer, Encode, Reconstruct, and Query, subject to the restrictions below. The query to Encode additionally adds the returned ciphertext to a growing list of ciphertexts which emulate a database on which Query is called.

The adversary issues its challenge by invoking the $\mathcal{O}^{\mathsf{Test}}$ oracle with a pair (m^*, id^*), where m^* is a message and id^* is an identity. The oracle returns either an encoding c^* of (m^*, id^*) (where $\mathcal{A}$ may not know the secret key for id^* via KeyDer, Reconstruct, or Query), or an encoding of a random message with respect to a fresh random identity. $\mathcal{A}$ wins if it can identify which of these c^* encodes.

The oracle $\mathcal{O}^{\mathsf{Reconstruct}}$ serves as a reconstruction oracle for non-challenge identities. It is unavailable for the challenge identity; this restriction is what allows the proof to rely on CPA anonymity and IND-CPA security of the IBE scheme. The same challenge-identity restriction applies to key-derivation and query access: after the challenge identity is fixed, the adversary may obtain keys, reconstruction results, and query executions for non-challenge identities, but it may not use these oracles to obtain the secret key for the challenge identity or to reconstruct challenge-identity ciphertexts.

$\mathcal{A}$'s advantage in the experiment is $\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}(\lambda) = 2|\Pr[\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}|$.

Definition 4 (Bulletin Board Privacy). *A bulletin board scheme is* private *if for all* λ, $\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

A bulletin board is *query private* if a single corrupted non-colluding server learns no information from its view of Query beyond the number of PIR invocations.

Definition 5 (Query Privacy with Count Leakage). *Let* $\mathsf{View}^{\mathsf{Query}}_{\lambda,s,\mathsf{db}}(S_i)$ *be the random variable denoting the view of* S_i *interacting with honest receiver and honest servers* $S_1, \ldots, S_s$ *(excluding* S_i*) in an execution of the* Query *protocol with security parameter* λ*, where each server holds a set of records* db*. An* s*-server bulletin board scheme is* query private with count leakage *if for all* λ*, all* $i \leq s$*, and all retrieval counts* q*, there exists a simulator* $\mathcal{S}$ *such that* $\mathcal{S}(1^\lambda, i, s, \mathsf{db}, q) * \lambda, i, s, \mathsf{db}, q \approx_c \mathsf{View}^{\mathsf{Query}}_{\lambda,s,\mathsf{db}}(S_i)_{\lambda,i,s,\mathsf{db},q}$.

8 Private Bulletin Board From IBE and HIBE

We now define PIB, our instantiation of a $\mathcal{PIB}^3$ scheme. PIB is composed from a semantically secure and CPA-anonymous IBE scheme,[2] a HIBE with searchable

[2] The Boneh-Franklin scheme [10] is proved CPA-anonymous by Abdalla et al. [3]. The construction requires an IBE message space large enough to include a hash of the HIBE searchable ciphertext together with the payload.

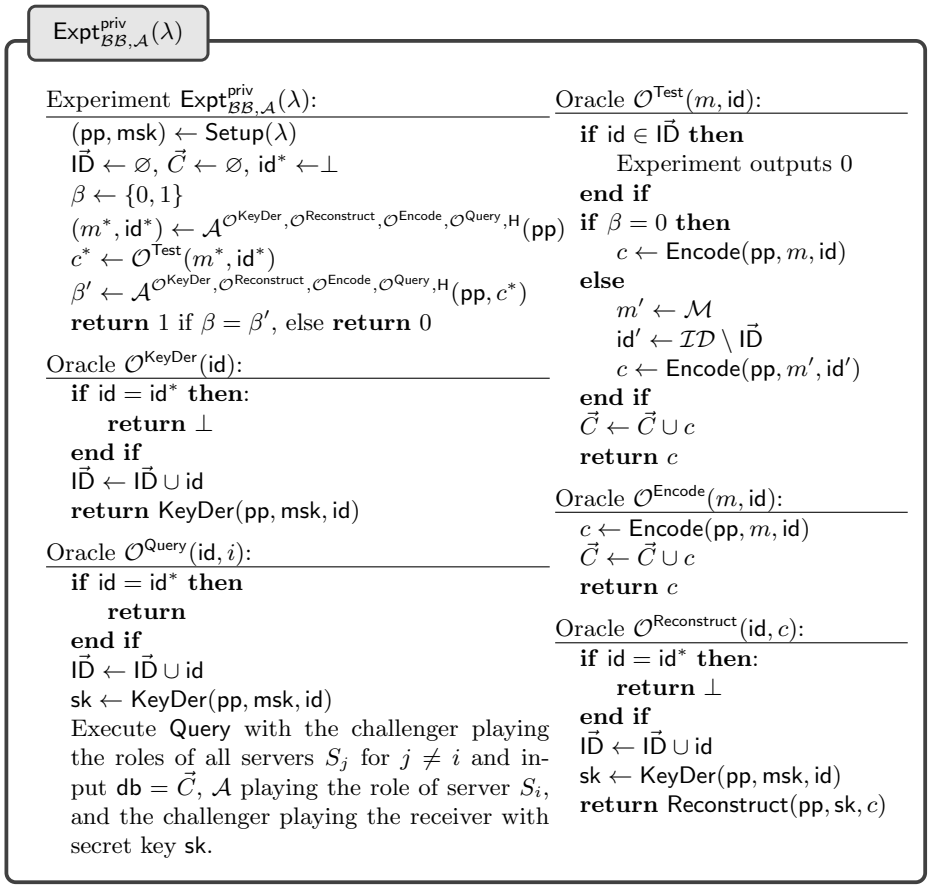


Fig. 4: Bulletin Board Privacy Experiment

ciphertexts, and PIR. Section 8.1 gives the construction, Section 8.2 states the security theorem, and Appendix A gives the communication optimization.

8.1 Bulletin-Board Construction

Figure 5 gives the non-query algorithms of PIB, and Figure 6 gives the two-server Query protocol. The identity input to PIB is a vector $\vec{id} = (\mathsf{id}, t)$, where id is the receiver identity used by both IBE and HIBE, and t is the public epoch. The setup algorithm samples public parameters for both schemes and a hash function $\mathsf{H} \in \mathcal{H}$.

Storing messages. To send message m to identity id in epoch t, the sender first creates a HIBE searchable ciphertext c_{hibe} for $\vec{id} = (\mathsf{id}, t)$. It then encrypts $\mathsf{H}(c_{\mathsf{hibe}})||m$ under IBE identity id to obtain c_{ibe}, and stores the pair $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$.

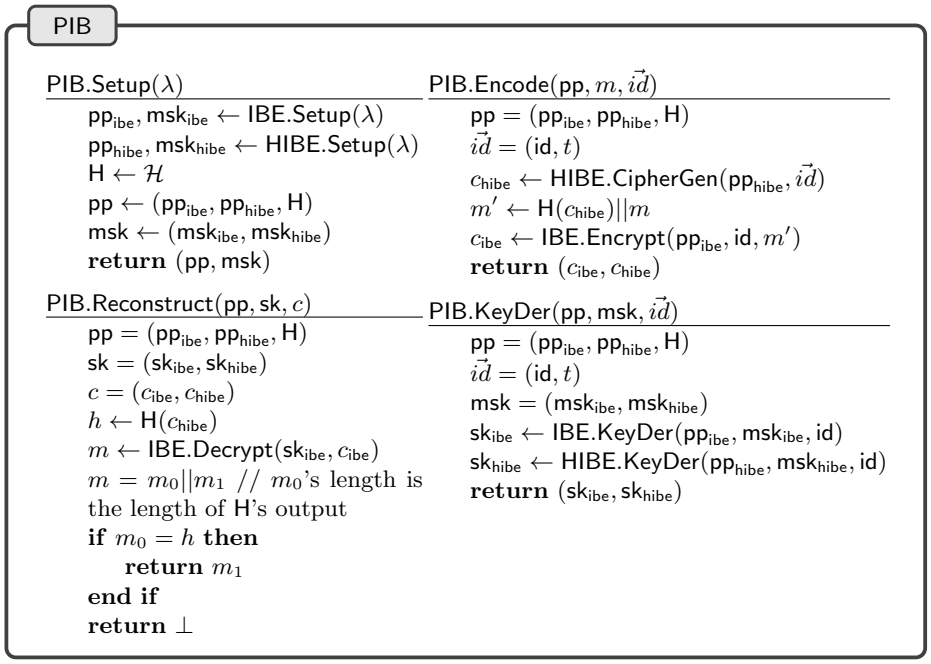


Fig. 5: Bulletin Board Scheme PIB excluding the Query protocol.

The hash binds the IBE ciphertext to the searchable ciphertext; this binding is used in the proof in Appendix H.

Query protocol. A naive single-server query sends a HIBE token to the server, which runs HIBE.Check over the database and returns matching IBE ciphertexts. This is insecure: Appendix F shows that a malicious server can detect when a targeted recipient retrieves its messages. Our construction therefore uses two non-colluding servers holding replicated databases. The receiver splits its search token, each server partially evaluates Check, and the receiver combines the responses to learn the matching index set I. It then invokes PIR to retrieve the records at indices in I. The PIR calls hide the elements of I, while the unpadded protocol reveals $|I|$ as retrieval-count leakage.

8.2 Security of PIB

The full proof of the main theorem (Theorem 1) is deferred to Appendix H, we only provide a proof sketch below.

Theorem 1 (Security of Bulletin Board Scheme). *If $\mathcal{A}$ is an adversary against bulletin-board privacy for* PIB *in the leakage model of Figure 4, then there exist adversaries $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{E}$ such that*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{pir}}_{\mathcal{PIR},\mathcal{B}}(\lambda) + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{C}}(\lambda) + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{D}}(\lambda) + \mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{E}}(\lambda).$$

Protocol 1 2-Server Query Protocol $\mathsf{PIB.Query}$

Common Input

- IBE public parameters $\mathsf{pp}_{\mathsf{ibe}}$
- HIBE public parameters $\mathsf{pp}_{\mathsf{hibe}}$

Party Inputs

- Every server S_i has a list of records $C = [C_1, \ldots, C_n]$, where $C_i = (C^{\mathsf{IBE}} * i, C^{\mathsf{HIBE}} * i)$, and $C^{\mathsf{HIBE}} * i = (c * 0, i, c * 1, i, c * 2, i)$
- Receiver R has identity id, a secret key $\mathsf{sk} = (\mathsf{sk}_{\mathsf{ibe}}, \mathsf{sk}_{\mathsf{hibe}})$, and an epoch t

Outputs R outputs the records whose search ciphertexts were generated with respect to its public key. S_1 and S_2 output $\perp$.
Protocol

1. R generates a search token for epoch t: $(S, T) \leftarrow \mathsf{HIBE.Token}(\mathsf{pp}_{\mathsf{hibe}}, \mathsf{sk}_{\mathsf{hibe}}, (\mathsf{id}, t))$. R sends S to S_1 and T to S_2
2. For every searchable ciphertext $C^{\mathsf{HIBE}} * i = (c * 0, i, c_{1,i}, c_{2,i})$ in its list, S_1 computes $x_i = e(S, c_{0,i})$. For every searchable ciphertext $C^{\mathsf{HIBE}} * i = (c * 0, i, c_{1,i}, c_{2,i})$ in its list, S_2 computes $y_i = c_{2,i} \cdot e(T, c_{1,i})$.
3. S_1 sends $X = [x_1, \ldots, x_n]$ to R. S_2 sends $Y = [y_1, \ldots, y_n]$ to R. R computes $I = i : x_i = y_i$.
4. R invokes $\mathcal{F}^{\mathsf{pir}}$ with S_1 and S_2 as servers to retrieve the IBE ciphertexts at the indices in I.

