Oblivious File Retrieval with Preprocessing: A Preliminary Analysis

Thang Hoang, Hoang-Dung (Thomas) Nguyen

Virginia Tech

Abstract

TBD

1 Introduction

2 Preliminaries

Notation.

Pseudorandom Set. For efficient storage and substitution of set elements in our scheme, we use a pseudorandom set (PRS). Given that the set elements are indices from DB with $\sqrt{N}$ partitions, our PRS is constructed from $\mathsf{PRF} : \{0,1\}^\lambda \times [\sqrt{N}] \to [\sqrt{N}]$ with the following algorithms $\mathsf{PRS} = (\mathsf{Gen}, \mathsf{Eval})$:

- $\underline{\mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)}$: It outputs a PRF key $\mathsf{sk} \xleftarrow{\$} \{0,1\}^\lambda$
- $\underline{\mathcal{S} \leftarrow \mathsf{Eval}(\mathsf{sk}, Y)}$: Given a PRF key $\mathsf{sk} \in \{0,1\}^\lambda$ and an auxiliary $Y = (y_1, \ldots, y_t)$, it computes a set $\mathcal{S} = (s_0, \ldots, s_{\sqrt{N}-1})$ such that $s_j = j\sqrt{N} + \mathsf{PRF}(\mathsf{sk}, j)$ for $j \in [\sqrt{N}]$ and replaces some elements by the auxiliary as $s_{\gamma_i} = y_i$, where $\gamma_i = \lfloor y_i/\sqrt{N} \rfloor$ for $i \in [t]$.

3 Models

System Model. We consider a data storage platform with a client and a storage server. The server maintains the database DB of N entries, each being of size B. The client can access an arbitrary entry in DB. Our system can be considered as a Read-Only Oblivious RAM (RORAM) defined as follows.

Definition 1. RORAM scheme is a tuple of PPT algorithms $\mathsf{RORAM} = (\mathsf{Setup}, \mathsf{Read}, \mathsf{Answer}, \mathsf{Refresh})$:

- $\underline{(\mathcal{H}, \mathsf{E_L}, \mathsf{E_R}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)}$: Given a database DB with N as the number of entries, it outputs a private hint $\mathcal{H}$ and two encrypted databases $\mathsf{E_L}$, $\mathsf{E_R}$.

- $\underline{(Q_\mathsf{L}, Q_\mathsf{R}, \mathsf{st}) \leftarrow \mathsf{Read}(x, \mathcal{H})}$: Given a desired index x and the private hint $\mathcal{H}$, it outputs two read queries Q_L for $\mathsf{E_L}$ and Q_R for $\mathsf{E_R}$, and a client state st.
- $\underline{(\mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R}) \leftarrow \mathsf{Answer}(\mathsf{E_L}, \mathsf{E_R}, Q_\mathsf{L}, Q_\mathsf{R})}$: Given two read queries Q_L, Q_R and the two databases $\mathsf{E_L}$, $\mathsf{E_R}$, it outputs two responses $\mathcal{R}_\mathsf{L}$ and $\mathcal{R}_\mathsf{R}$.
- $\underline{(b_x, \mathcal{H}') \leftarrow \mathsf{Refresh}(\mathsf{st}, \mathcal{H}, \mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R})}$: Given the client state st, the private hint $\mathcal{H}$ and two responses $\mathcal{R}_\mathsf{L}$, $\mathcal{R}_\mathsf{R}$, it outputs the decrypted entry $b_x = \mathsf{DB}[x]$ and an updated hint $\mathcal{H}'$.

Definition 2 (RORAM Correctness). A single-server RORAM scheme is correct if for any database DB and $(\mathcal{H}, \mathsf{E_L}, \mathsf{E_R}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$, given security parameter λ and an unbound number of prior queries, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for any index $x \in [N]$:

$$\Pr\left[b_x \neq \mathsf{DB}[x] \;\middle|\; \begin{array}{r} (Q_\mathsf{L}, Q_\mathsf{R}, \mathsf{st}) \leftarrow \mathsf{Read}(x, \mathcal{H}) \\ (\mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R}) \leftarrow \mathsf{Answer}(\mathsf{E_L}, \mathsf{E_R}, Q_\mathsf{L}, Q_\mathsf{R}) \\ (b_x, \mathcal{H}') \leftarrow \mathsf{Refresh}(\mathsf{st}, \mathcal{H}, \mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R}) \end{array} \right] \leq \mathsf{negl}(\lambda)$$

Threat Model. In our system, the client is trusted. We consider the server untrusted, in which it attempts to infer what data entry the client is trying to retrieve. We consider the semi-honest adversarial model, in which the adversary is curious about the client's query, but will follow the protocol faithfully.

Security Model. We define the security of our scheme against the aforementioned threat model using the Ideal/Real paradigm, such that an adversary $\mathcal{A}$ corrupting one server learns nothing about the entry being retrieved. Let $\mathcal{F}$ be the RORAM's ideal functionality that answers the client query honestly as described in XXX. Let $\mathcal{S}$ be an ideal simulator that emulates the view of the real-world adversary. Let $\mathcal{Z}$ be the environment that provides inputs for all entities and receives corresponding outputs. $\mathcal{Z}$ can get any adversarial views at any time. We define the Ideal/Real world as follows:

- Ideal: In the setup, on receiving (DB, N), $\mathcal{F}$ notifies $\mathcal{S}$ about the size N of DB (but not its content). $\mathcal{S}$ emulates the adversarial view of the setup execution and replies to $\mathcal{F}$ with ok or $\perp$. In the online, on receiving an index $x \in [N]$, $\mathcal{F}$ notifies $\mathcal{S}$ about the event (but not x). $\mathcal{S}$ emulates the adversarial view of the online execution and replies to $\mathcal{F}$ with ok or $\perp$. If $\mathcal{S}$ says ok, $\mathcal{F}$ returns the entry $\mathsf{DB}[x]$.
- Real: In the setup, on receiving (DB, N), the client honestly executes $(\mathcal{H}, \mathsf{E_L}, \mathsf{E_R}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$ with the server to obtain a client hint $\mathcal{H}$, and two encrypted databases ($\mathsf{E_L}$, $\mathsf{E_R}$). In the online, on receiving an index x, the client honestly executes $(Q_\mathsf{L}, Q_\mathsf{R}, \mathsf{st}) \leftarrow \mathsf{Read}(x, \mathcal{H})$ and sends read queries $(Q_\mathsf{L}, Q_\mathsf{R})$ to the server. The server returns $(\mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R}) \leftarrow \mathsf{Answer}(\mathsf{E_L}, \mathsf{E_R}, Q_\mathsf{L}, Q_\mathsf{R})$. The client obtains outputs b_x by executing $(b_x, \mathcal{H}') \leftarrow \mathsf{Refresh}(\mathsf{st}, \mathcal{H}, \mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R})$.

Definition 3 (RORAM Security). Any protocol $\Pi_\mathcal{F}$ is said to be secure in realizing $\mathcal{F}$ if for every PPT real adversary $\mathcal{A}$, there is a PPT simulator $\mathcal{S}$, such that for all non-uniform, polynomial-time environment $\mathcal{Z}$, the following distributions are computationally indistinguishable:

$$|\Pr[\textsc{Real}_{\Pi_\mathcal{F}, \mathcal{A}, \mathcal{Z}}(\lambda) = 1] - \Pr[\textsc{Ideal}_{\mathcal{F}, \mathcal{S}, \mathcal{Z}}(\lambda) = 1]| \leq \mathsf{negl}(\lambda)$$

[[[[[]]]]]

```
• $\mathcal{F}$.Setup(DB, $N$):
1: store DB and $N$
```

```
• $(b_x) \leftarrow \mathcal{F}$.Read($x$):
1: if $x \in [N]$ then
2:     return DB$[x]$
3: else
4:     return $\perp$
```

Figure 1: RORAM's Ideal Functionality

4 Technical Roadmap

4.1 OO-PIR on Public Storage (Piano)

Our idea is inspired by a single-server OO-PIR blueprint, called Piano, from XXX, which contains two phases: offline and online. The server maintains a public database DB with N entries. DB is divided into $\sqrt{N}$ partitions $(P_0, \ldots, P_{\sqrt{N}-1})$, with P_j covers indices in the range $(j\sqrt{N}, \ldots, (j+1)\sqrt{N}-1)$ for $j = XX, \ldots, YYY$.

