# On the Definition of Malicious Private Information Retrieval

Bar Alon
Department of Computer Science
Georgetown University
Washington, DC, USA
alonbar08@gmail.com

Amos Beimel*
Department of Computer Science
Ben Gurion University
Be'er-Sheva, Israel
amos.beimel@gmail.com

April 15, 2025

**Abstract**

A multi-server private information retrieval (PIR) protocol allows a client to obtain an entry of its choice from a database, held by one or more servers, while hiding the identity of the entry from small enough coalitions of servers. In this paper, we study PIR protocols in which some of the servers are malicious and may not send messages according to the pre-described protocol. In previous papers, such protocols were defined by requiring that they are correct, private, and robust to malicious servers, i.e., by listing 3 properties that they should satisfy. However, 40 years of experience in studying secure multiparty protocols taught us that defining the security of protocols by a list of required properties is problematic.

In this paper, we rectify this situation and define the security of PIR protocols with malicious servers using the real vs. ideal paradigm. We study the relationship between the property-based definition of PIR protocols and the real vs. ideal definition, showing the following results:

- We prove that if we require full security from PIR protocols, e.g., the client outputs the correct value of the database entry with high probability even if a minority of the servers are malicious, then the two definitions are equivalent. This implies that constructions of such protocols that were proven secure using the property-based definition are actually secure under the "correct" definition of security.
- We show that if we require security-with-abort from PIR protocols (called PIR protocols with error-detection in previous papers), i.e., protocols in which the user either outputs the correct value or an abort symbol, then there are protocols that are secure under the property-based definition; however, they do not satisfy the real vs. ideal definition, that is, they can be attacked allowing selective abort. This shows that the property-based definition of PIR protocols with security-with-abort is problematic.
- We consider the compiler of Eriguchi et al. (TCC 22) that starts with a PIR protocol that is secure against semi-honest servers and constructs a PIR protocol with security-with-abort; this compiler implies the best-known PIR protocols with security-with-abort. We show that applying this protocol does *not* result in PIR protocols that are secure according to the real vs. ideal definition. However, we prove that a simple modification of this compiler results in PIR protocols that are secure according to the real vs. ideal definition.

*Partially supported by ISF grant 391/21 and by ERC project NFITSC (101097959).

# 1 Introduction

A private information retrieval (PIR) protocol [16] allows a client to obtain an entry of its choice from a database held by one or more servers, such that nothing is revealed to any small enough coalition of servers about the item being revealed. For example, an investor might want to know the value of a specific stock without revealing which stock they are interested in. This is modeled by setting the database to be an $N$-bit string $D$, the client holding a retrieval index $i$, and it wishes to learn the $i^{\text{th}}$ entry of $D$, i.e., $D_i$, without revealing $i$. The trivial solution is to let a single server send the entire database to the client. Chor et al. [16] showed that this is optimal in the information-theoretic setting when there is a single server. PIR protocols were used to construct secure multiparty protocols [30, 9], locally decodable codes [34], and inspired constructions of conditional disclosure of secrets [39, 40]. Most constructions of PIR protocols guarantee that any single server will not learn any information on the client's retrieval index [16, 2, 30, 31, 6, 35, 46, 48, 22, 33, 14, 20, 27, 1], e.g., there is a 3-server PIR protocol secure against a single server with communication complexity $2^{\tilde{O}(\sqrt[3]{\log N})}$ [27]. When there are many servers, it is natural to require that the protocol guarantees the privacy of the client against colluding subsets of the servers. Constructions of such PIR protocols are given in [30, 6, 46, 5]. It is also natural to consider *malicious* servers that may deviate from the protocol. In the stock market example above, one may consider the case where some of the servers try to give the wrong information on the stock. Such PIR protocols were first discussed in [7] and further studied in [47, 46, 28, 18, 49, 43, 44, 37, 4, 24, 23, 25].