Fig. 6: Two-Server Query Protocol $\mathsf{PIB.Query}$

Proof sketch. The proof proceeds by hybrids from a corrupted server's real view to a simulated view. PIR privacy hides the retrieved indices up to the number of PIR invocations. Fixed-epoch anonymity of the HIBE searchable ciphertext hides the receiver identity within the public epoch. CPA anonymity of IBE hides the payload-ciphertext identity, and IND-CPA security of IBE hides the payload. The HIBE ciphertext is included in the IBE plaintext so that the ciphertext components remain bound. Adaptive uploads and active probing are covered up to retrieval-count leakage: if maliciously uploaded records change the number of matches for a later query, that change is part of the leakage given to the simulator. If an application needs to hide the retrieval count itself, the protocol can be padded to a fixed bound with an explicit overflow policy.

Performance. The communication complexity of PIB can be optimized by tuning parameters on the lengths of the messages sent by the servers to the receiver. Appendix A shows that the optimized communication complexity of the retrieval protocol is $O(n \log n)$, and by batching queries it can be completed in 4 rounds.

References

1. URL: https://oc.app/.
2. Private information retrieval: A primer, 2008. URL: https://www.cs.bgu.ac.il/ beimel/Papers/PIRsurvey.pdf.
3. Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. Cryptology ePrint Archive, Paper 2005/254, 2005. https://eprint.iacr.org/2005/254. URL: https://eprint.iacr.org/2005/254.
4. Kinan Dak Albab, Rawane Issa, Mayank Varia, and Kalman Graffi. Batched differentially private information retrieval. In *USENIX Security Symposium*, pages 3327–3344. USENIX Association, 2022.
5. Martin R. Albrecht, Lenka Mareková, Kenneth G. Paterson, and Igors Stepanovs. Four attacks and a proof for telegram. In *IEEE Symposium on Security and Privacy*, pages 87–106. IEEE, 2022.
6. Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the signal protocol. In *EUROCRYPT (1)*, volume 11476 of *Lecture Notes in Computer Science*, pages 129–158. Springer, 2019.
7. Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the servers computation in private information retrieval: PIR with preprocessing. In *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 55–73. Springer, 2000.
8. Kul Bhushan. Offline messaging apps like bridgefy firechat gain popularity amid internet shutdowns. HT Tech, 2022. https://tech.hindustantimes.com/tech/news/offline-messaging-apps-like-bridgefy-firechat-gain-popularity-amid-internet-shutdowns-story-GVBABExmFCR5byMKScdCII.html.
9. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.
10. Dan Boneh and Matthew Franklin. Identity based encryption from the weil pairing. Cryptology ePrint Archive, Report 2001/090, 2001. https://eprint.iacr.org/2001/090.
11. X Boyen and L Martin. Identity-Based Cryptography Standard (IBCS) #1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems. RFC 5091, RFC Editor, December 2007. URL: https://www.rfc-editor.org/rfc/5091.txt.
12. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *EUROCRYPT (2)*, volume 9057 of *Lecture Notes in Computer Science*, pages 337–367. Springer, 2015.
13. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In *CCS*, pages 1292–1303. ACM, 2016.
14. Zvika Brakerski and Yael Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. *IACR Cryptol. ePrint Arch.*, page 86, 2010.
15. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.
16. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–88, 1981.

17. Raymond Cheng, William Scott, Elisaweta Masserova, Irene Zhang, Vipul Goyal, Thomas E. Anderson, Arvind Krishnamurthy, and Bryan Parno. Talek: Private group messaging with hidden access patterns. *CoRR*, abs/2001.08250, 2020.
18. Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. In *EuroS&P*, pages 451–466. IEEE, 2017.
19. Jean-Sébastien Coron. On the exact security of full domain hash. In *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2000.
20. Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. Riposte: An anonymous messaging system handling millions of users. In *IEEE Symposium on Security and Privacy*, pages 321–338. IEEE Computer Society, 2015.
21. Henry Corrigan-Gibbs, Alexandra Henzinger, and Dmitry Kogan. Single-server private information retrieval with sublinear amortized time. Cryptology ePrint Archive, Paper 2022/081, 2022. https://eprint.iacr.org/2022/081. URL: https://eprint.iacr.org/2022/081.
22. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation onion router. In *13th USENIX Security Symposium (USENIX Security 04)*, San Diego, CA, August 2004. USENIX Association. URL: https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router.
23. Craig Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.
24. Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.
25. Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. In *STOC*, pages 151–160. ACM, 1998.
26. Larry Greenemeier. How was egypt's internet access shut off? Scientific American, 2011. https://www.scientificamerican.com/article/egypt-internet-mubarak/.
27. James Griffiths. Blocking social media would be 'the end of the open internet of hong kong.' it also wouldn't work. CNN, 2019. https://www.cnn.com/2019/08/29/tech/hong-kong-internet-block-emergency-powers-intl-hnk/index.html.
28. Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.
29. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography from anonymity. In *FOCS*, pages 239–248. IEEE Computer Society, 2006.
30. Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Outsourced symmetric private information retrieval. In *CCS*, pages 875–888. ACM, 2013.
31. Daniel Kales, Olamide Omolola, and Sebastian Ramacher. Revisiting user privacy for certificate transparency. In *EuroS&P*, pages 432–447. IEEE, 2019.
32. Albert Kwon, Henry Corrigan-Gibbs, Srinivas Devadas, and Bryan Ford. Atom: Horizontally scaling strong anonymity. In *SOSP*, pages 406–422. ACM, 2017.
33. Albert Kwon, David Lu, and Srinivas Devadas. XRD: scalable messaging system with cryptographic privacy. In *NSDI*, pages 759–776. USENIX Association, 2020.

34. David Lazar, Yossi Gilad, and Nickolai Zeldovich. Karaoke: Distributed private messaging immune to passive traffic analysis. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 711–725, Carlsbad, CA, October 2018. USENIX Association. URL: https://www.usenix.org/conference/osdi18/presentation/lazar.
35. Wouter Lueks and Ian Goldberg. Sublinear scaling for multi-client private information retrieval. In *Financial Cryptography*, volume 8975 of *Lecture Notes in Computer Science*, pages 168–186. Springer, 2015.
36. Yiping Ma, Ke Zhong, Tal Rabin, and Sebastian Angel. Incremental offline/online PIR. In *USENIX Security Symposium*, pages 1741–1758. USENIX Association, 2022.
37. Victor Miller. Short programs for functions on curves.
38. Nicholas Pippenger. On the evaluation of powers and monomials. *SIAM J. Comput.*, 9(2):230–250, 1980.
39. Matt Richtel. Egypt cuts off most internet and cell service. The NY Times, 2011. https://www.nytimes.com/2011/01/29/technology/internet/29cutoff.html.
40. Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.
41. Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nickolai Zeldovich. Stadium: A distributed metadata-private messaging system. In *SOSP*, pages 423–440. ACM, 2017.
42. Jelle van den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. Vuvuzela: scalable private messaging resistant to traffic analysis. In *SOSP*, pages 137–152. ACM, 2015.

A Collisions and Performance

Collision Analysis. To improve the communication complexity of the messages *sent by the servers to the receiver*, it is possible to modify Step 3 of PIB.Query so that S_1 sends $H(x_i)$ for all $x_i \in X$ and S_2 sends $H(y_i)$ for all $y_i \in Y$ to the receiver, where H is a hash function (separate from the hash function used in the PIB scheme to hash c_{hibe}). If H is compressing, then this allows the two servers to send smaller messages to the receiver.

For correctness of this modification, H must not output $H(x_i) = H(y_i)$ for which $x_i \neq y_i$. [3] Modeling H as a random oracle, the probability of this outcome can be tuned by choosing the output length H. In the following analysis, let $n = |X| + |Y|$ and let ℓ be the output length of H.

By the birthday bound heuristic, $\Pr[\mathsf{collide}] \approx \frac{n^2}{2 \cdot 2^\ell}$. Where $n = 100,000$ and $\Pr[\mathsf{collide}] = 2^{-80}$, $\ell \geq 113$ bits. For $\Pr[\mathsf{collide}] = 2^{-100}$, $\ell \geq 133$ bits.

Optimizing Communication A spurious collision occurs when some recipient R receives $H(x_i) = H(y_i)$ but $x_i \neq y_i$. In this case, R has received the index of a record that *it cannot decrypt*. Therefore, the cost of the error is an extra invocation of PIR by R, but no privacy is lost for the scheme.

[3] In the standard model, a universal hash function suffices, and is efficient to implement.

Therefore, we can trade off spurious collisions in order to reduce the total communication complexity of the protocol. For the most efficient (non-preprocessing) PIR protocols [12, 13], the communication complexity of a PIR is $O(c\log(n))$, where c is a small constant and n is the number of records in the database. It follows that the communication of a ciphertext retrieval is

$$n\ell + (F+m)c\log(n) \tag{1}$$

where F is the number of spurious collisions and m is the number of "true" records that belong to the receiver. The first term represents the number of bits sent by each server to the receiver in the first step, and the second term represents the size of the PIR query that the client sends to the servers. (The final response by the servers is $(F+m)C$, where C is the size of a ciphertext, and we omit it for this analysis.)

The communication complexity is minimized by tuning ℓ to minimize $n\ell + Fc\log(n)$. F is a function of n and ℓ, derived from the birthday bound:

$$F(n,\ell) = \max(0, n - 2^\ell + 2^\ell(\frac{2^\ell - 1}{2^\ell})^n) \tag{2}$$

We can simplify the term we wish to minimize from Equation (1):

$$n\ell + c\log(n)\max(0, n - 2^\ell + 2^\ell(\frac{n-1}{n})^n) \tag{3}$$

Setting $\ell = O(\log(n))$, we simplify

$$n\log(n) - nc\log(n)(\frac{n-1}{n})^n \tag{4}$$

and find that the communication complexity is $O(n\log(n))$ for the retrieval protocol. By batching all PIR queries that the client wishes to make, it can also be completed in 4 rounds.

B Experimental Results

This section first describes the experimental setup, then reports the measured performance results and their analysis. We conclude the section with a brief discussion of future optimizations to consider to further improve the performance of such a system. (The implementation will be open sourced once the paper is published.)

B.1 Experimental Setup

We implemented a prototype of the computationally heavy aspects of PIB in order to empirically evaluate its feasibility. All experiments were performed using an IBE configuration resulting in a 128 bit security parameter [11]. Experimental tests for different parameter configurations (number and size of records held by

the servers) were executed on AWS EC2 using a size c6a.8xlarge machine with 32 virtual processors (vCPUs). Each test was repeated 10 times, except for the final row in Figure 8, which was run twice. The variances of all results were negligible and are therefore omitted.

B.2 Evaluation

We separately evaluated the servers' computation for the two flows of the PIB.Query protocol. In the first flow, the client sends a share of a search token to each bulletin board server, and each server partially evaluates the Check function over every searchable ciphertext using its share of the token. In the second flow, the client then invokes a PIR protocol [13] with the servers.

Check*: Evaluating Many Pairings:* Figure 7 displays the evaluation of both S_1's and S_2's partial computation of the Check function (Step 2 in Figure 6: Two-Server Query Protocol PIB.Query). Specifically, each server must compute pairings over all records in its database, but over different elements of the ciphertext. "S_i Search Time" denotes the time (in seconds) required to perform the computation of S_i's role, given a share of the search token. The cost scales linearly, and even for large databases (100,000 ciphertexts), the times are reasonable for our application. Appendix B.4 contains additional optimizations for the pairing computations. The experiments were run on a machine with 32 vCPUs. For the smaller sizes, pairings were distributed across 64 threads, while for the larger sizes they were distributed across 128 threads.

PIR over a Database of Ciphertexts: Our implementation adapts the PIR scheme by Boyle et al. [13] and modifies the library by Kales et al. [31] for our application. Specifically, [31] required only 256-bit records, but our records are very large. For this reason, we also did not fold codewords into the final layer using the early termination option [13]. In the DPF tree, we maintained PRF seeds of size 128-bits until the final layer, when they were expanded to match the ciphertext sizes.