In the offline phase, the client precomputes a private structure $\mathcal{H}$ containing $M = O(\sqrt{N}\log N)$ hints of aggregated DB entries, such that the use of each hint only permits one private retrieval from DB but in $O(\sqrt{N})$ server computation. To create $\mathcal{H}$, the client first defines $\mathcal{H} = (h_1, \ldots, h_M)$, where hint $h_i = (\mathcal{S}_i, \rho_i)$ contains a set of random DB indices $\mathcal{S}_i = \{s_0^{(i)}, \ldots, s_{\sqrt{N}-1}^{(i)}\}$ and a parity $\rho_i = \bigoplus_{j=0}^{\sqrt{N}-1} \mathsf{DB}[s_j^{(i)}]$, where $\bigoplus$ denotes the logical-XOR operation. To generate each set $\mathcal{S}_i$, the client uses a PRF key sk_i and locally sample each index $s_j^{(i)} \leftarrow j\sqrt{N} + \mathsf{PRF}(\mathsf{sk}_i, j)$. For each parity ρ_i, the client needs to compute the XOR aggregation $(\mathsf{DB}[s_1^{(i)}] \oplus \ldots \oplus \mathsf{DB}[s_{\sqrt{N}-1}^{(i)}])$ in a private manner to preserve the private structure of $\mathcal{H}$. Since DB is publicly maintained by the server, the client thus needs to stream the entire database DB to privately compute M parity ρ_i. For each obtained entry $\mathsf{DB}[\gamma]$ per partition P_j, the client performs an $O(1)$ membership test on each set $\mathcal{S}_i$ to determine if index $\gamma \in \mathcal{S}_i$ by checking if $\gamma = j\sqrt{N} + \mathsf{PRF}(\mathsf{sk}_i, j)$. If $\gamma \in \mathcal{S}_i$ is true, the client updates $\rho_i \leftarrow \rho_i \oplus \mathsf{DB}[\gamma]$. In total, the client needs $O(N)$ offline bandwidth cost to stream the entire database. To compute the private structure $\mathcal{H}$, the client performs $O(N\sqrt{N}\log N)$ membership tests and $O(N\log N)$ XOR operations. Since $\mathcal{H}$ contains $O(\sqrt{N}\log N)$ hints and each hint is non-reusable, each offline execution can support $Q = O(\sqrt{N}\log N)$ retrieval.

In the online phase, the client can privately retrieve a desired entry $\mathsf{DB}[x]$ by consuming any hint $h_i = (\mathcal{S}_i, \rho_i)$ such that $x \in \mathcal{S}_i$. Finding the required hint h_i costs $O(M)$ as membership test costs $O(1)$ per hint and there are $M = O(\sqrt{N}\log N)$ hints. To recover $\mathsf{DB}[x]$ from parity ρ_i, the client needs a puncture parity $\hat{\rho}$ of the puncture set $\mathcal{S}_z i \setminus \{x\}$, with $\hat{\rho} = \bigoplus_{j=0}^{\sqrt{N}-2} \mathsf{DB}[\hat{s}_j]$ and $\hat{s}_j \in \mathcal{S}_i \setminus \{x\}$. However, trivially sending $\mathcal{S}_i \setminus \{x\}$ to obtain the punctured parity $\hat{\rho}$ will reveal the partition P_k of index x, as each index $s_j^{(i)}$ in the original set $\mathcal{S}_i$ is previously sampled as $s_j^{(i)} \leftarrow j\sqrt{N} + \mathsf{PRF}(\mathsf{sk}_i, j)$ and belong to a distinct partition P_j with $j \in [N]$. To securely obtain $\hat{\rho}$, the solution is to puncture and patch $\mathcal{S}_i$ as

$\mathcal{S}_i \setminus \{x\} \cup \{z\}$ using a random index z from the same partition P_k of x. In this case, the server can only observe $\sqrt{N}$ random indices, each from a distinct $\sqrt{N}$ partition and cannot infer which partition is of client interest. The client obtains a patched parity $\bar{\rho} = \hat{\rho} \oplus \mathsf{DB}[z]$. Assuming if $\mathsf{DB}[z]$ is already existed locally, the client can retrieve $\hat{\rho} = \bar{\rho} \oplus \mathsf{DB}[z]$ and use parity ρ_i from hint h_i to recover $\mathsf{DB}[x] = \rho_i \oplus \hat{\rho}$. Thus, to support this patching strategy, Piano makes use of the database streaming process in the offline phase to obtain and store $O(\log N)$ random entries per partition, where each of these random entries can only be used once for patching. To evenly balance the use of all patching entries across $\sqrt{N}$ partitions so that no partition running out of patch before $Q = O(\sqrt{N} \log N)$ queries, Piano assumes the client only executes random and distinct queries over a window of Q queries. The randomness of queries is ensured by publicly shuffling the database DB upfront according to a pseudorandom permutation, so that the partition for puncturing (and patching) per consumed hint is independent of the client query. The distinctness of queries is ensured by maintaining a stash of size Q at the client side to store the most recent Q query results for duplicate suppression.

4.2 Private Hint Retrieval with O(1) Bandwidth

In contrast to previous works [?], in this paper we focus on encrypted storage (as traditionally considered in the ORAM setting).

4.3 Data Structure for Encrypted Storage

Given a client database DB of N *encrypted* entries, the data structure of RORAM is as follows:

- Client: The client maintains a private hint structure $\mathcal{H} = (h_1, \ldots, h_M)$, where each hint h_i represents a set of index $\mathcal{S}_i = (s_0, \ldots, s_{n-1})$ that has an offline parity $\rho_i = \sum_{j=0}^{\sqrt{N}-1} \mathsf{DB}[s_j]$.
- Server: The server maintains two encrypted buffers $(\mathsf{E_L}, \mathsf{E_R})$ of size N. The buffer $\mathsf{E_L}$ is a replicate of DB. The buffer $\mathsf{E_R}$ is a permuted version of $\mathsf{E_L}$ according to some random permutation $\pi \overset{\$}{\leftarrow} \mathbb{S}_N$. Each buffer is divided into $\sqrt{N}$ partitions. Partition P_k covers indices in the range $[k\sqrt{N} \ldots (k+1)\sqrt{N} - 1]$. We denote $P_k = \big(k\sqrt{N}, \ldots, (k+1)\sqrt{N} - 1\big)$.

We define a permutation space $\mathbb{S}_N = \{\pi : [N] \rightarrowtail\twoheadrightarrow [N]\}$, containing all permutation π that ensures for all pair $(i, j) \in N^2$, if $\lfloor \frac{i}{\sqrt{N}} \rfloor = \lfloor \frac{j}{\sqrt{N}} \rfloor$ then $\lfloor \frac{\pi(i)}{\sqrt{N}} \rfloor = \lfloor \frac{\pi(j)}{\sqrt{N}} \rfloor$. Intuitively, if both index $(i, j) \in P_k$ in buffer $\mathsf{E_L}$, then there exists a partition $P_{k'}$ from buffer $\mathsf{E_R}$ that contains both permuted index $(\pi(i), \pi(j))$.