The security of PIR protocols in the literature is defined by listing the desired properties from the protocol. However, PIR is a special case of *secure multiparty computation (MPC)*, where it is well-known that separating the security properties is problematic. Indeed, there are MPC protocols that satisfy the natural security properties (such as privacy and correctness) yet should clearly be considered insecure [42, 13, 29]. Instead, the security of MPC protocols is defined using the *real vs. ideal world* paradigm. Intuitively, we consider an ideal process for computing a function via a trusted party that the adversary cannot corrupt. It is then required that any attack performed in the real world can be simulated in the ideal world, where the adversary is more limited. Different ideal processes capture different security properties, e.g., full security and security-with-abort.

Defining malicious security of PIR protocol via properties is problematic and it is unclear if there are other security properties that can be attacked in such protocols. In this work, we rectify this situation and define the security of PIR protocols using the real vs. ideal world paradigm. Specifically, we are interested in the notions of *full security* (also known as guaranteed output delivery) and *security-with-abort*. The former captures the requirement that the client always obtain the "correct" output, and the latter allows the client to abort but never output an "incorrect" value. We ask how such definitions relate to the property-based definitions used so far, and if there are efficient PIR protocols satisfying the real vs. ideal security definition. We consider the perfect setting, the statistical setting, and the computational setting.

## 1.1 Our Contributions

Our conceptual contribution is defining the ideal processes of the PIR functionality for full security and security-with-abort against malicious adversaries (see Section 3 for the formal definitions). We then compare the simulation-based security definition to the property-based security definition.

### 1.1.1 The Ideal World Processes

In this section, we describe the ideal world process for each of the desired security notions. A PIR protocol is then considered secure according to an ideal world if any adversary corrupting a subset of the servers can be simulated in the ideal world. We stress that definitions for PIR protocols do not have any privacy requirements from the client; in particular, the client may learn the whole database. Thus, we only consider adversaries that do not corrupt the client.

**Full security of PIR protocols.** We first describe the definition of fully secure PIR protocols. Roughly, the definition captures the requirement that the client outputs the correct value $D_i$, regardless of the messages sent by the malicious servers. Observe that this only makes sense when the majority of the servers are honest.[1] The ideal world for full security proceeds as follows. The client sends $i$ to the trusted party and each honest server sends $D$ to the trusted party. The corrupted servers may change their databases (based on the instruction of the ideal-world adversary) and send them to the trusted party. Let $D'$ denote the database that was sent by more than $1/2$ of the servers (note that $D' = D$ if there is an honest majority). The trusted party then sends $D'_i$ to the client. The ideal functionality is only defined when there is an honest majority.

**Secure-with-abort PIR protocols.** We next describe the definition of secure-with-abort PIR protocols. Here, we allow the client to output $\perp$ (indicating abort). The security definition captures the requirement that the client never outputs $1 - D_i$. Note that, unlike the ideal world for full security, here the definition allows a majority of corrupted servers.

The ideal-world computation proceeds as follows. The client sends $i$ to the trusted party and each honest server sends $D$ to the trusted party. The corrupted servers may change their databases (based on the instruction of the ideal-world adversary) and send them to the trusted party. If the trusted party receives the same database $D$ from all servers, then it sends $D_i$ to the client; otherwise, it sends $\perp$ to the client.

### 1.1.2 Comparing Simulation-Based Security to Property-Based Security

We compare the simulation-based security definitions to the property-based security definitions. In the following, we fix the total number of servers to be $\ell$, and let $t$ bound the number of corrupted servers. We refer the reader to Section 4 for the formal statements and proofs of the theorems.

**Full security.** We first compare full security with the property-based definition. In addition to correctness and $t$-privacy, here we also consider the notion of *$t$-Byzantine robust*[2] PIR protocols defined by Beimel and Stahl [7]. Roughly, it requires that the client outputs $D_i$ even if $t$ malicious servers arbitrarily deviate from the protocol. Note that similarly to full security, this definition only makes sense when the majority of the servers are honest. As would be expected, full security implies all 3 security properties. We show that the converse also holds.