Figure 8 contains the evaluation of the PIR component of PIB.Query. Recall that after a receiver learns the indices of the ciphertexts that it can decrypt, it must retrieve these via PIR. In our experiments, we varied both the size of the encrypted plaintext messages and the number of records held by the

# Ciphertexts	S_1 Search Time	S_2 Search Time
1000	0.471 s	0.481 s
10000	4.64 s	4.64 s
50000	22.51 s	22.48 s
100000	45.00 s	44.97 s

Fig. 7: PIB.Query ciphertext-search times per server (Step 2 in Figure 6: 2-Server Query Protocol PIB.Query). S_1 denotes one of the servers and S_2 the other one.

Ciphertext Size	# Ciphertexts	DPF Gen	DPF Eval
0.3 kb	1000	2.42×10^{-5}	8.35×10^{-4}
	10000	2.66×10^{-5}	1.29×10^{-2}
	100000	2.86×10^{-5}	0.104
1kb	1000	2.77×10^{-5}	2.64×10^{-3}
	10000	3.30×10^{-5}	4.23×10^{-2}
	100000	3.33×10^{-5}	0.337
10kb	1000	2.96×10^{-5}	2.99×10^{-3}
	10000	1.57×10^{-4}	0.397
100kb	1000	1.23×10^{-3}	0.256
	10000	1.22×10^{-3}	4.021
	100000	1.24×10^{-3}	35.597

Fig. 8: PIR Evaluation Times, in seconds (Step 4 in Figure 6: 2-Server Query Protocol PIB.Query.) DPF stands for distributed point function.

database. The computation was bottlenecked by the full domain evaluation of the distributed point function.

The Client's Query*:* We did not benchmark the full client's computation during its computation of Query. The client first computes the Gen operation of a distributed point function, which is shown in Figure 8 to be very efficient. The client's larger computation is a series of equality checks, which is an inexpensive operation and can be performed in parallel (and on a GPU).

B.3 Latency and Communication

Computational Latency: The latency of Steps 1 and 2 of PIB.Query is measured by

1. the time for a client to send a search token to each server
2. the time for each server to perform its search and return a response (Figure 7)
3. the time for the client to perform its equality check, and
4. the time to run PIR (Figure 8).

The total computational latency by the servers for our largest test scenario – 100,000 records containing plaintext payloads of 100kb – requires little above 80 seconds. For smaller messages or smaller databases, it measures in seconds or tenths of seconds. The computations for the receiver are all efficient; the most expensive is the equality check of n records, where n is the size of the database, but as explained these can be easily parallelized.

The communication of these steps is only between client and each server; the servers do not communicate with each other. Note that the communication can be optimized by applying a compressing function (as described in Appendix A) costs one additional hash evaluation per record and can also be performed in parallel.

Communication Cost: The communication cost of retrieval is

1. The receiver sends a share of a search token to each server. Each share is one group element.
2. The servers respond with the partial evaluation over each record in the database. For each record, they send one group element. Using the optimization in Appendix A, this reduces to $n\ell$, where ℓ is the hash output length and n is the number of records. We explained that setting $\ell = \log n$ optimally trades false positives for communication complexity.
3. The receiver computes DPF Gen for each apparent match. Each generated key is size $\lambda \log(n)+C$, where C (a ciphertext) is the size of a database record and λ is the security parameter.[4] (R is set in the first column of Figure 8, and $\lambda = 128$.)
4. The servers each respond with a message of size C.

The full (optimized) communication cost is therefore $O((n + \lambda)\log(n) + C + G)$, where G is the size of a group element. Specifically, the receiver sends $F(\lambda \log(n) + C) + G$ bits to each server, where F is the number of records that the receiver retrieves. Each server responds with a total of $n\ell + FC$ bits, where $\ell \leq G$.

B.4 Potential Future Optimizations

It is possible to further improve the servers' computation of the Check function algorithmically. Recall that the receiver sends a share of a search token to each server, and the server computes a function $f(T, r)$ over every record r in its database, where T is its share of the search token. T is a group element, and f contains a pairing over T. T can be used for a one-time preprocessing step that is amortized over every x.

Recall that Miller's algorithm [37] uses the double-and-add technique akin to exponentiation via logarithmic multiplications, where at each point the current accumulator is doubled and then another element is conditionally added to the accumulator. It is faster in many cases to find an addition-subtraction chain that computes the same product. Because the element T determines the precise chain (for a symmetric Tate pairing, this applies to both S_1 and S_2's computation, as the pairing arguments can be switched), this can be computed once, and the resulting chain can be used for every r to reduce the number of point multiplications.

Given the addition-subtraction chain – and the points it implies must be computed – it is possible to apply Pippenger's algorithm [38] to further reduce the cost of multiplication. We leave these optimizations as future work.

To improve on PIR costs, it is also possible to offload some operations to a GPU (as by Talek [17]).

[4] A ciphertext is the length of an encrypted plaintext plus $O(\lambda)$.

C Related Work

C.1 Anonymous Messaging

We compare our scheme only against recent large scale anonymous messaging systems with cryptographic guarantees, eliding discussion of systems that achieve differential privacy [34, 41, 42] and weaker guarantees.

The first large-scale anonymous bulletin board was Riposte [20], which leaked the identities of the parties who had posted messages, but prevented linking a posting party to any specific message. Subsequently, several systems have improved both the efficiency and privacy guarantees of scalable anonymous messaging. Atom [32] employs a series of mix networks and rerandomizable encryption to anonymize the sender of a message on a bulletin board, and scales horizontally to increase throughput of the system for each added server. XRD [33] improves on the scalability of Atom by using weaker cryptographic primitives and similarly scales horizontally, but as the system scales horizontally, clients must send more messages in order to communicate.

Talek [17] requires at least one honest non-colluding server among a group of messaging servers, and achieves *access sequence indistinguishability*, which states that the adversary cannot distinguish between an honest user's access pattern and a random access pattern of the same length, for honest parties communicating with each other. Our setting is different because receivers do not begin with the locations of the records they wish to retrieve. Instead, a receiver must first discover which records were encrypted to its identity and epoch, and must do so without revealing its identity to any corrupted non-colluding bulletin-board server. Thus, our security goal strengthens the retrieval problem along an identity-discovery dimension: the server should learn neither which database indices are retrieved nor which recipient identity is being served, except through the leakage explicitly modeled by our construction, namely public epochs, database size, and retrieval counts in the unpadded protocol. This guarantee is not directly comparable to Talek's access-sequence indistinguishability, because the two systems model different client knowledge and leakage profiles; however, the $\mathcal{PIB}^3$ construction addresses the additional identity-based discovery problem that arises when senders can post using only a recipient identity and receivers later come online to find their messages.

These highly scalable and anonymous schemes are optimized for speed and require many participating servers to be online serving clients. To obtain security, they often depend on mix networks [16] or require dummy messages to obscure the traffic of real messages [32, 17]. In our disruption-prone networks, these speed-optimized designs deteriorate, and the adversary can disrupt a system by taking down a link between servers. Instead, we put a premium on the number of messages sent through the censored network and do not require the servers to communicate with each other.

C.2 Private Information Retrieval (PIR)

PIR allows a client to retrieve a record from a remote database, without the database learning which record was retrieved. It has been a fast-advancing line of research in recent years [4, 36, 21]. However, it is not sufficient for our application because recipients must also discover the records held by the server(s) which it can decrypt. As a subroutine, we use a two-party computational PIR based on the distributed point functions of Boyle et al. [12, 13].

Our temporal scheme assumes that parties who wish to communicate add records over time. Online/Offline PIR [36] is a preprocessing model where the database changes, and show how the preprocessing for a database update needs to cost according to the size of the update. This could be useful for improving query costs if preprocessing is done at the beginning of each epoch as the databases purge stale records and make available new records that they have recently received, but the preprocessing must apply universally to all clients. We do not use a preprocessing PIR, but instead employ a highly efficient 2-server PIR [13]. This better fits our model, where we do not expect clients to repeatedly access the PIR enough to justify the costs of preprocessing.

C.3 Alternative Identity-Based Dropbox Designs

We briefly compare $\mathcal{PIB}^3$ with several strawman identity-based dropbox designs. These alternatives illustrate why standard IBE and PIR do not immediately yield an efficient identity-based bulletin board.

Naive approach. The naive approach to an identity-based dropbox is simply to upload IBE encryptions of messages to a dropbox server. When a recipient wishes to retrieve its messages, it downloads the entire database and attempts to decrypt every record. By the anonymity and semantic security of the encryption scheme, this scheme is private; however, it is not efficient.

Garbled-circuit equality checks. One could try to have two servers perform the client's equality checks via a garbled circuit, and then use a sorting network to move matching records to the top of a list. Then a bounded number of indices at the top of the list could be forwarded to the client. However, the sorting network requires $n \log^2 n$ comparisons on indices of size $\log n$, yielding at least $n \log^3 n$ communication. This is less efficient than the communication-optimized $\mathcal{PIB}^3$ retrieval protocol.

Identity reconstruction. Another approach is for a sender to upload an encrypted message together with a secret sharing of the intended receiver identity. When a receiver queries, it anonymously secret-shares its own identity to the dropbox servers. The servers then use their shares to determine which records belong to the requester.

This requires the servers to re-randomize, reconstruct, and compare identities for the database records, costing at least $O(n)$ communication between

the servers. If the scheme additionally authenticates the client, then the client must provide a certificate or proof that the servers evaluate inside MPC for each record, possibly including group operations. This substantially increases both server communication and computation.

In comparison, $\mathcal{PIB}^3$ has optimized retrieval communication $O(n \log n)$ and does not require interaction between the bulletin-board servers during retrieval, although the servers must store records in the same order.

D Additional Preliminaries

This appendix contains the standard PIR and IBE syntax and games used in Section 4.

D.1 Private Information Retrieval (PIR)

We adapt a definition of private information retrieval (PIR) from Beimel [2].

Definition 6 (Private Information Retrieval). *A Private Information Retrieval (PIR) protocol is a tuple of algorithms (Q, A, C), where*

- $(q_1, \ldots, q_k), \mathsf{st} \leftarrow Q(1^n, i)$ *is a randomized query algorithm which returns k queries, one for each server, and a state* st*. The client sends q_i to server i, where k is the total number of servers.*
- $a_j \leftarrow A(j, q_j, X)$ *returns an answer to the client, where j is the index of the server, q_j is the query sent to that server, and X is the database.*
- $x \leftarrow C(\mathsf{st}, i, a_1, \ldots, a_k)$ *is a reconstruction algorithm that takes the state* st*, the index i, and the answers $a_1, \ldots, a_k$ produced by A, and returns a record x.*

Definition 7 (Correctness of PIR). *A PIR scheme is* correct *if for every database of length n with records of size ℓ, every database $X = [x_1, \ldots, x_n] \in \{0,1\}^{\ell n}$, and every index $i \in \{1, \ldots, n\}$, if $(q_1, \ldots, q_k), \mathsf{st} \leftarrow Q(1^n, i)$ and $a_j \leftarrow A(j, q_j, X)$, then*

$$C(\mathsf{st}, i, a_1, \ldots, a_k) = x_i.$$

Definition 8 (Privacy of PIR). *A PIR scheme is* private *if there exists a simulator $\mathcal{S}$ such that for every λ and i,*

$$\left\{q_j : (q_1, \ldots, q_k), \mathsf{st} \leftarrow Q(1^\lambda, i)\right\}_{j \in [1 \ldots k]} \approx_c \left\{\mathcal{S}(1^\lambda, j)\right\}_{j \in [1 \ldots k]}.$$

D.2 Identity Based Encryption (IBE)

An Identity Based Encryption scheme is defined as in [10]. Let every identity id belong to a defined set $\mathcal{ID}$.