4.4 Private Read Protocol for Encrypted Storage

```
• $(\mathcal{H}, \mathsf{E_L}, \mathsf{E_R}) \leftarrow \mathsf{RORAM.Setup}(\mathsf{DB}, N)$:
1: for $i = 1$ to $N$ do
2:     $\mathsf{E_L}[i] \leftarrow \mathsf{F}(\mathbf{k}_i, \mathbf{k}_\mathsf{L}) + \mathsf{DB}[i]$
3:     $\mathsf{E_R}[i] \leftarrow \mathsf{F}(\mathbf{k}_i, \mathbf{k}_\mathsf{R}) + \mathsf{DB}[\pi(i)]$
4: for $i = 1$ to $M = \sqrt{N}\log N$ do
5:     $\mathsf{sk}_i \leftarrow \mathsf{PRS.Gen}(1^\lambda)$, $\mathbf{e}_i \xleftarrow{\$} \{0,1\}^{\sqrt{N}}$
6:     $(s_0, \ldots, s_{\sqrt{N}-1}) \leftarrow \mathsf{PRS.Eval}(\mathsf{sk}_i, \perp)$
7:     $\rho_i \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{DB}[s_j] \;\; \forall\, \mathbf{e}_i[j] = 0$
8:     $\hat{\rho}_i \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{DB}[s_j] \;\; \forall\, \mathbf{e}_i[j] = 1$
9:     $h_i \leftarrow (\mathsf{sk}_i, \mathbf{e}_i, \rho_i, \hat{\rho}_i, Y_i)$ with $Y_i \leftarrow \perp$
10: $\mathcal{H} \leftarrow (h_1, \ldots, h_M)$
11: return $(\mathcal{H}, \mathsf{E_L}, \mathsf{E_R})$
```

Figure 2: Offline Phase

```
• $(Q_\mathsf{L}, Q_\mathsf{R}, \mathsf{st}) \leftarrow \mathsf{RORAM.Read}(x, \mathcal{H})$:
1: parse $\mathcal{H} = (h_1, \ldots, h_M)$
2: search $h_i = (\mathsf{sk}_i, \mathbf{e}_i, \cdot)$ where $x \in \mathcal{S}_i \leftarrow \mathsf{PRS.Eval}(\mathsf{sk}_i, Y_i)$
3: $\bar{\mathcal{S}} \leftarrow \mathcal{S}_i \setminus \{x\} \cup \{z\}$ where $z \xleftarrow{\$} P_k$ with $k = \lfloor \frac{x}{\sqrt{N}} \rfloor$
4: $\bar{\mathcal{S}} \leftarrow \{\pi(s) \mid s \in \bar{\mathcal{S}}\}$
5: $Q_\mathsf{R} \leftarrow (\bar{\mathcal{S}}, \mathbf{e}_i)$ with $\mathbf{e}_i[k] = \mathbf{e}_i[k] \oplus 1$
6: $(s^*_0, \ldots, s^*_{\sqrt{N}-1}) \leftarrow \mathsf{PRS.Eval}(\mathsf{sk}^*, \perp)$ with $\mathsf{sk}^* \leftarrow \mathsf{PRS.Gen}(1^\lambda)$
7: for $j \in [\sqrt{N}]$ do
8:     $\mathbf{s}_j = (\bar{s}_0, \ldots, \bar{s}_\gamma, \ldots, \bar{s}_{\kappa-1})$ with $\mathbf{s}_j \xleftarrow{\$} P_j \wedge \bar{s}_\gamma = s^*_j$
9:     $\mathbf{q}_j \leftarrow \mathsf{SSPIR.Query}(\mathbf{s}_j, \gamma)$
10: $Q_\mathsf{L} \leftarrow (\mathbf{q}_0, \ldots, \mathbf{q}_{\sqrt{N}-1}, \mathbf{e}^*)$ with $\mathbf{e}^* \xleftarrow{\$} \{0,1\}^{\sqrt{N}}$
11: $\mathsf{st} \leftarrow (z, \mathsf{sk}^*, \mathbf{e}^*)$
12: return $(Q_\mathsf{L}, Q_\mathsf{R}, \mathsf{st})$
```

Figure 3: Online Phase: Read

```
• $(\mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R}) \leftarrow \mathsf{RORAM.Answer}(\mathsf{E}_\mathsf{L}, \mathsf{E}_\mathsf{R}, Q_\mathsf{L}, Q_\mathsf{R})$:
  1: parse $Q_\mathsf{R} = (s_0, \ldots, s_{\sqrt{N}-1}), (e_0, \ldots, e_{\sqrt{N}-1})$
  2: $\rho \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{E}_\mathsf{R}[s_j] \ \ \forall\, e_j = 0$
  3: $\hat{\rho} \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{E}_\mathsf{R}[s_j] \ \ \forall\, e_j = 1$
  4: parse $Q_\mathsf{L} = (\mathbf{q}_0, \ldots, \mathbf{q}_{\sqrt{N}-1}), (e_0^*, \ldots, e_{\sqrt{N}-1}^*)$
  5: $\rho^* \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{SSPIR.Answer}(\mathbf{q}_j, \mathsf{E}_\mathsf{L}) \ \ \forall\, e_j^* = 0$
  6: $\hat{\rho}^* \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{SSPIR.Answer}(\mathbf{q}_j, \mathsf{E}_\mathsf{L}) \ \ \forall\, e_j^* = 1$
  7: return $(\rho^*, \hat{\rho}^*), (\rho, \hat{\rho})$
```

Figure 4: Online Phase: Answer

```
• $(b_x, \mathcal{H}) \leftarrow \mathsf{RORAM.Refresh}(\mathsf{st}, \mathcal{H}, \mathcal{R}_\mathsf{L}, \mathcal{R}_\mathsf{R})$:
  1: parse $\mathcal{H} = (h_1, \ldots, h_i, \ldots, h_M)$, $h_i = (\mathsf{sk}_i, \mathbf{e}_i, \rho_i, \hat{\rho}_i, Y_i)$
  2: parse $\mathcal{R}_\mathsf{R} = (\rho, \hat{\rho})$, $\mathcal{R}_\mathsf{L} = (\rho^*, \hat{\rho}^*)$, $\mathsf{st} = (z, \mathsf{sk}^*, \mathbf{e}^*)$
Recover Item:
  3: $\sigma \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{F}(\mathbf{k}_j, \mathbf{k}_\mathsf{R}) \ \ \forall\, \mathbf{e}_i[j] = 0$
  4: $\hat{\sigma} \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{F}(\mathbf{k}_j, \mathbf{k}_\mathsf{R}) \ \ \forall\, \mathbf{e}_i[j] = 1$
  5: if $\mathbf{e}_i[k] = 0$ then
  6:     $(b_z, b_x) \leftarrow (\rho - \rho_i - \sigma, \hat{\rho}_i - \hat{\rho} - \hat{\sigma})$
  7: else
  8:     $(b_x, b_z) \leftarrow (\rho_i - \rho - \sigma, \hat{\rho} - \hat{\rho}_i - \hat{\sigma})$
Random Hint:
  9: $\sigma^* \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{F}(\mathbf{k}_j, \mathbf{k}_\mathsf{L}) \ \ \forall\, \mathbf{e}^*[j] = 0$
 10: $\hat{\sigma}^* \leftarrow \sum_{j=0}^{\sqrt{N}-1} \mathsf{F}(\mathbf{k}_j, \mathbf{k}_\mathsf{L}) \ \ \forall\, \mathbf{e}^*[j] = 1$
 11: $\rho^* \leftarrow \mathsf{SSPIR.Recover}(\rho^*) - \sigma^*$
 12: $\hat{\rho}^* \leftarrow \mathsf{SSPIR.Recover}(\hat{\rho}^*) - \hat{\sigma}^*$
 13: $h^* \leftarrow (\mathsf{sk}^*, \mathbf{e}^*, \rho^*, \hat{\rho}^*, Y^*)$ with $Y^* \leftarrow \perp$
Replace Hint:
 14: search $h_j = (\mathsf{sk}_j, \mathbf{e}_j, \rho_j, \hat{\rho}_j, Y_j)$ where $z \in \mathsf{PRS.Eval}(\mathsf{sk}_j, Y_j)$
 15: if $\mathbf{e}_j[k] = 0$ then
 16:     $\rho_j \leftarrow \rho_j + b_z - b_x$
 17: else
 18:     $\hat{\rho}_j \leftarrow \hat{\rho}_j + b_z - b_x$
 19: $h'_j \leftarrow (\mathsf{sk}_j, \mathbf{e}_j, \rho_j, \hat{\rho}_j, Y'_j)$ with $Y'_j \leftarrow Y_j \cup \{x\} \setminus \{z\}$
 20: return $\mathcal{H} \leftarrow (h_1, \ldots, h'_j, \ldots, h^*, \ldots, h_M)$
```