Definition 9 (Identity Based Encryption). *An Identity Based Encryption scheme* IBE *is a tuple of algorithms* $(\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Encrypt}, \mathsf{Decrypt})$*, where*

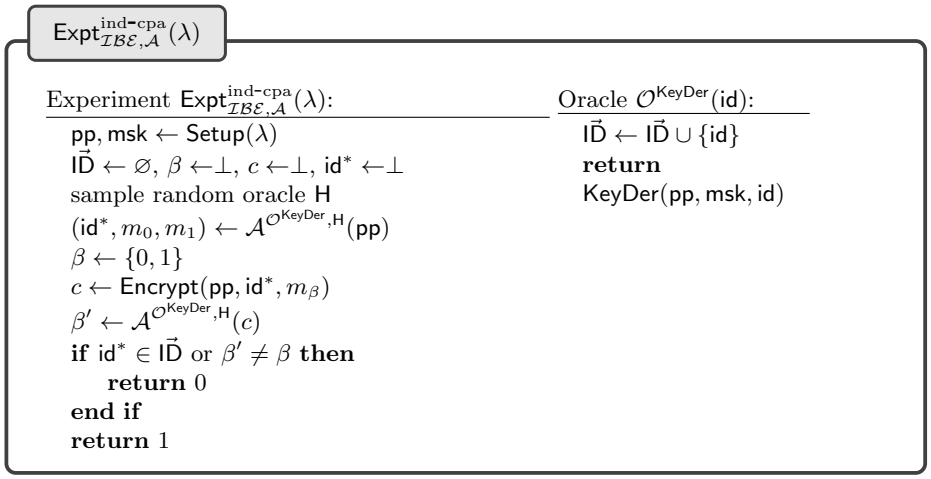


Fig. 9: IBE IND-CPA Experiment

- $\mathsf{pp}, \mathsf{msk} \leftarrow \mathsf{IBE.Setup}(\lambda)$ *generates public system parameters* pp *and a master key* msk.
- $\mathsf{sk} \leftarrow \mathsf{IBE.KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathsf{id})$ *derives a secret key for identity* id *using the public parameters and the master key.*
- $c \leftarrow \mathsf{IBE.Encrypt}(\mathsf{pp}, \mathsf{id}, m)$ *encrypts a message* m *for identity* id *using the public parameters.*
- $m \leftarrow \mathsf{IBE.Decrypt}(\mathsf{sk}, c)$ *decrypts a ciphertext* c *using secret key* sk.

IND-CPA Security. We derive our definition of IND-CPA security from [10] and present it in Figure 9. An adversary issues a challenge identity, for which it does not know the secret key, and two messages. The challenger samples one message at random and returns an encryption of that message for the challenge identity. We define $\mathcal{A}$'s advantage in the game as

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Expt}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|.$$

Definition 10 (IBE IND-CPA Security). *An IBE scheme is* IND-CPA-secure *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for any PPT adversary* $\mathcal{A}$ *and all* λ*,*

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)(\lambda).$$

CPA Anonymity. We additionally require the property that our IBE scheme is *CPA anonymous*, sometimes denoted as "key private." Intuitively, this means that given a ciphertext, no adversary without a decryption oracle should be able to infer the public key with respect to which a message was encrypted. We

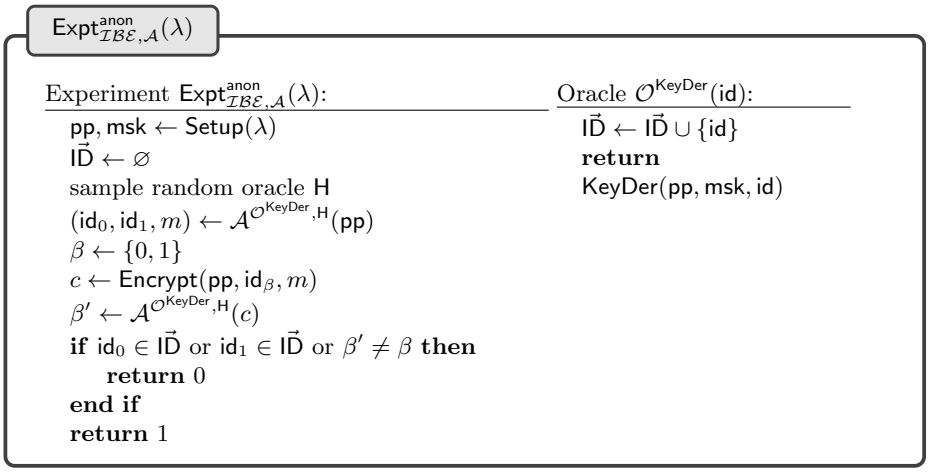


Fig. 10: IBE Anonymity Experiment

present the security game that formalizes this notion in Figure 10. We define $\mathcal{A}$'s advantage in the game as

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Expt}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|.$$

Definition 11 (IBE CPA Anonymity). *An IBE scheme is* CPA anonymous *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for any PPT adversary* $\mathcal{A}$ *and all* λ*,*

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)(\lambda).$$

E Additional Definitions for $\mathsf{HIBE}^{\mathsf{SC}}$

This appendix contains the formal syntax, correctness definition, and security experiments for the $\mathsf{HIBE}^{\mathsf{SC}}$ syntax summarized in Section 5.

Definition 12 (Hierarchical Identity Based Encryption with Searchable Ciphertexts). *A Hierarchical Identity Based Encryption with Searchable Ciphertexts scheme* $\mathsf{HIBE}^{\mathsf{SC}}$ *is a tuple of algorithms*

$$(\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{CipherGen}, \mathsf{Token}, \mathsf{Check}),$$

where algorithms Setup *and* KeyDer *are as in an* IBE *scheme, and:*

- $c \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{CipherGen}(\mathsf{pp}, \vec{id})$ *computes a searchable ciphertext* c *with respect to identity vector* $\vec{id}$*.*
- $\tau \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Token}(\mathsf{pp}, \mathsf{sk}, \vec{id})$ *returns a search token* τ *with respect to identity vector* $\vec{id}$*.*

– $b \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(\mathsf{pp}, \tau, c)$ *returns either* $b = 0$ *or* $b = 1$.

Every ciphertext c *is generated with respect to some identity vector* $\vec{id}$*. The intended correctness behavior is that* $\mathsf{Check}(\mathsf{pp}, \tau, c)$ *returns* 1 *when* c *was generated with respect to* $\vec{id} = (\mathsf{id}, t)$ *and* τ *was generated for the same vector using a secret key derived for the first-level identity* id*. If either the first-level identity or the epoch differs, then* Check *should return* 0*, except with negligible probability.*

Definition 13 (Correctness of HIBE with Searchable Ciphertexts). *A HIBE with Searchable Ciphertexts scheme is* correct *if for all* λ *and all* $\mathsf{id}, \vec{id}$ *such that* $\mathsf{id} \subset \vec{id}$*:*

$$\Pr\left[\mathsf{Check}(\mathsf{pp}, \tau, c) = 1 : \begin{array}{c} \mathsf{pp}, \mathsf{msk} \leftarrow \mathsf{Setup}(\lambda) \\ \mathsf{sk} \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathsf{id}) \\ c \leftarrow \mathsf{CipherGen}(\mathsf{pp}, \vec{id}) \\ \tau \leftarrow \mathsf{Token}(\mathsf{pp}, \mathsf{sk}, \vec{id}) \end{array}\right] = 1.$$

Soundness. Soundness requires that no adversary *without* access to a secret key for identity id can generate a token τ for which $\mathsf{Check}(\mathsf{pp}, \tau, c) = 1$, where c is a ciphertext honestly generated with respect to an identity vector $\vec{id} = (\mathsf{id}, t)$. This is formalized via the game $\mathsf{Expt}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ in Figure 11. We define $\mathcal{A}$'s advantage in the game as

$$\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Expt}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|.$$

Definition 14 (HIBE Soundness). *A HIBE with Searchable Ciphertexts scheme is* sound *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for any probabilistic polynomial-time adversary* $\mathcal{A}$ *and all* λ*,*

$$\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)(\lambda).$$

Fixed-Epoch Anonymity. The anonymity game $\mathsf{Expt}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda)$ is defined similarly to Abdalla et al. [3] and given formally in Figure 11. An adversary $\mathcal{A}$ with access to a KeyDer oracle and two hash functions issues a challenge consisting of two first-level identities $\mathsf{id}_0, \mathsf{id}_1$ and one public epoch t, where $\mathcal{A}$ does not have either secret key. The challenger responds with a ciphertext generated with respect to either (id_0, t) or (id_1, t). The epoch is fixed across the two challenge identities and is treated as public leakage. We define $\mathcal{A}$'s advantage in the game as

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Expt}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|.$$

Definition 15 (HIBE Fixed-Epoch Anonymity). *A HIBE scheme is* fixed-epoch anonymous *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for any probabilistic polynomial-time adversary* $\mathcal{A}$ *and all* λ*,*

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)(\lambda).$$

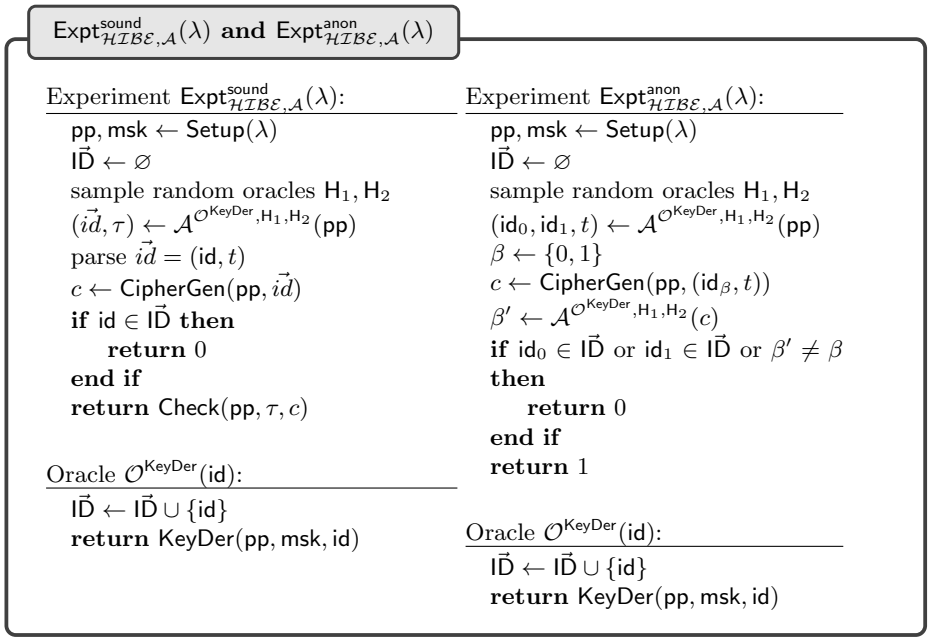


Fig. 11: HIBE Soundness Experiment and HIBE Fixed-Epoch Anonymity Experiment

F Single-Server Bulletin Board Leakage

This section explains the leakage of the naive single-server bulletin board scheme described in Section 8.1; specifically, this scheme is the same as the one presented in Figure 5, but instead of distributing the implementation of the bulletin board into two servers, it uses a single server. The Query protocol for this strawman is simple: a receiver R generates a search token via $T \leftarrow \mathsf{HIBE}^{\mathsf{SC}}.\tau(\mathsf{sk}_{\mathsf{hibe}}, t)$, and sends T to the server. The server runs $\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(c, T)$ for every searchable ciphertext c in its database. For each c such that Check returns 1, the server returns the ciphertext to R.

There are two issues with this scheme. We refer to the first as *access profile leakage*. The second is a chosen-identity attack.

F.1 Access Profile Leakage

In the scheme above, the server is able to group together records as belonging to a single recipient. This is because the server identifies the set of all records that verify with respect to a specific token T send by the recipient. Note that when using the temporal scheme $\mathsf{HIBE}^{\mathsf{SC}}$, the server cannot link records for id at time t with records for id at time $t + 1$.

F.2 Chosen-Identity Attack

The single-server scheme above permits a *chosen-identity attack*, in which the server detects whether messages are sent (or are not sent) to a particular identity id at a specific epoch t. Recall that the server only needs the public parameters for the HIBE scheme to generate a searchable ciphertext c with respect to the pair (id, t). Whenever it receives a search token T, the server computes $\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(c, T)$. If the result is 1 then this is the identity and epoch of the receiver making the request. If the result is 0 then this receiver did not issue the request.

We believe this attack is inherent to any searchable ciphertext scheme, but that it might be averted using state-of-the-art PIR techniques such as functional secret sharing [13] that returns a (masked) record only if the Check function returns 1. This is left as future work.

G Proofs of $\mathsf{HIBE}^{\mathsf{SC}}$

In this section we prove the security of $\mathsf{HIBE}^{\mathsf{SC}}$.

G.1 Ciphertext Identity Indifferentiability

As a stepping stone to proving fixed-epoch ciphertext anonymity (G.2), we present an intermediate property of our scheme which we call *ciphertext identity indifferentiability*. (Ciphertext identity indifferentiability is defined only with respect to our scheme, and is not a general property of HIBE with searchable ciphertexts.) In the ciphertext identity indifferentiability game, the adversary provides an identity vector to its challenger, and the challenger either returns a valid ciphertext as in our scheme or a ciphertext in which the third component c_2 is sampled at random.

The indifferentiability game is presented in Figure 12. The adversary $\mathcal{A}$ has access to an oracle $\mathcal{O}^{\mathsf{KeyDer}}$ which answers KeyDer queries on the sampled public parameters. The challenger tracks the set of identities $\vec{\mathsf{ID}}$ queried by $\mathcal{A}$ to the KeyDer oracle and the game aborts if $\mathcal{A}$ challenges an identity that it has already queried. $\mathcal{A}$ also has access to H_1 and H_2 and can query them n_{id} and n_t times respectively.

$\mathcal{A}$'s advantage is defined as $\mathsf{Adv}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 2\left|\Pr[\mathsf{Expt}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$.

Lemma 1 (Ciphertext Identity Indifferentiability). *Let $n_{kd} = \mathsf{poly}(\lambda)$ be the number of queries that $\mathcal{A}$ may make to $\mathcal{O}^{\mathsf{KeyDer}}$ and $e = 2.71828\ldots$. If $\mathcal{A}$ is an adversary for the identity indifferentiability game with advantage $\mathsf{Adv}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{A}}$, then there exists an adversary $\mathcal{B}$ for the decisional bilinear Diffie-Hellman game such that $\mathsf{Adv}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{A}} \leq en_{\mathsf{id}}\mathsf{Adv}^{\mathsf{dbdh}}_{\mathcal{B}}$.*

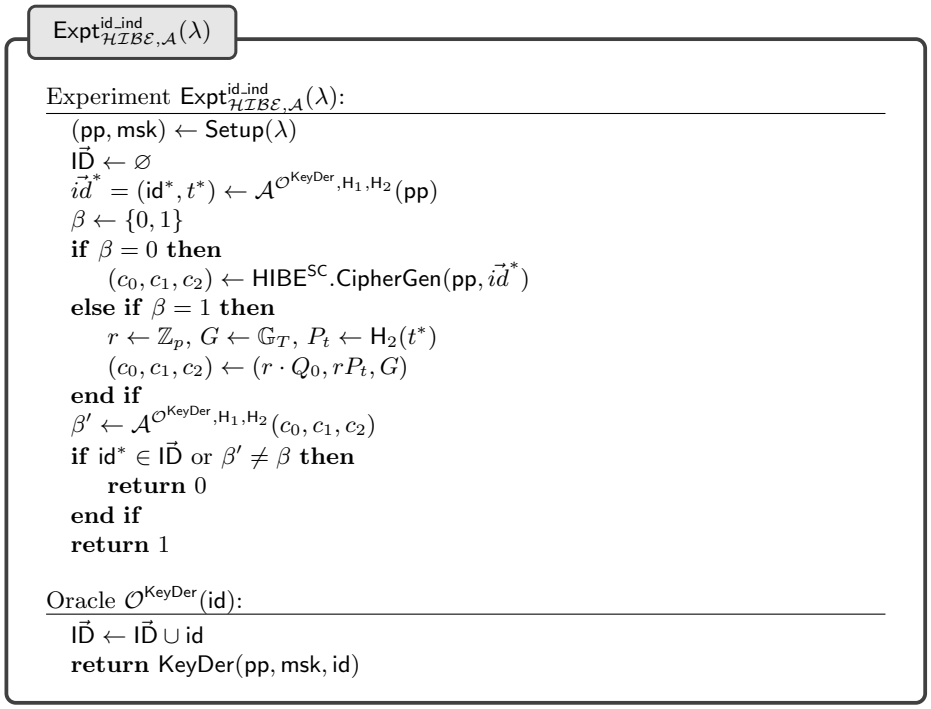


Fig. 12: HIBE Identity Indifferentiability Experiment

Proof Sketch: Recall that $\mathcal{B}$ receives (P, aP, bP, cP, h) as input, and it must decide whether $h = e(P, P)^{abc}$, where P generates $\mathbb{G}_1$ and $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. Intuitively, $\mathcal{B}$ will assign these challenges as follows in its simulation of the $\mathsf{id\_ind}$ game for $\mathcal{A}$:

- P is assigned to the generator Q_0.
- aP is programmed as the hash of the challenge identity $\mathsf{H}_1(\mathsf{id}^*)$.
- bP is assigned as the master public key P_0, which implicitly assigns b to the master secret key.
- cP is used such that c is the ephemeral randomness of the challenge ciphertext.
- h is forwarded to $\mathcal{A}$ as the challenge element c_2. Recall that either $h = e(P, P)^{abc} = e(bP, caP)$ or h is a random element in $\mathbb{G}_T$. If the former, then $\mathcal{B}$ provides $\mathcal{A}$ with a valid ciphertext. If the latter, then $\mathcal{B}$ provides $\mathcal{A}$ with a randomly chosen group element in the position c_2, as in the $\mathsf{id\_ind}$ game.

We adapt the above framework to improve the tightness of the reduction via a technique by Coron [19]. The technique is to guess with probability p (for some p set later) whether $\mathcal{A}$ will query $\mathcal{O}^{\mathsf{KeyDer}}$ for each id that it submits to H_1, and to guess with probability $1 - p$ that the identity will not be sent to $\mathcal{O}^{\mathsf{KeyDer}}$. If $\mathcal{B}$ guesses that id will not be queried to $\mathcal{O}^{\mathsf{KeyDer}}$, then $\mathcal{B}$ responds to the query

with saP, where s is randomly sampled and aP is passed from its own decisional bilinear Diffie-Hellman instance. If $\mathcal{B}$ guesses that the identity will be queried to $\mathcal{O}^{\mathsf{KeyDer}}$, then $\mathcal{B}$ does not embed a in its response, and simply replies with sP for a random s.

Later, when $\mathcal{A}$ issues its challenge (id^*, t), if $\mathcal{B}$ responded with sP to $H(\mathsf{id}^*)$ then $\mathcal{B}$ aborts and the attack fails. However, if $\mathcal{B}$ responded saP to $H(\mathsf{id}^*)$ then $\mathcal{B}$ replies with h^s. Observe that if $h = e(P,P)^{abc}$ then $h^s = e(P,P)^{abcs} = e(bP, acsP) = e(bP, cH(\mathsf{id}^*))$ as required. If $h \neq e(P,P)^{abc}$, then h^s is still a random element of $\mathbb{G}_T$.

Proof. $\mathcal{B}$ first emulates $\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Setup}(\lambda)$ and returns the public parameters to $\mathcal{A}$. $\mathcal{B}$ assigns $\mathbb{G}_1$, $\mathbb{G}_T$, and e as the groups and pairing function for the HIBE scheme, assigns $Q_0 = P$, and sets $P_0 = bP$. (Implicitly this assigns b to the master secret key for the HIBE scheme.)

As in the proof by Abdalla, $\mathcal{B}$ maintains lists L_1, L_2 to store the programmed outputs of the random oracles H_1 and H_2. Without loss of generality, we assume that any id queried to $\mathcal{O}^{\mathsf{KeyDer}}$ is first queried to $\mathsf{H}_1(\mathsf{id})$ and any t returned in $\mathcal{A}$'s challenge is first queried to $\mathsf{H}_2(t)$. If not, then $\mathcal{B}$ simulates those queries before proceeding.

- In response to $\mathsf{H}_1(\mathsf{id})$ queries:
  - With probability p: Sample $s \leftarrow \mathbb{Z}_p^*$, add $(\mathsf{id}, s, \mathsf{no})$ to L_1, and return sP.
  - Otherwise: Sample $s \leftarrow \mathbb{Z}_p^*$, add $(\mathsf{id}, s, \mathsf{yes})$ to L_1, and return saP.
- In response to $\mathsf{H}_2(t)$ queries: Sample $s \leftarrow \mathbb{Z}_p^*$, add (t, s) to L_2 and return sP.
- In response to $\mathsf{KeyDer}(\mathsf{id})$ queries: Find $(\mathsf{id}, s, \mathsf{no})$ in L_1 and return sbP. If $(\mathsf{id}, s, \mathsf{yes})$ is in L_1, then abort.

When $\mathcal{A}$ issues its challenge (id^*, t), if there is a record $(\mathsf{id}^*, s, \mathsf{no})$ in L_1 then $\mathcal{B}$ aborts. If there is a record $(\mathsf{id}^*, s_{\mathsf{id}}, \mathsf{yes})$ in L_1, then $\mathcal{B}$ responds by setting $c_0 = cP$, $c_1 = s_t cP$, and $c_2 = h^{s_{\mathsf{id}}}$, where s_t is looked up in L_2 as the record (t, s_t). If there is no record for id^* in L_1 then $\mathcal{B}$ simulates the query to H_1 with the guess that id^* will not be queried and proceeds as if there were a yes record above.

We must argue that the distribution provided to $\mathcal{A}$ by $\mathcal{B}$ is identically distributed to the $\mathsf{id\_ind}$ game, conditioned on the fact that $\mathcal{B}$ does not abort. Observe that because cP is a random element provided to $\mathcal{B}$, c_0 is identically distributed as in the $\mathsf{id\_ind}$ game, as c_0 is specified by $\mathsf{CipherGen}$ as rQ_0 for random r. Moreover, the distribution of (c_0, c_1) provided to $\mathcal{A}$ by $\mathcal{B}$ is distributed identically in the simulation as in the $\mathsf{id\_ind}$ game, as c_1 is specified by $\mathsf{CipherGen}$ as rP_t; in this game, the randomness of P_t is stored by logging the responses of H_2, and the challenge c takes the place of r, which ensures that the ephemeral randomness of c_1 is consistent with c_0. Finally, c_2 is distributed identically to that of the $\mathsf{id\_ind}$ game. If $\beta = 0$, then $e(P_0, c\mathsf{H}_1(\mathsf{id}^*)) = e(bP, c(as)P) = e(P,P)^{(as)bc} = h^s$, where $\mathsf{H}_1(\mathsf{id}^*)$ is programmed to asP. If $\beta = 1$, then h^s is a uniformly random element, as in the $\mathsf{id\_ind}$ game.

It follows that the simulation constructed by $\mathcal{B}$ for $\mathcal{A}$ is perfect under the condition that $\mathcal{B}$ does not abort due to guessing incorrectly, and therefore conditioned on guessing correctly $\mathcal{B}$ wins its game with the same probability as $\mathcal{A}$.

What remains is to set p. By the analysis of Coron [19], the probability of the reduction succeeding is maximized when $p = 1 - \frac{1}{n_{kd}+1}$, where $n_{kd} = \mathsf{poly}(\lambda)$ is the maximum number of times that $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{KeyDer}}$. It follows as in Coron [19] that $\mathsf{Adv}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{A}} \leq e n_{\mathsf{id}} \mathsf{Adv}^{\mathsf{dbdh}}_{\mathcal{B}}$, where the factor e comes from the standard approximation $\left(1 - \frac{1}{n_{\mathsf{id}}}\right)^{-(n_{\mathsf{id}}+1)} \approx e$ for large n_{id}.