Figure 5: Online Phase: Refresh

```
$\mathsf{QP}^{\mathsf{single}}_{\mathcal{A},\Pi}(\mathsf{DB}, N)$:
1: $(\mathcal{H}, \mathbf{E}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$
2: for $i = 1$ to $M$ do
3:     $Q_i^* \leftarrow \mathsf{HintQuery}(N)$
4:     $(\mathcal{R}_i, \mathcal{H}) \leftarrow \mathsf{Answer}(Q_i^*, \mathcal{H}, \mathbf{E})$
5: $(x_0, x_1) \leftarrow \mathcal{A}(Q_1^*, \ldots, Q_M^*)$
6: $b \xleftarrow{\$} \{0, 1\}$
7: $Q \leftarrow \mathsf{ReadQuery}(x_b, \mathcal{H})$
8: $(\mathcal{R}, \mathcal{H}') \leftarrow \mathsf{Answer}(Q, \mathcal{H}, \mathbf{E})$
9: $b' \leftarrow \mathcal{A}(Q_1^*, \ldots, Q_M^*, Q)$
10: If $b = b'$, output 1. Else, output 0.
```

Figure 6: Game $\mathsf{QP}^{\mathsf{single}}_{\mathcal{A},\Pi}$

```
$\mathsf{QP}^{\mathsf{multi}}_{\mathcal{A},\Pi}(\mathsf{DB}, N)$:
1: $(\mathcal{H}, \mathbf{E}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$
2: for $i = 1$ to $M$ do
3:     $Q_i^* \leftarrow \mathsf{HintQuery}(N)$
4:     $(\mathcal{R}_i, \mathcal{H}) \leftarrow \mathsf{Answer}(Q_i^*, \mathcal{H}, \mathbf{E})$
5: $\mathsf{st}_0 \leftarrow \mathcal{A}(Q_1^*, \ldots, Q_M^*)$
6: $b \xleftarrow{\$} \{0, 1\}$
7: for $i = 1$ to $t$ do
8:     $(x_i^0, x_i^1) \leftarrow \mathcal{A}(\mathsf{st}_{i-1})$
9:     $Q_i \leftarrow \mathsf{ReadQuery}(x_i^b, \mathcal{H})$
10:    $(\mathcal{R}_i, \mathcal{H}) \leftarrow \mathsf{Answer}(Q_i, \mathcal{H}, \mathbf{E})$
11:    $\mathsf{st}_i \leftarrow \mathcal{A}(\mathsf{st}_{i-1}, Q_i)$
12: $b' \leftarrow \mathcal{A}(\mathsf{st}_t)$
13: If $b = b'$, output 1. Else, output 0.
```

Figure 7: Game $\mathsf{QP}^{\mathsf{multi}}_{\mathcal{A},\Pi}$

5 Security Analysis

5.1 OO-PIR Security Definition

Our starting point is a single-server OO-PIR protocol $\Pi = (\mathsf{Setup}, \mathsf{HintQuery}, \mathsf{ReadQuery}, \mathsf{Answer})$:

- $(\mathcal{H}, \mathbf{E}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$: Given a DB of size N, it setups a private hint $\mathcal{H}$ and a replica $\mathbf{E} = \mathsf{DB}$.
- $Q^* \leftarrow \mathsf{HintQuery}(N)$: Given an index domain $[N]$, it outputs a hint query Q^*.
- $Q \leftarrow \mathsf{ReadQuery}(x, \mathcal{H})$: Given an index x and hint $\mathcal{H}$, it outputs a read query Q.
- $(\mathcal{R}, \mathcal{H}') \leftarrow \mathsf{Answer}(\tilde{Q}, \mathcal{H}, \mathbf{E})$: Given $\tilde{Q} \in (Q, Q^*)$, hint $\mathcal{H}$, replica $\mathbf{E}$, it returns $\mathcal{R}$ and updates hint $\mathcal{H}'$.

Definition 4 (Single-Query OO-PIR Security). Consider the 1-query experiment $\mathsf{QP}^{\mathsf{single}}_{\mathcal{A},\Pi}(\mathsf{DB}, N)$ in Figure 6 with a stateful PPT adversary $\mathcal{A}$ and the protocol Π. An OO-PIR is *1-query secure* if:

$$\Pr\left[\mathsf{QP}^{\mathsf{single}}_{\mathcal{A},\Pi}(\mathsf{DB}, N) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

Definition 5 (Multi-Query OO-PIR Security). Consider the t-query experiment $\mathsf{QP}^{\mathsf{multi}}_{\mathcal{A},\Pi}(\mathsf{DB}, N)$ in Figure 7 with a stateful PPT adversary $\mathcal{A}$ and the protocol Π. An OO-PIR is *t-query secure* if:

$$\Pr\left[\mathsf{QP}^{\mathsf{multi}}_{\mathcal{A},\Pi}(\mathsf{DB}, N) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

We remark that Piano follows the OO-PIR definition $\Pi = (\mathsf{Setup}, \mathsf{HintQuery}, \mathsf{ReadQuery}, \mathsf{Answer})$. In the offline phase, the client executes $(\mathcal{H}, \mathbf{E}) \leftarrow \mathsf{Setup}(\mathsf{DB}, N)$ with the server, where the client receives database parameter N from the server to set up the structure for private hint $\mathcal{H}$. The server

```
$\mathsf{RS}^{\mathsf{t=1,n=1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$:
1: $x \xleftarrow{\$} \mathcal{S}$
2: $(y, \mathbf{R}) \leftarrow \mathcal{O}(x, \mathbf{L})$
3: $x' \leftarrow \mathcal{A}(y, \mathcal{S}, \mathbf{R}, \mathbf{L})$
4: If $x = x'$, output 1. Else, output 0.
```

Figure 8: Game $\mathsf{RS}^{\mathsf{t=1,n=1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$

```
$\mathsf{RS}^{\mathsf{t=1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$:
1: $\mathbf{x} \xleftarrow{\$} \mathcal{S}, \mathbf{x} = (x_1, \ldots, x_n)$
2: for $i = 1 \rightarrow n$ do
3:     $(y_i, \mathbf{R}) \leftarrow \mathcal{O}(x_i, \mathbf{L})$
4: $\mathbf{y} \leftarrow (y_1, \ldots, y_n)$
5: $\mathbf{x}' \leftarrow \mathcal{A}(\mathbf{y}, \mathcal{S}, \mathbf{R}, \mathbf{L})$
6: If $\mathbf{x} = \mathbf{x}'$, output 1. Else, output 0.
```