G.2 Fixed-Epoch Ciphertext Anonymity

We use Lemma 1 to reduce fixed-epoch anonymity of our scheme to the decisional bilinear Diffie-Hellman assumption. Specifically, if there exists an adversary $\mathcal{A}$ for the fixed-epoch anonymity game, then there must exist an adversary $\mathcal{B}$ for the identity indifferentiability game.

Lemma 2 (Fixed-Epoch Ciphertext Anonymity). *Let $n_{\mathsf{id}} = \mathsf{poly}(\lambda)$ be the number of queries that $\mathcal{A}$ makes to H_1. If $\mathcal{A}$ is an adversary for the fixed-epoch identity anonymity game, with advantage $\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}$, then there exists an adversary $\mathcal{B}$ for the identity indifferentiability game with advantage $\mathsf{Adv}^{\mathsf{id\_ind}}_{\mathcal{HIBE},\mathcal{B}} \leq \mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{A}}$.*

Sketch. The strategy for $\mathcal{B}$ is simple: it uses $\mathcal{A}$'s ability to distinguish between a ciphertext generated for a chosen identity and a random ciphertext in order to determine whether it has received a ciphertext generated for id_0 or id_1 of its own choosing. Simply, when $\mathcal{A}$ issues a challenge (id^*, t), $\mathcal{B}$ assigns id^* to either id_0 or id_1, samples a random identity for the identity not chosen, and issues its own challenge $(\mathsf{id}_0, \mathsf{id}_1, t)$. The reduction depends on the fact that for two ciphertexts $c = (c_0, c_1, c_2)$ where c is generated with respect to (id_0, t) and $d = (d_0, d_1, d_2)$, where d is generated with respect to (id_1, t), the joint distributions (c_0, c_1) and (d_0, d_1) are identical. Therefore, when $\mathcal{B}$ receives its challenger's response, there are only two cases:

1. $\mathcal{B}$'s challenger returned a ciphertext with respect to (id^*, t).
2. $\mathcal{B}$'s challenger returned a ciphertext with respect to $\mathcal{B}$'s randomly chosen identity.

Because the two cases are distributed identically except for the third element in the ciphertext, the challenge provided to $\mathcal{A}$ is exactly the same as in the $\mathsf{id\_ind}$ game; either it receives a ciphertext generated with respect to its own chosen identity, or it receives a ciphertext with a random c_2.

Proof. Assume there exists an adversary $\mathcal{A}$ for the fixed-epoch identity anonymity game anon. We will construct an adversary $\mathcal{B}$ for the ciphertext indifferentiability game that uses anon as a black box. When $\mathcal{B}$ receives the public

parameters pp from its challenger, it forwards pp to $\mathcal{A}$. $\mathcal{B}$ then proceeds by forwarding all of $\mathcal{A}$'s queries to $\mathcal{B}$'s own oracles: when $\mathcal{A}$ queries H_1 or H_2, $\mathcal{B}$ forwards the queries to its own H_1 or H_2 and returns the result. When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{KeyDer}}$, $\mathcal{B}$ forwards the query to its own oracle $\mathcal{O}^{\mathsf{KeyDer}}$ and returns the result.

‘‘‘ $\mathcal{B}$ additionally builds its own challenge from the challenge issued by $\mathcal{A}$. When $\mathcal{A}$ issues its challenge (id^*, t^*), $\mathcal{B}$ flips a coin β and assigns $\mathsf{id}_\beta = \mathsf{id}^*$. $\mathcal{B}$ samples a random identity $\mathsf{id}_{1-\beta}$ for which it has not yet queried $\mathcal{O}^{\mathsf{KeyDer}}$, and then issues the challenge $(\mathsf{id}_0, \mathsf{id}_1, t^*)$ to its own challenger. When $\mathcal{B}$ receives a ciphertext $c = (c_0, c_1, c_2)$ from its challenger, it forwards the challenge to $\mathcal{A}$. When $\mathcal{A}$ outputs β', $\mathcal{B}$ outputs $\beta \oplus \beta'$.

‘‘‘ The distribution of $\mathcal{A}$'s view when run by $\mathcal{B}$ is identical to its view in the $\mathsf{id\_ind}$ game. Consider that when $\mathcal{B}$'s challenger chooses $1 - \beta$ as its challenge bit, $\mathcal{B}$ receives a ciphertext corresponding to a *random* identity. It follows that c_2 in the challenge $\mathcal{B}$ receives is a uniformly random element from the point of view of $\mathcal{A}$, just as $\mathcal{A}$ would receive in the case that its own challenger selects 1 as its challenge bit. Everything else in the view of $\mathcal{A}$ is identical in the $\mathsf{id\_ind}$ game as in the anon game. Note that the other elements c_0, c_1 are distributed identically in the challenge received by $\mathcal{B}$ as in the challenge received by $\mathcal{A}$, since the ephemeral randomness is consistent between c_0 and c_1 and because $\mathcal{B}$ uses $\mathcal{A}$'s choice of t for its own challenge. It follows that $\mathcal{B}$'s simulation of the $\mathsf{id\_ind}$ game is perfect, and $\mathcal{B}$ wins its game with the same probability as $\mathcal{A}$. ‘‘‘

G.3 Soundness

Soundness of $\mathsf{HIBE}^{\mathsf{SC}}$ follows via a direct reduction to the semantic security of Abdalla et al.'s [3] scheme.

Lemma 3. *If there is an adversary $\mathcal{A}$ for the soundness experiment with advantage $\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}$, then there exists an adversary $\mathcal{B}$ that can decrypt arbitrary ciphertexts of Abdalla's 2-level HIBE with probability $\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{HIBE},\mathcal{A}}$.*

Proof. In order to win the soundness game, $\mathcal{A}$ must generate a token τ for which $\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(\tau, c) = 1$. If there exists an adversary which can do this, then there exists another adversary $\mathcal{B}$ that can decrypt arbitrary ciphertexts from Abdalla's 2-level HIBE scheme [3].

‘‘‘ For convenience in the following discussion, we rename the hash functions H_1 and H_2 from $\mathsf{HIBE}^{\mathsf{SC}}$ to $H_{1,1}$ and $H_{1,2}$, respectively, for consistency with Abdalla's notation. (This also disambiguates H_2, which is a separate random oracle in Abdalla's scheme.) We additionally map Abdalla's variables for the generator point and master public key to our own; we review the notation as follows.

In Abdalla's scheme for two levels, a ciphertext c^{Abd} is a triple $(c_0^{\mathsf{Abd}}, c_1^{\mathsf{Abd}}, c_2^{\mathsf{Abd}})$, where $c_0^{\mathsf{Abd}} = rQ_0$, $c_1^{\mathsf{Abd}} = rH_{1,2}(t)$, and $c_2^{\mathsf{Abd}} = m \oplus H_2(e(rH_{1,1}(\mathsf{id}), P_0))$. In our scheme, a ciphertext c is also a triple, where $c_0 = rQ_0$, $c_1 = rH_{1,2}(t)$, and $c_2 = e(rH_{1,1}(\mathsf{id}), P_0)$. Only the third elements,

c_2^{Abd} and c_2, differ; in our scheme, the third element does not include a message m, and the output of the pairing e is not hashed by H_2.

In $\mathcal{A}$'s soundness game, it receives a KeyDer oracle and access to random oracles. In the reduction, when $\mathcal{A}$ queries its own KeyDer oracle or hash function(s), $\mathcal{B}$ simply forwards the queries to its own corresponding oracle or hash function(s) (for its CPA game), and returns the result. (Note that $\mathcal{B}$ has one hash function more than $\mathcal{A}$ and does not need to query it more than once *after* $\mathcal{A}$ has completed, as explained below).

If $\mathcal{A}$ wins the soundness game, then it must construct $\tau = (S, T)$ for which $e(S, rQ_0) = e(rH_{1,1}(\mathsf{id}), P_0) \cdot e(T, rH_{1,2}(t))$. Recall that in Abdalla's 2-level HIBE, decryption is $c_2^{\mathsf{Abd}} \oplus H_2(\kappa)$, where H_2 is modeled as a random oracle and $\kappa = e(rH_{1,1}(\mathsf{id}), P_0)$. Therefore, in order to recover κ, $\mathcal{B}$ runs $\mathcal{A}$, retrieves $\tau = (S, T)$, and computes

$$\kappa = e(S, rQ_0) \cdot e(T, rH_{1,2}(t))^{-1}.$$

Then $\mathcal{B}$ queries Abdalla's random oracle H_2 on κ and decrypts c_2^{Abd}. Thus $\mathcal{B}$ succeeds with the same probability that $\mathcal{A}$ wins. "'

G.4 Correctness

Lemma 4 (Correctness of HIBE scheme). $\mathsf{HIBE}^{\mathsf{SC}}$ *presented in Figure 2 is correct.*

Proof. Recall that $(c_0, c_1, c_2) = (rQ_0, rP_t, e(P_0, r \cdot P_{\mathsf{id}}))$, and $(S, T) = (\mathsf{msk}P_{\mathsf{id}} + sP_t, sQ_0)$, for randomly sampled r and s. The lemma follows from the fact that the Check function returns 1 if $c_2 \cdot e(T, c_1) = e(c_0, S)$. We expand notation and apply properties of pairings to show the equality.

$$\begin{aligned}
c_2 \cdot e(T, c_1) &= e(P_0, r\mathsf{H}_1(\mathsf{id})) \cdot e(sQ_0, r\mathsf{H}_2(t)) \\
&= e(\mathsf{msk}Q_0, r\mathsf{H}_1(\mathsf{id})) \cdot e(sQ_0, r\mathsf{H}_2(t)) \\
&= e(Q_0, r\mathsf{msk}\mathsf{H}_1(\mathsf{id})) \cdot e(Q_0, rs\mathsf{H}_2(t)) \\
&= e(Q_0, r\mathsf{msk}\mathsf{H}_1(\mathsf{id}) + rs\mathsf{H}_2(t)) \\
&= e(rQ_0, \mathsf{msk}\mathsf{H}_1(\mathsf{id}) + s\mathsf{H}_2(t)) \\
&= e(c_0, S)
\end{aligned}$$

H Proofs of PIB

In this section we prove theorem Theorem 1. Before re-stating and proving the theorem, we begin with two supporting lemmas. The first argues correctness of our scheme, and the second shows that the Query protocol is private up to retrieval-count leakage, which will aid in the full proof.

Lemma 5 (Decodability and Completeness). PIB *is decodable and complete.*

Proof sketch. Completeness follows from correctness of $\mathsf{HIBE}^{\mathsf{SC}}$. Decodability follows from correctness of IBE. We do not present a full formal proof because it follows directly from these properties.

Lemma 6 (Query Privacy with Count Leakage). PIB *is* query private *up to the number of PIR invocations.*

Proof. To prove the lemma, we define a simulator which, on input 1^λ, number of servers s, size of database d, index of a corrupted server $i \in 1, 2$, and the number q of PIR invocations, produces a distribution such that

$$\{\mathcal{S}(1^\lambda, i, s, d, q)\}_{\lambda,i,s,d,q} \approx_c \{\mathsf{View}^{\mathsf{Query}}_{\lambda,s,d}(S_i)\}_{\lambda,i,s,d,q}.$$

First observe that S_1 receives only one message before PIR: a share of a HIBE search token, which is uniformly random. S_1 also receives a message for each invocation of PIR, but by the privacy of PIR this view is simulatable given the number q of invocations. Therefore, $\mathcal{S}$ samples an element at random as S_1's share of the search token S, and then simulates S_1's view of the q PIR executions that follow.