Figure 9: Game $\mathsf{RS}^{\mathsf{t=1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$

in Piano does nothing as the replica $\mathbf{E} = \mathsf{DB}$ is already created and maintained by the server due to the public database setting. The client then executes $Q^* \leftarrow \mathsf{HintQuery}(N)$ a number of times to create multiple hint queries. In Piano, each hint query Q^* is basically just a command signal (index i) to retrieve a database entry $\mathsf{DB}[i]$ in a deterministic manner. Upon each signal index i, the client executes $(\mathcal{R}, \mathcal{H}') \leftarrow \mathsf{Answer}(\tilde{Q} = Q^*, \mathcal{H}, \mathbf{E})$ to receive $\mathsf{DB}[i]$ and perform a linear scan on the defined hint structure $\mathcal{H}$ to aggregate $\mathsf{DB}[i]$ into whichever hint entry contains $\mathsf{DB}[i]$. In the online, on receiving a desired index x, the client executes $Q \leftarrow \mathsf{ReadQuery}(x, \mathcal{H})$ to find a hint entry that contains $\mathsf{DB}[x]$ and use it to generate a retrieval query Q. The client then executes $(\mathcal{R}, \mathcal{H}') \leftarrow \mathsf{Answer}(\tilde{Q} = Q, \mathcal{H}, \mathbf{E})$ with the server to recover $\mathsf{DB}[x]$ and update the private hint $\mathcal{H}$ by discarding the consumed hint entry.

To instantiate a new private retrieval protocol, our initial idea is to adjust the single-server OO-PIR protocol Piano with a small tweak. Instead of having the server maintain one (public) replica of the database $\mathbf{E} = \mathsf{DB}$ as in Piano, we now let the server maintain two (*encrypted*) replicas $(\mathsf{E_L}, \mathsf{E_R})$ of DB, which makes the server state becomes $\mathbf{E} = (\mathsf{E_L}, \mathsf{E_R})$. The two replicas can be viewed as two uniformly random and independent domain spaces at the beginning, but possess a secret relation. The problem of single-server OO-PIR on any two indistinguishable but related spaces can be described as a random selection game.

5.2 Random Selection Game

We start by considering a random selection game RS, where adversary $\mathcal{A}$ has access to two domain spaces $(\mathbf{L}, \mathbf{R})$ that are uniformly independent from each other, with $|\mathbf{L}| = |\mathbf{R}| = N$. The purpose of game RS is to challenge the adversary $\mathcal{A}$ in guessing whether an element $y \in \mathbf{R}$ corresponds to which element $x \in \mathbf{L}$. We define an oracle $\mathcal{O} : \mathbf{L} \longrightarrow \mathbf{R}$ such that upon receiving input $(x, \mathbf{L})$, the oracle outputs a distinct $(y, \mathbf{R}) \leftarrow \mathcal{O}(x, \mathbf{L})$. It is clear that if we sample $x \xleftarrow{\$} \mathbf{L}$, then under the adversary view $\mathcal{A}$ when receiving $(y, \mathbf{R})$, every element $x \in \mathbf{L}$ has an equal probability of $1/N$ to be the correct guess.

Our game RS is special because, instead of sampling $x \xleftarrow{\$} \mathbf{L}$, we sample $x \xleftarrow{\$} \mathcal{S}$, with $\mathcal{S} \leftarrow \mathsf{Gen}(\mathbf{L})$ is a compressed data structure where each entry $\mathbf{x} \in \mathcal{S}$ is a representation of n random elements from $\mathbf{L}$. Moreover, we want to challenge the adversary $\mathcal{A}$ with t adaptive queries in a game RS that invokes the same oracle $\mathcal{O}$. We first describe the game $\mathsf{RS}^{\mathsf{t=1,n=1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ starting with $t = 1$ and $n = 1$.

```
$\mathsf{RS}^{\mathsf{t>1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$:
1: $\mathsf{st}_0 \leftarrow \mathcal{A}(\mathcal{S})$
2: for $j = 1 \to t$ do
3:     $\mathbf{x}_j \xleftarrow{\$} \mathcal{S}, \mathcal{S} \leftarrow \mathcal{S} \setminus \{\mathbf{x}_j\}$
4:     $(\mathbf{y}_j, \mathbf{R}) \leftarrow \tilde{\mathcal{O}}(\mathbf{x}_j, \mathbf{L})$
5:     $\mathsf{st}_j \leftarrow \mathcal{A}(\mathbf{y}_j, \mathsf{st}_{j-1})$
6: $\mathbf{x} \xleftarrow{\$} \mathcal{S}$
7: $(\mathbf{y}, \mathbf{R}) \leftarrow \tilde{\mathcal{O}}(\mathbf{x}, \mathbf{L})$
8: $\mathbf{x}' \leftarrow \mathcal{A}(\mathbf{y}, \mathsf{st}_t)$
9: If $\mathbf{x} = \mathbf{x}'$, output 1. Else, output 0.
```

Figure 10: Game $\mathsf{RS}^{\mathsf{t>1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$

```
$(\mathbf{y}, \mathbf{R}) \leftarrow \tilde{\mathcal{O}}(\mathbf{x}, \mathbf{L})$:
1: parse $\mathbf{x} = (x_1, \ldots, x_n)$
2: $\tilde{\mathbf{x}} = (\tilde{x}_1, \ldots, \tilde{x}_{n'}) \xleftarrow{\$} \mathbf{L}$ with $\mathbf{x} \subset \tilde{\mathbf{x}}$
3: for $i = 1 \to n'$ do
4:     $(y_i, \mathbf{R}) \leftarrow \mathcal{O}(\tilde{x}_i, \mathbf{L})$
5: $\mathbf{y} = (y_1, \ldots, y_{n'})$
6: return $(\mathbf{y}, \mathbf{R})$
```

Figure 11: Extended Oracle

Definition 6 (Single Query - Single Random Selection). Consider experiment $\mathsf{RS}^{\mathsf{t=1,n=1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ in Figure 8 with a stateful PPT adversary $\mathcal{A}$ and a random oracle $\mathcal{O}$. The random selection game is secure if:

$$\Pr\left[\mathsf{RS}^{\mathsf{t=1,n=1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S}) = 1\right] \leq \frac{1}{|\mathcal{S}|} + \mathsf{negl}(\lambda)$$

Next, we want to reduce $|\mathcal{S}|$ by expanding each entry $\mathbf{x} \in \mathcal{S}$ as an vector to represent $|\mathbf{x}| = n > 1$ elements $x_i \in \mathbf{L}$. A simple method is to generate $\mathcal{S}$ where each vector $\mathbf{x} = (x_1, \ldots, x_n)$ contains n distinct $x_i \in \mathbf{L}$. Or we can divide $\mathbf{L}$ into n partitions P_i, each covers $m = N/n$ elements in range $[i \cdot m \ldots (i+1) \cdot m - 1]$ and let each vector $\mathbf{x} = (x_1, \ldots, x_n)$ contains n distinct $x_i \in P_i$. Regardless of the approach, each entry $\mathbf{x}$ now contains n times more elements than before, so the size $|\mathcal{S}|$ can be n times smaller but still allow $\mathcal{S}$ to cover all elements in $\mathbf{L}$ uniformly. Note that the oracle $\mathcal{O}$ still ideally map each elements $x \in \mathbf{L}$ to distinct output $y \in \mathbf{R}$. In Figure 9, we define the next game $\mathsf{RS}^{\mathsf{t=1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$.

Definition 7 (Single Query - Multi Random Selection). Consider experiment $\mathsf{RS}^{\mathsf{t=1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ in Figure 9 with a stateful PPT adversary $\mathcal{A}$ and a random oracle $\mathcal{O}$. The random selection game is secure if:

$$\Pr\left[\mathsf{RS}^{\mathsf{t=1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S}) = 1\right] \leq \frac{1}{|\mathcal{S}|} + \mathsf{negl}(\lambda)$$

Finally, we consider the game $\mathsf{RS}^{\mathsf{t>1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ in Figure 10, with an extended oracle $\tilde{\mathcal{O}} : \mathbf{L} \to \mathbf{R}$ such that, upon receiving a vector $\mathbf{x} = (x_1, \ldots, x_n)$, it outputs a vector $\mathbf{y} = (y_1, \ldots, y_{n'})$ containing $n' > n$ random elements. This simple extension helps us to later prove that there exists a real-world instantiation of $\tilde{\mathcal{O}}$, where $n' = O(n)$ (with small hidden constant), and achieves the security with $t < |\mathcal{S}|$ adaptive queries.

Definition 8 (Multi Query - Multi Random Selection). Consider experiment $\mathsf{RS}^{\mathsf{t>1,n>1}}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ in Figure 10 with a stateful PPT adversary $\mathcal{A}$ and a random oracle $\tilde{\mathcal{O}}$. The random selection game is secure if:

$$\Pr\left[\mathsf{RS}^{\mathsf{t>1,n>1}}_{\mathcal{A},\tilde{\mathcal{O}}}(\mathcal{S}) = 1\right] \leq \frac{1}{|\mathcal{S}|} + \mathsf{negl}(\lambda)$$

5.3 Strawman Protocol Π_{str}

Lemma 1. *Protocol Π_{str} is 1-query secure by Definition 4.*

We need to prove that game $\mathsf{RS}^{t=1,n>1}_{\mathcal{A},\mathcal{O}}(\mathcal{S})$ maintain the

Proof. □

Lemma 2. *Protocol Π_{str} is not t-query secure for $t \geq 2$ by Definition 5.*

Define the Similarity Leakage function $\mathcal{L}(\mathbf{L}, \mathbf{R})$:

- Location Pattern (LP): Given two selection vectors $\mathbf{x}, \mathbf{x}'$ with $\mathsf{LP}(\mathbf{x}, \mathbf{x}', \mathbf{L}) = \{k \in [n] : \mathbf{x}[k] = \mathbf{x}'[k]\}$, there exists another two selection vectors $\mathbf{y}, \mathbf{y}'$ with $\mathsf{LP}(\mathbf{y}, \mathbf{y}', \mathbf{R}) = \{k \in [n] : \mathbf{y}[k] = \mathbf{y}'[k]\}$ such that $\mathsf{LP}(\mathbf{x}, \mathbf{x}', \mathbf{L}) \subseteq \mathsf{LP}(\mathbf{y}, \mathbf{y}', \mathbf{R})$.
- Volume Pattern (VP): Given a volume k, there exists two pairs of vectors $(\mathbf{x}, \mathbf{x}') \in \mathbf{L}$ and $(\mathbf{y}, \mathbf{y}') \in \mathbf{R}$ such that $\mathsf{VP}(k) = |\mathsf{LP}(\mathbf{x}, \mathbf{x}', \mathbf{L})| \leq |\mathsf{LP}(\mathbf{y}, \mathbf{y}', \mathbf{R})|$

A critical statement we can derive from the leakage function is that given a concrete volume pattern VP, an adversary $\mathcal{A}$ can reduce the search space to identify a selection vector $\mathbf{x}$ with higher probability. This is because the space $\mathcal{S}$ where $\mathbf{x}$ belongs to, only has a limited amount of samples, so $\mathcal{A}$ observes some non-uniformity in the distribution of $\mathbf{x}$. For higher volume VP, there are fewer candidates.

Importantly, a location pattern LP between any pair $(\mathbf{x}, \mathbf{x}')$ implies a volume pattern VP, but a volume pattern VP does not yet imply a specific location pattern since multiple LP, and hence multiple pairs $(\mathbf{x}, \mathbf{x}')$, can satisfy the same VP.

The location pattern LP exists whenever we follow the partition model but do not permute the position for partitions. The volume pattern VP will always exist since we are trying to compress from space $\mathbf{L}$ (database indices) into a smaller space $\mathcal{S}$ (client hints) and draw $\mathbf{x} \in \mathcal{S}$ to create $(\mathbf{y}, \mathbf{R}) \longleftarrow \mathcal{O}(\mathbf{x}, \mathbf{L})$. In the next section, we describe how the non-uniformity in $\mathcal{S}$ arises.

5.4 Probability Analysis

Left Analysis. Remark that in the random selection game RS, we are having two indistinguishable but related domain spaces $(\mathbf{L}, \mathbf{R})$ which can be intuitively referred to as (Left, Right). We start with the Left space $\mathbf{L}$, which we use to generate the compressed data structure $\mathcal{S}$ (for client hints). We have $|\mathbf{L}| = N$ which is divided into $n = \sqrt{N}$ partitions P_i, each covers n elements in range $[i \cdot n \ldots (i+1) \cdot n - 1]$. Let $|\mathcal{S}| = M = n \log(n)$, meaning $\mathcal{S}$ contains vectors $\mathbf{x} = (x_1, \ldots, x_n)$ of n distinct element $x_i \in P_i$.

The volume pattern happens whenever there exists two random independent vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{x}' = (x'_1, \ldots, x'_n)$ that shares some common elements. Let X be the random variable that denotes the number of position $i \in [n]$ such that $x_i = x'_i$. Since $\{x_i, x'_i\} \in P_i = [i \cdot n \ldots (i+1) \cdot n - 1]$, we have

$\Pr[x_i = x'_i] = \frac{1}{n}$. Thus, X follows a Binomial distribution:

$$X \approx \mathsf{Binom}(n = n, p = \frac{1}{n}) \approx \mathsf{Poisson}(\lambda = np = 1)$$

This can be approximated with Poisson for an arbitrary large n, so the probability that two random vectors $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{x}' = (x'_1, \dots, x'_n)$ share k common elements is:

$$\Pr[X = k] = \frac{\lambda^k e^{-\lambda}}{k!} = \frac{e^{-1}}{k!}$$

We want to bound k such that $\Pr[X \geq k]$ is a negligible probability $2^{-\kappa}$, for all pair of vectors in $\mathcal{S}$.

$$\begin{aligned}(1 - \Pr[X \geq k])^{\binom{M}{2}} &\geq 1 - 2^{-\kappa} \\ \left(1 - \sum_{i=k}^{\infty} \frac{e^{-1}}{i!}\right) &\geq (1 - 2^{-\kappa})^{\binom{M}{2}^{-1}}\end{aligned}$$

For RHS, use binomial approximation $(1 - x)^a \approx (1 - ax)$ when $|ax| < 1$, we have:

$$\begin{aligned}\left(1 - \sum_{i=k}^{\infty} \frac{e^{-1}}{i!}\right) &\geq 1 - \frac{2^{-\kappa}}{\binom{M}{2}} \\ \sum_{i=k}^{\infty} \frac{1}{i!} &\leq \frac{e \cdot 2^{-\kappa}}{\binom{M}{2}}\end{aligned}$$

For the inequality to hold, where the RHS is extremely small, the LHS needs to decay rapidly, which makes the tail sum also extremely small, dominated by the first term $\frac{1}{k!}$ in LHS:

$$\begin{aligned}\frac{1}{k!} &\leq \frac{e \cdot 2^{-\kappa}}{\binom{M}{2}} \\ k! &\geq \frac{2^{\kappa}}{e}\binom{M}{2} \qquad (1)\end{aligned}$$

Let $k = \underaccent{\sim}{k}$ be the solution for the above inequality. Given concrete $|\mathcal{S}| = M$ and a statistical parameter κ for negligible probability, we can solve the inequality to obtain a lower bound $\underaccent{\sim}{k}$ such that no pair of vectors $(\mathbf{x}, \mathbf{x}') \in \mathcal{S}$ shares more than $\underaccent{\sim}{k}$ common elements with an overwhelming probability.