The simulator for S_2 is analogous, substituting only the share of the token provided by the receiver. S_1 and S_2 receive different parts of a receiver's search token (S and T), but conditioned on receiving only one share of the token, each share is uniformly random. The number q is necessary because an execution with a different number of PIR invocations would have a distinguishable transcript length.

Theorem 1 (Security of Bulletin Board Scheme). *If $\mathcal{A}$ is an adversary against bulletin-board privacy for* PIB *in the leakage model of Figure 4, then there exist adversaries $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{E}$ such that*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{pir}}_{\mathcal{PIR},\mathcal{B}}(\lambda) + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{C}}(\lambda) + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{D}}(\lambda) + \mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{E}}(\lambda).$$

Proof. We define a sequence of games for which we will define adversaries that break the underlying primitives of our construction.

- GAME_0: This is the Bulletin Board privacy game ($\mathsf{Expt}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}}$ in Figure 4).
- GAME_1: This is like GAME_0, except that whenever a call to PIR is made on some index, the actual execution in the view of S_1 is replaced with a simulation.
- GAME_2: This is like GAME_1, except that when a record $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ is added to S_1's database, the record c_{hibe} is always replaced with a ciphertext for a random first-level identity at the same public epoch.
- GAME_3: This is like GAME_2, except that for every record $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ added to S_1's database, the ciphertext c_{ibe} is always replaced with an encryption with respect to a random first-level identity.

We now define adversaries for the series of game hops.

Claim. There exists an adversary $\mathcal{B}$ such that

$$\mathsf{Adv}^{\mathsf{pir}}_{\mathcal{PIR},\mathcal{B}} \geq |\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_0] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1]| \,.$$

Proof. In the reduction, $\mathcal{A}$ builds its database of records, with $\mathcal{B}$ answering $\mathcal{A}$'s queries and maintaining the databases. When $\mathcal{A}$ requests public parameters from Setup, $\mathcal{B}$ runs Setup and learns the master keys for both the IBE and HIBE schemes as well. When $\mathcal{A}$ makes a query to one of its oracles, $\mathcal{B}$ simulates the response because it knows the master key. When $\mathcal{A}$ queries one of its random oracles, $\mathcal{B}$ simulates the response. Finally, when $\mathcal{A}$ requests a PIR on its database, $\mathcal{B}$ submits the database to its own PIR challenger, and returns the response to $\mathcal{A}$. If $\mathcal{B}$'s challenger sampled bit 0, then the response provided by $\mathcal{B}$ to $\mathcal{A}$ is the view of a real execution; if $\mathcal{B}$'s challenger sampled bit 1, then it is a simulation. When $\mathcal{A}$ outputs a bit, $\mathcal{B}$ outputs the same bit and distinguishes the two executions with the same probability that $\mathcal{A}$ does.

Claim. There exists an adversary $\mathcal{C}$ such that

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{C}} \geq |\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2]|.$$

Proof. First, observe that the IBE and HIBE scheme instantiations are independent of each other in Figure 5. Therefore, although IBE and HIBE ciphertexts are generated with respect to the same identity, these ciphertexts are independent of each other except for the fact that the HIBE ciphertext is hashed into the plaintext of the IBE ciphertext. This binding does not prevent the HIBE anonymity hybrid, because the IBE component is still generated honestly using the resulting HIBE ciphertext.

To simulate the $\mathsf{PIB.Setup}$ algorithm, $\mathcal{C}$ runs the $\mathsf{IBE.Setup}$ algorithm to generate public parameters and a secret key for the IBE scheme, and uses the output of its own challenger's $\mathsf{HIBE.Setup}$ as the public parameters for the HIBE scheme. When $\mathcal{A}$ makes a query that involves the IBE scheme, $\mathcal{C}$ uses its knowledge of the IBE master key in order to respond correctly. When $\mathcal{A}$ makes a query that involves the HIBE scheme, $\mathcal{C}$ forwards the query to its own corresponding oracle.

Specifically:

- When $\mathcal{A}$ makes a query id to $\mathcal{O}^{\mathsf{KeyDer}}$, $\mathcal{C}$ forwards the identity for the HIBE part of the response to its own KeyDer oracle, while running KeyDer internally for the IBE part of the response using its knowledge of the master key.
- When $\mathcal{A}$ makes queries to the HIBE random oracles H_1 and H_2, $\mathcal{C}$ forwards them to its own HIBE random oracles. When $\mathcal{A}$ makes queries to its IBE random oracles, $\mathcal{C}$ simulates them internally.
- When $\mathcal{A}$ makes queries to $\mathcal{O}^{\mathsf{Encode}}$, $\mathcal{C}$ generates both IBE and HIBE ciphertexts internally using its knowledge of the public keys and access to the random oracles.
- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Reconstruct}}$, $\mathcal{C}$ responds by using its knowledge of the IBE master key to faithfully run $\mathsf{Decrypt}$ on the IBE part of the submitted ciphertext. It performs all checks on the submitted ciphertext as necessary

using its knowledge of the decryption. If verification in $\mathsf{Reconstruct}$ passes with respect to c_{hibe}, then $\mathcal{C}$ forwards the plaintext to $\mathcal{A}$.
- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Query}}$, $\mathcal{C}$ runs $\mathcal{O}^{\mathsf{Query}}$ as if it is S_2 and also provides $\mathcal{A}$ with a random token-share S; by Lemma 6, this element is random and $\mathcal{A}$'s view is indistinguishable from a real execution given the retrieval count.
- When $\mathcal{A}$ inputs (id^*, m^*) to Test for public epoch t^*, $\mathcal{C}$ samples a random first-level identity id' and inputs $(\mathsf{id}^*, \mathsf{id}', t^*)$ to its own fixed-epoch HIBE anonymity challenger. $\mathcal{C}$ forwards the response HIBE ciphertext to $\mathcal{A}$ as the HIBE component of the challenge record, and generates the IBE component honestly using the same HIBE ciphertext in the bound plaintext.

Notice that when $\mathcal{C}$'s challenger's bit $\beta = 0$, the HIBE part of the challenge ciphertext response returned to $\mathcal{A}$ is generated with respect to id^* at the public epoch t^*, and $\mathcal{A}$'s environment is identical to GAME_1. When $\mathcal{C}$'s challenger's bit $\beta = 1$, the HIBE part of the challenge ciphertext returned to $\mathcal{A}$ is generated with respect to a random first-level identity at the same public epoch t^*, and $\mathcal{A}$'s environment is identical to GAME_2.

When $\mathcal{A}$ outputs its guess β', $\mathcal{C}$ outputs the same bit. It follows that $\mathcal{C}$ wins its game with the same probability that $\mathcal{A}$ distinguishes GAME_1 from GAME_2.

Claim. There exists an adversary $\mathcal{D}$ such that

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{D}} \geq |\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3]|.$$

Proof. $\mathcal{D}$ is constructed similarly to $\mathcal{C}$ above, except that $\mathcal{D}$ is an adversary for an IBE scheme, so it forwards all oracle calls involving IBE to its own IBE oracle. For $\mathcal{A}$'s oracle calls that involve HIBE, $\mathcal{D}$ first generates all HIBE parameters on its own by calling $\mathsf{HIBE.Setup}$ and uses its knowledge of the master key to correctly respond to all queries. In addition, $\mathcal{D}$ generates every HIBE ciphertext with respect to a random first-level identity at the same public epoch, maintaining the invariant established in GAME_2.

Specifically:

- When $\mathcal{A}$ makes a query id to $\mathcal{O}^{\mathsf{KeyDer}}$, $\mathcal{D}$ forwards the identity for the IBE part of the response to its own KeyDer oracle, while running KeyDer internally for the HIBE part of the response using its knowledge of the master key.
- When $\mathcal{A}$ makes queries to the HIBE random oracles H_1 and H_2, $\mathcal{D}$ internally simulates the random oracles. When $\mathcal{A}$ makes queries to its IBE random oracles, $\mathcal{D}$ forwards them to its own random oracles.
- When $\mathcal{A}$ makes queries to $\mathcal{O}^{\mathsf{Encode}}$, $\mathcal{D}$ generates both IBE and HIBE ciphertexts internally using its knowledge of the public keys and access to the random oracles, but always generates the HIBE part of the ciphertext with respect to a random first-level identity while preserving the public epoch.
- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Reconstruct}}$ with $c = (c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ for a non-challenge identity, $\mathcal{D}$ obtains the corresponding IBE secret key through its own KeyDer oracle, decrypts c_{ibe} locally, verifies that $\mathsf{H}(c_{\mathsf{hibe}})$ is equal to the first part of the plaintext, and if valid returns the response to $\mathcal{A}$. Otherwise it returns $\perp$.

- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Query}}$, $\mathcal{D}$ responds as in the description of $\mathcal{C}$ above.
- When $\mathcal{A}$ inputs (m^*, id^*) to its $\mathcal{O}^{\mathsf{Test}}$ oracle for public epoch t^*, $\mathcal{D}$ samples another identity id' at random, and also samples a HIBE ciphertext c_{hibe} for a random first-level identity at the same public epoch t^*. $\mathcal{D}$ outputs $(\mathsf{id}^*, \mathsf{id}', \mathsf{H}(c_{\mathsf{hibe}})||m^*)$ to its own IBE anonymity challenger. $\mathcal{D}$ receives the response ciphertext c_{ibe} and returns $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ to $\mathcal{A}$ to complete the response.

When $\mathcal{D}$'s challenger's bit $\beta = 0$, the IBE part of the challenge ciphertext response returned to $\mathcal{A}$ is generated with respect to id^*, and $\mathcal{A}$'s environment is identical to GAME_2. Note that in both cases, the ciphertext c_{hibe} is for a random first-level identity at the same public epoch. When $\mathcal{D}$'s challenger's bit $\beta = 1$, the IBE part of the challenge ciphertext returned to $\mathcal{A}$ is generated with respect to a random first-level identity, and $\mathcal{A}$'s environment is identical to GAME_3.

When $\mathcal{A}$ outputs its guess β', $\mathcal{D}$ outputs the same bit. It follows that $\mathcal{D}$ wins its game with the same probability that $\mathcal{A}$ distinguishes GAME_2 from GAME_3.

Claim. There exists an adversary $\mathcal{E}$ such that

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{E}} \geq 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3] - \frac{1}{2}\right|.$$

Proof. $\mathcal{E}$ works similarly to the above adversaries $\mathcal{C}$ and $\mathcal{D}$, but it always responds to queries by $\mathcal{A}$ with ciphertexts that have been chosen for random first-level identities. At this point all identity information has already been randomized in previous hybrids.

Specifically:

- When $\mathcal{A}$ makes a query id to $\mathcal{O}^{\mathsf{KeyDer}}$, $\mathcal{E}$ forwards the identity for the IBE part of the response to its own KeyDer oracle, while running KeyDer internally for the HIBE part of the response using its knowledge of the master key.
- When $\mathcal{A}$ makes queries to the HIBE random oracles H_1 and H_2, $\mathcal{E}$ internally simulates the random oracles. When $\mathcal{A}$ makes queries to its IBE random oracles, $\mathcal{E}$ forwards them to its own random oracles.
- When $\mathcal{A}$ makes queries to $\mathcal{O}^{\mathsf{Encode}}$, $\mathcal{E}$ generates both IBE and HIBE ciphertexts internally using its knowledge of the public keys and access to the random oracles, but always generates an encoding with respect to random first-level identities for both the HIBE and IBE parts of the ciphertext.
- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Reconstruct}}$ with $c = (c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ for a non-challenge identity, $\mathcal{E}$ obtains the corresponding IBE secret key through its own KeyDer oracle and decrypts c_{ibe} locally. Let $m' = h||m$, where h is the same length as the hash output of H. If $h = \mathsf{H}(c_{\mathsf{hibe}})$, then $\mathcal{E}$ forwards m to $\mathcal{A}$. Otherwise, it replies $\perp$ to $\mathcal{A}$.
- When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{Query}}$, $\mathcal{E}$ responds as in the description of $\mathcal{C}$ above.
- When $\mathcal{A}$ inputs (m^*, id^*) to its $\mathcal{O}^{\mathsf{Test}}$ oracle for public epoch t^*, $\mathcal{E}$ samples another message m' at random. It also samples a HIBE ciphertext c_{hibe} for a random first-level identity at the same public epoch t^*. $\mathcal{E}$ computes $\widetilde{m^*} = \mathsf{H}(c_{\mathsf{hibe}})||m^*$ and $\widetilde{m'} = \mathsf{H}(c_{\mathsf{hibe}})||m'$, and outputs $(\mathsf{id}^*, \widetilde{m^*}, \widetilde{m'})$ to its own IND-CPA challenger. $\mathcal{E}$ receives the response ciphertext c_{ibe} and forwards the pair $(c_{\mathsf{ibe}}, c_{\mathsf{hibe}})$ to $\mathcal{A}$.

When $\mathcal{A}$ outputs β', $\mathcal{E}$ outputs the same bit. Notice that when $\mathcal{E}$'s challenger's bit $\beta = 0$, the challenge ciphertext response returned to $\mathcal{A}$ encrypts $\widetilde{m^*}$. When $\mathcal{E}$'s challenger's bit $\beta = 1$, the challenge ciphertext returned to $\mathcal{A}$ encrypts $\widetilde{m'}$, which contains the same HIBE hash prefix and a random payload message. Therefore the environment that $\mathcal{E}$ constructs for $\mathcal{A}$ is identical to GAME_3 with either the real message or a random message. It follows that $\mathcal{E}$ wins its IND-CPA game with the same probability that $\mathcal{A}$ wins GAME_3.

The proof concludes from composition of the claims:

$$
\begin{aligned}
\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{BB},\mathcal{A}} &= 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_0] - \frac{1}{2}\right| \\
&= 2\Big|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_0] - \frac{1}{2} \\
&\qquad + \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1] \\
&\qquad + \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2] \\
&\qquad + \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3]\Big| \\
&\leq 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_0] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1]\right| \\
&\quad + 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_1] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2]\right| \\
&\quad + 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_2] - \Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3]\right| \\
&\quad + 2\left|\Pr[\mathcal{A} \text{ wins } \mathsf{GAME}_3] - \frac{1}{2}\right| \\
&\leq 2\mathsf{Adv}^{\mathsf{pir}}_{\mathcal{PIR},\mathcal{B}} + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{HIBE},\mathcal{C}} \\
&\quad + 2\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{IBE},\mathcal{D}} + \mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE},\mathcal{E}}.
\end{aligned}
$$

I Multi-Server $\mathcal{PIB}^3$

The $\mathcal{PIB}^3$ scheme presented in Section 8 can be generalized to a multi-server scheme by distributing the search token to more than two parties. The algorithmic modification to generate tokens for N parties is presented in Figure 13, via an additional argument to both the Token and Check algorithms.

The protocol PIB.Query that implements HIBE.Check must additionally be adapted to N parties, and requires an N-party PIR scheme. The adapted protocol is presented in Figure 14.

Proof sketches for correctness and privacy are presented below as direct extensions of Lemma 4 and Theorem 1. The privacy statement is conditional on the natural coalition-resilience properties of the token sharing and PIR subroutines.

Correctness. The difference between the multi-server protocol and the two-server version is the distribution of the relevant part of the token into many pieces, and the computation performed by S_2 in the two-server protocol is now distributed across $S_2, \ldots, S_N$. To show correctness, it is enough to show that $\prod_{j=2}^{N} y_i^j$ yields

$\mathsf{HIBE}^{\mathsf{SC}}$ Token Modification

$\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Token}(\mathsf{pp}, \mathsf{sk}, \vec{id}, N)$

$(Q_0, P_0) = \mathsf{pp.mpk}$
$(\mathsf{id}, t) = \vec{id}$
$P_t \leftarrow \mathsf{H}_2(t)$
$s_1, \ldots, s_{N-1} \leftarrow \mathbb{Z}^* * p$
$s \leftarrow \sum * i = 1^{N-1} s_i$
$S \leftarrow \mathsf{sk} + sP_t \; // = \mathsf{msk}P_{\mathsf{id}} + sP_t$
for $i \in [1, \ldots, N-1]$ **do**
  $T_i \leftarrow s_i Q_0$
end for
return $(S, T_1, \ldots, T_{N-1})$

$\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}(\mathsf{pp}, \tau, c, N)$

$e = \mathsf{pp}.e$
$(S, T_1, \ldots, T_{N-1}) = \tau$
$(c_0, c_1, c_2) = c$
$x \leftarrow e(S, c_0)$
$y \leftarrow c_2 \cdot \prod_{i=1}^{N-1} e(T_i, c_1)$
return 1 if $x = y$, 0 otherwise

Fig. 13: Multi-token Modified Two-Level $\mathsf{HIBE}^{\mathsf{SC}}$ with searchable ciphertexts

the same value y_i used in the two-party protocol, where the second server computes $y_i = c_{2,i} \cdot e(T, c_{1,i})$. The proof follows from bilinearity. For a searchable ciphertext $c = (c_0, c_1, c_2)$, the computation performed by R after receiving messages from S_2 through S_N is:

Protocol 2 Multi-Server Query Protocol $\mathsf{PIB.Query}$

Common Input

- IBE public parameters $\mathsf{pp}_{\mathsf{ibe}}$
- HIBE public parameters $\mathsf{pp}_{\mathsf{hibe}}$

Party Inputs

- Every server S_j has a list of M records $C = [C_1, \ldots, C_M]$, where $C_i = (C^{\mathsf{IBE}} * i, C^{\mathsf{HIBE}} * i)$, and $C^{\mathsf{HIBE}} * i = (c * 0, i, c * 1, i, c * 2, i)$.
- Receiver R has identity id, a secret key $\mathsf{sk} = (\mathsf{sk}_{\mathsf{ibe}}, \mathsf{sk}_{\mathsf{hibe}})$, and an epoch t.

Protocol

1. R generates a search token for epoch t:

$$(S, T_1, \ldots, T_{N-1}) \leftarrow \mathsf{HIBE.Token}(\mathsf{pp}_{\mathsf{hibe}}, \mathsf{sk}_{\mathsf{hibe}}, (\mathsf{id}, t), N).$$

   R sends S to S_1, and for $j \in [2, \ldots, N]$ sends T_{j-1} to S_j.
2. For every searchable ciphertext $C_i^{\mathsf{HIBE}} = (c_{0,i}, c_{1,i}, c_{2,i})$ in its list, S_1 computes $x_i = e(S, c_{0,i})$. For every searchable ciphertext $C_i^{\mathsf{HIBE}} = (c_{0,i}, c_{1,i}, c_{2,i})$ in its list, S_2 computes $y_i^2 = c_{2,i} \cdot e(T_1, c_{1,i})$. For every searchable ciphertext $C_i^{\mathsf{HIBE}} = (c_{0,i}, c_{1,i}, c_{2,i})$ in its list, each server S_j for $j \in [3, \ldots, N]$ computes $y_i^j = e(T_{j-1}, c_{1,i})$.
3. S_1 sends $X = [x_1, \ldots, x_M]$ to R. Each S_j for $j \neq 1$ sends $Y_j = [y_1^j, \ldots, y_M^j]$ to R. R computes $y_i = \prod_{j=2}^{N} y_i^j$, and then computes $I = \{i : x_i = y_i\}$.
4. R invokes N-server $\mathcal{F}^{\mathsf{pir}}$ with all servers to retrieve the IBE ciphertexts at the indices in I.

Outputs R outputs the records whose search ciphertexts were generated with respect to its public key and epoch. Every server S_j outputs $\perp$.

Fig. 14: Modified N-Server Query Protocol

$$
\begin{aligned}
\prod_{j=2}^{N} y_i^j &= y_i^2 \cdot \prod_{j=3}^{N} y_i^j \\
&= c_{2,i} \cdot e(T_1, c_{1,i}) \cdot \prod_{j=3}^{N} e(T_{j-1}, c_{1,i}) \\
&= c_{2,i} \cdot e(T_1, c_{1,i}) \cdot \prod_{k=2}^{N-1} e(T_k, c_{1,i}) \\
&= c_{2,i} \cdot \prod_{k=1}^{N-1} e(T_k, c_{1,i}) \\
&= c_{2,i} \cdot \prod_{k=1}^{N-1} e(s_k Q_0, c_{1,i}) \\
&= c_{2,i} \cdot \prod_{k=1}^{N-1} e(Q_0, c_{1,i})^{s_k} \\
&= c_{2,i} \cdot e(Q_0, c_{1,i})^{\sum_{k=1}^{N-1} s_k} \\
&= c_{2,i} \cdot e(Q_0, c_{1,i})^{s} \\
&= c_{2,i} \cdot e(sQ_0, c_{1,i}).
\end{aligned}
$$

This is the same value used in the two-party retrieval protocol and in $\mathsf{HIBE}^{\mathsf{SC}}.\mathsf{Check}$.

Privacy. The multi-server construction supports a stronger statement than the single-corrupted-server claim, provided the underlying PIR scheme and the token-sharing method are analyzed against the same coalition.

Let $\mathcal{C} \subsetneq S_1, \ldots, S_N$ be a strict coalition of corrupted servers. We say that the multi-server query protocol has *coalition-simulatable token shares* for $\mathcal{C}$ if the joint distribution of the token shares received by the servers in $\mathcal{C}$ is simulatable from the public parameters and the public epoch t, without knowing the receiver identity id. For the token sharing in Figure 13, this property holds for every strict coalition $\mathcal{C}$: if $S_1 \notin \mathcal{C}$, the coalition sees only independently random values $T_i = s_i Q_0$; if $S_1 \in \mathcal{C}$ but some $S_j \notin \mathcal{C}$ for $j \geq 2$, then the missing value $s_{j-1} P_t$ statistically masks $S = \mathsf{sk} + sP_t$ from the coalition.

Theorem 2 (Conditional Multi-Server Coalition Privacy). *Assume that the N-server PIR protocol used in Figure 14 is private against a coalition $\mathcal{C}$, and that the PIR simulator is given the number $q = |I|$ of PIR invocations. Then the N-server $\mathcal{PIB}^3$ protocol is private against the coalition $\mathcal{C}$ with leakage consisting of the public epoch, the database size, and the retrieval count q. In particular, if the underlying N-server PIR scheme is private against every coalition of size at most $N-1$, then the multi-server $\mathcal{PIB}^3$ protocol is private against any coalition of at most $N-1$ colluding servers, subject to the same leakage.*

Proof sketch. The simulator for a coalition $\mathcal{C}$ proceeds as in the two-server proof, but simulates the joint view of all servers in $\mathcal{C}$. Before the PIR phase, the coalition's query-specific view consists of its token shares and the partial check values computed from those shares. By the coalition-simulatable token-share property described above, the simulator can sample the coalition's token shares without knowing the receiver identity id, conditioned only on the public epoch t and public parameters. Given these simulated shares and the public database, the simulator can compute the same partial-check messages that the corrupted servers would compute.

For the retrieval phase, the coalition observes $q = |I|$ PIR executions. By the assumed coalition privacy of the N-server PIR scheme, the joint PIR view of the corrupted servers in $\mathcal{C}$ is simulatable given q. Thus the entire query transcript observed by $\mathcal{C}$ is simulatable from the stated leakage.

The remaining ciphertext-privacy hybrids are unchanged from the proof of Theorem 1. The HIBE ciphertext hybrid uses fixed-epoch anonymity and therefore preserves the public epoch. The IBE identity hybrid uses CPA anonymity, and the payload hybrid uses IND-CPA security. Consequently, any coalition satisfying the stated PIR condition learns no information beyond the public epoch, database size, and retrieval count. If an application also requires hiding the retrieval count, the multi-server protocol must pad retrievals to a fixed bound and specify how overflow is handled.