For example, with $n = 2^{18}$, $M = n \log n = (2^{18} \cdot 18)$ and $\kappa = 40$, we obtain a lower bound $\underaccent{\sim}{k} = 25$. This means for any $k < \underaccent{\sim}{k}$, there is a non-negligible probability that a pair of vectors $(\mathbf{x}, \mathbf{x}')$ with k common elements will exist in $\mathcal{S}$, making the pattern $\mathsf{VP}(k)$ non-uniform (as it rarely happens).

General Case. We can generalize this bound for VP on $t \geq 2$ independent vectors $(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)})$. Let $X^{(t)}$ be the random variable denotes the number of position $i \in [n]$ such that $x_i^{(1)} = \dots = x_i^{(t)}$, i.e. all t vectors agree on the same element at position i. We have $\Pr[x_i^{(1)} = \dots = x_i^{(t)}] = n^{-(t-1)}$.

$$X^{(t)} \approx \mathsf{Binom}(n = n, p = n^{-(t-1)}) \approx \mathsf{Poisson}(\lambda = np = n^{-(t-2)})$$

The probability that t random vectors share k common elements:

$$\Pr[X^{(t)} = k] = \frac{\lambda^k e^{-\lambda}}{k!}$$

We want that no t-tuple of vectors in $\mathbb{S}$ can share more than k elements with overwhelming probability:

$$(1 - \Pr[X^{(t)} \geq k])^{\binom{M}{t}} \geq 1 - 2^{-\kappa}$$
$$\left(1 - \sum_{i=k}^{\infty} \frac{\lambda^i e^{-\lambda}}{i!}\right) \geq (1 - 2^{-\kappa})^{\binom{M}{t}^{-1}}$$

For RHS, use binomial approximation $(1-x)^a \approx (1 - ax)$ when $|ax| < 1$, we have:

$$\left(1 - \sum_{i=k}^{\infty} \frac{\lambda^i e^{-\lambda}}{i!}\right) \geq 1 - \frac{2^{-\kappa}}{\binom{M}{t}}$$
$$e^{-\lambda} \sum_{i=k}^{\infty} \frac{\lambda^i}{i!} \leq \frac{2^{-\kappa}}{\binom{M}{t}}$$

For the inequality to hold, where the RHS is extremely small and $e^{-\lambda} \approx 1$ for small λ, the tail sum in LHS needs to decay rapidly, so the first term $\frac{\lambda^k}{k!}$ becomes the dominant factor. We have:

$$\frac{\lambda^k}{k!} \leq \frac{2^{-\kappa}}{\binom{M}{t}}$$
$$k! \cdot \lambda^{-k} \geq 2^{\kappa} \binom{M}{t}$$
$$k! \cdot n^{k(t-2)} \geq 2^{\kappa} \binom{M}{t} \tag{2}$$

For $t = 2$, this inequality is consistent with Equation 1 and its lower bound solution $k = \underset{\sim}{k}$. We want to characterize the growth of solution $\underset{\sim}{k}$ for increasing $t \geq 2$. Define:

- $\mathsf{LHS}(k, t) = k! \cdot n^{k(t-2)}$
- $\mathsf{RHS}(t) = 2^{\kappa} \binom{M}{t}$

Given that $\binom{M}{t+1} = \binom{M}{t} \frac{M-t}{t+1}$, we have the following relation:

- $\mathsf{LHS}(k, t+1) = k! \cdot n^{k(t+1-2)} = \mathsf{LHS}(k, t) \cdot n^k$
- $\mathsf{RHS}(t+1) = 2^{\kappa} \binom{M}{t+1} = \mathsf{RHS}(t) \cdot \frac{M-t}{t+1}$

Given the lower bound $\underset{\sim}{k}$ when $t = 2$, we have that for arbitrary large n, $M = n \log n$ and increasing $t \geq 2$, $k = \underset{\sim}{k}$ also satisfies $n^k > \frac{M-t}{t+1}$. Thus, the following holds for non-increasing $\underset{\sim}{k}$:

$$\mathsf{LHS}(\underset{\sim}{k}, t) \geq \mathsf{RHS}(t) \implies \mathsf{LHS}(\underset{\sim}{k}, t+1) \geq \mathsf{RHS}(t+1)$$

This means, with respect to the space $\mathcal{S}$ containing M vectors $\mathbf{x} = (x_1, \ldots, x_n)$, we have $\mathscr{k} - 1$ is the maximal volume pattern VP that $\mathcal{A}$ can observe with non-negligible probability. Given a volume k for $k \in [2 \ldots \mathscr{k})$, the closer k approaches $\mathscr{k}$, the higher sparsity we observe on the number of candidate pairs $(\mathbf{x}, \mathbf{x}') \in \mathcal{S}$ that satisfies $\mathsf{VP}(k) = |\mathsf{LP}(\mathbf{x}, \mathbf{x}', \mathbf{L})|$, which we will refer to as extreme cases. Hence, there exists a non-uniformity, with respect to each $\mathsf{VP}(k)$ in the space $\mathcal{S}$.

Right Analysis. Let $\mathcal{S}'$ be the space containing all vectors $\mathbf{y}$ with $(\mathbf{y}, \mathbf{R}) \leftarrow \mathcal{O}(\mathbf{x}, \mathbf{L})$ and $\mathbf{x} \in \mathcal{S}$ that we can apply for random selection on the Right space $\mathbf{R}$. Since $\mathcal{O}$ is an ideal bijective mapping, $\mathcal{S}'$ also observes an identical non-uniformity of $\mathsf{VP}(k)$ from $\mathcal{S}$. Given a pair $(\mathbf{y}, \mathbf{y}')$, $\mathcal{A}$ can make use $\mathsf{VP}(k) = |\mathsf{LP}(\mathbf{y}, \mathbf{y}', \mathbf{R})|$ to reduce the space for candidate $(\mathbf{x}, \mathbf{x}') \in \mathcal{S}$. We want to prevent this space reduction by breaking the volume dependency pattern between $\mathcal{S}$ and $\mathcal{S}'$.

Our idea is to amplify the space $\mathcal{S}'$ by sampling vector $\mathbf{y}$ with $(\mathbf{y}, \mathbf{R}) \leftarrow \tilde{\mathcal{O}}(\mathbf{x}, \mathbf{L})$ where $\tilde{\mathcal{O}}$ is an extended oracle and $\mathbf{y} = (y_1, \ldots, y_{n'})$ containing $n' > n$ random elements. In this analysis, we will use a simplest form of $\tilde{\mathcal{O}}$ that outputs $\mathbf{y}$ with $|\mathbf{y}| = n' = (w \cdot n)$ and contains w elements y_j from each partition $P_i \in \mathbf{R}$. Thus, we rewrite $\mathbf{y} = (y_1, \ldots, y_{n'}) = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n)$, that is the vector $\mathbf{y}$ contains n sub-vectors $\bar{\mathbf{y}}_i$, where $\bar{\mathbf{y}}_i$ has w elements $(y_1^{(i)}, \ldots, y_w^{(i)})$.

Let $\mathbf{y} = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n)$ and $\mathbf{y}' = (\bar{\mathbf{y}}'_1, \ldots, \bar{\mathbf{y}}'_n)$ be two random independent vectors sharing some common elements. We define that $\bar{\mathbf{y}}_i \approx \bar{\mathbf{y}}'_i$ only if there is at least one common element shared between the two sub-vectors $\bar{\mathbf{y}}_i$ and $\bar{\mathbf{y}}'_i$. As $|\bar{\mathbf{y}}_i| = |\bar{\mathbf{y}}'_i| = w$, the probability $\Pr[\bar{\mathbf{y}}_i \approx \bar{\mathbf{y}}'_i]$ is as follows:

$$\Pr[\bar{\mathbf{y}}_i \approx \bar{\mathbf{y}}'_i] = 1 - \left(1 - \frac{w}{n}\right)^w = 1 - e^{-\frac{w^2}{n}} = \frac{w^2}{n}$$

Let Y be the random variable that denotes the number of common positions $i \in [n]$ such that $\bar{\mathbf{y}}_i \approx \bar{\mathbf{y}}'_i$. Thus, Y follows a Binomial distribution:

$$Y \approx \mathsf{Binom}(n = n, p = \frac{w^2}{n})$$

Given a lower bound $k = \mathscr{k}$ on the number of common positions between two random independent vectors $\mathbf{y} = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n)$ and $\mathbf{y}' = (\bar{\mathbf{y}}'_1, \ldots, \bar{\mathbf{y}}'_n)$, we want to find a positive integer w such that $\Pr[Y \geq k]$ is an overwhelming probability of $1 - 2^{-\kappa}$ for all pairs $(\mathbf{y}, \mathbf{y}') \in \mathcal{S}'$:

$$\Pr[Y \geq k]^{\binom{M}{2}} \geq 1 - 2^{-\kappa}$$
$$\Pr[Y \geq k] \geq (1 - 2^{-\kappa})^{\binom{M}{2}^{-1}}$$

For RHS, use binomial approximation $(1 - x)^a \approx (1 - ax)$ when $|ax| < 1$, we have:

$$\Pr[Y \geq k] \geq 1 - \frac{2^{-\kappa}}{\binom{M}{2}}$$
$$\Pr[Y \leq k - 1] \leq \frac{2^{-\kappa}}{\binom{M}{2}}$$

Since RHS is extremely small, we apply the Chernoff lower-tail bound with $\mu = \mathbb{E}[Y] = np = w^2$:

$$\Pr[Y \leq (1-\delta)\mu] \leq \exp\left(\frac{-\mu\delta^2}{2}\right)$$

Define $k - 1 = (1-\delta)\mu \implies \delta = 1 - \frac{k-1}{\mu}$. We have:

$$\begin{aligned}
\exp\left(\frac{-\mu\delta^2}{2}\right) &\leq \frac{2^{-\kappa}}{\binom{M}{2}} \\
\frac{-\mu\delta^2}{2} &\leq \ln(2^{-\kappa}) - \ln\binom{M}{2} \\
\frac{w^2}{2}\left(1 - \frac{k-1}{w^2}\right)^2 &\geq \kappa\ln(2) + \ln\binom{M}{2}
\end{aligned}$$

For example, with $k =$ ƙ, $n = 2^{18}$, $M = n\log n = (2^{18} \cdot 18)$ and $\kappa = 40$, we obtain $w = 13$.

General Case. We can generalize this method for $t \geq 2$ independent vectors $(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t)})$. Let $Y^{(t)}$ be the random variable denotes the number of position $i \in [n]$ such that $\bar{\mathbf{y}}_i^{(1)} \approx \cdots \approx \bar{\mathbf{y}}_i^{(t)}$, i.e. all t sub-vectors at position i agree on at least one common element. We have:

$$\Pr[\bar{\mathbf{y}}_i^{(1)} \approx \cdots \approx \bar{\mathbf{y}}_i^{(t)}] = \frac{w^t}{n^{t-1}} = \tilde{p}$$

Thus, $Y^{(t)} \approx \mathsf{Binom}(n, \tilde{p})$. We want that all t-tuple of vectors $(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t)})$ can agree on at least k positions with overwhelming probability:

$$\begin{aligned}
\Pr[Y^{(t)} \geq k]^{\binom{M}{t}} &\geq 1 - 2^{-\kappa} \\
\Pr[Y^{(t)} \geq k] &\geq 1 - \frac{2^{-\kappa}}{\binom{M}{t}} \\
\Pr[Y^{(t)} \leq k' = k - 1] &\leq \frac{2^{-\kappa}}{\binom{M}{t}}
\end{aligned}$$

Apply Chernoff bound with $\delta = 1 - \frac{k'}{\mu}$ and $\mu = \mathbb{E}[Y^{(t)}] = n\tilde{p} = \frac{w^t}{n^{t-2}}$. We have:

$$\begin{aligned}
\Pr[Y^{(t)} \leq (1-\delta)\mu] \leq \exp\left(\frac{-\mu\delta^2}{2}\right) &\leq \frac{2^{-\kappa}}{\binom{M}{t}} \\
\frac{-\mu\delta^2}{2} &\leq \ln(2^{-\kappa}) - \ln\binom{M}{t} \\
\frac{w^t}{2n^{t-2}}\left(1 - \frac{k' n^{t-2}}{w^t}\right)^2 &\geq \kappa\ln(2) + \ln\binom{M}{t} \\
\frac{(w^t - k' n^{t-2})^2}{2n^{t-2} w^t} &\geq \kappa\ln(2) + \ln\binom{M}{t} \qquad (3)
\end{aligned}$$

We want to characterize the growth of w to satisfy Equation 3 for increasing $t \geq 2$. Define:

- $\mathsf{LHS}(w, t) = \dfrac{(w^t - k' n^{t-2})^2}{2n^{t-2} w^t}$
- $\mathsf{RHS}(t) = \kappa \ln(2) + \ln \binom{M}{t}$

Given that $\binom{M}{t+1} = \binom{M}{t} \frac{M-t}{t+1}$, we have the following relation:

- $\mathsf{LHS}(w, t+1) = \dfrac{(w^{t+1} - k' n^{t-1})^2}{nw \cdot (2n^{t-2} w^t)} = \dfrac{1}{nw} \cdot \dfrac{(w^{t+1} - k' n^{t-1})^2}{(w^t - k' n^{t-2})^2} \cdot \mathsf{LHS}(w, t)$
- $\mathsf{RHS}(t+1) = \kappa \ln(2) + \ln \binom{M}{t+1} = \mathsf{RHS}(t) + \ln \binom{M-t}{t+1}$

Let $G(t) = \dfrac{1}{nw} \cdot \dfrac{(w^{t+1} - k' n^{t-1})^2}{(w^t - k' n^{t-2})^2} = \dfrac{w}{n} \cdot \dfrac{\left(1 - \dfrac{k' n^{t-1}}{w^{t+1}}\right)^2}{\left(1 - \dfrac{k' n^{t-2}}{w^t}\right)^2}$ with $R = \dfrac{k' n^{t-2}}{w^t}$. For arbitrary large n, we have:

$$G(t) = \frac{w}{n} \cdot \left(\frac{1 - nw^{-1} R}{1 - R}\right)^2 \approx \frac{w}{n} \cdot \left(\frac{-nw^{-1} R}{-R}\right)^2 = \frac{n}{w} \gg 1$$

The multiplicative term $G(t)$ makes $\mathsf{LHS}(w, t+1)$ grow faster than $\mathsf{RHS}(t) + \ln \binom{M-t}{t+1}$.

Given the lower bound w when $t = 2$, we have that for arbitrary large n, $M = n \log n$, $k' = \mathcal{k} - 1$ and increasing $t \geq 2$, the following holds for non-increasing w:

$$\mathsf{LHS}(w, t) \geq \mathsf{RHS}(t) \implies \mathsf{LHS}(w, t+1) \geq \mathsf{RHS}(t+1)$$

This means, with respect to space $\mathcal{S}'$ (with amplification factor w) containing M vectors $\mathbf{y} = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n)$, we have $\mathcal{k}$ is the minimal volume pattern VP that $\mathcal{A}$ can observe with overwhelming probability.

Rerandomizable Encryption.

Key-Homomorphic Pseudorandom Function.

6 Security

7 Experiment