[1] If the number of servers is $\ell$ and $t$ of them might be corrupted, where $2t \geq \ell$, then the client cannot distinguish an execution with the first $t$ servers being corrupted and behaving honestly on input $0^N$ and the $\ell - t$ remaining honest servers holding $1^N$, from an execution with the last $\ell - t \leq t$ servers being corrupted and behaving honestly on input $1^N$ and the first $t$ servers being honest and holding $0^N$.

[2] Also known as $t$-error-correcting [23].

**Theorem 1.1** (Informal, equivalence of full security and the property-based definition)**.** *Any $\ell$-server PIR protocol is $t$-fully secure if and only if it is correct, $t$-private, and $t$-Byzantine robust.*

Thus, to prove full security it suffices to show that each property holds individually. In particular, this shows that previous Byzantine robust constructions, e.g., [7, 47, 46, 43, 44, 37, 4, 24, 23], are also fully secure.

**Security-with-abort.** Finally, we consider security-with-abort. Here, instead of $t$-Byzantine robustness, we consider the notion of *$t$-error-detecting* PIR protocols defined by Eriguchi et al. [24, 23]. In a $t$-error-detecting PIR protocol, the client can abort (i.e., by outputting $\perp$) but security requires that no set of $t$ malicious servers can force the client to output $1 - D_i$. Clearly, error-detecting is weaker than Byzantine robust; however, it is possible to construct $t$-error-detecting PIR protocols when $t \geq \ell/2$ [24, 23].

Similarly to the previous definitions, the simulation-based definition implies the property-based definition. However, we show that here the converse is *not* true.

**Theorem 1.2** (Informal, security-with-abort and the property-based definition)**.** *Any $\ell$-server PIR protocol that is $t$-secure-with-abort is correct, $t$-private, and $t$-error-detecting. However, the converse does not hold.*

To see why the properties do not imply simulation-based security, consider the following protocol, in which a single malicious server can cause a "selective abort", i.e., the user will abort if and only if $i = 1$. In the protocol, each server sends $D$ to the client and additionally, the first server sends an additional bit $b \in \{0, 1\}$. An honest server sends $b = 0$. If the client received different databases, or if it has input $i = 1$ and it received $b = 1$, then it outputs $\perp$. Otherwise, it receives only the database $D$, and the client outputs $D_i$. Clearly, $\Pi$ is correct, 1-private, and 1-error-detecting. However, it is *not* 1-secure-with-abort. This is because the first server can force a different probability for the client to abort on different indices $i$. Specifically, if the server sends $i = 1$ then the client always aborts on index $i = 1$, and it never aborts on index $i \neq 1$. Note that this holds even though the server does not know anything about $i$. See Example 4.4 for the formal argument. In Example 4.5 we show that even if we require the same probability for abort on all indexes $i$, then it is still insufficient to claim security-with-abort.

Colombo et al. [17] noticed that standard definitions for the security of PIR might allow for selective abort (however, they did not provide any concrete example). This motivated them to define *authenticated PIR* to capture security against such attacks. However, their definition is property-based, and it is unclear whether their definition is equivalent to our simulation-based security, or whether it captures security against attacks not mentioned in their paper.

**A compiler from a private PIR protocol to a secure-with-abort protocol.** Eriguchi et al. [23] showed a compiler that takes any $\ell$-server correct and $t$-private PIR protocol, and transforms it into a correct, $t$-private, and $t$-error-detecting $\ell$-server PIR protocol. In light of Theorem 1.2, it is natural to ask whether their compiler satisfies the stronger requirement of $t$-security-with-abort. We show that this is *not* the case; however, we prove that a simple modification to their construction results in a secure-with-abort protocol.

**Theorem 1.3** (Informal, compiler to secure-with-abort)**.** *There exists a compiler that transforms any $\ell$-server correct and $t$-private PIR protocol $\Pi$ into an $\ell$-server $t$-secure-with-abort PIR protocol.*

*The total communication complexity (i.e., the total number of bits sent) of the new protocol is* $\ell^2 \cdot \omega(\log N) \cdot (c + \ell \log \ell)$*, where* $c$ *is the communication complexity of* $\Pi$*.*

This result implies that for a constant number of servers, the communication complexity of $t$-secure-with-abort PIR protocol is equivalent to PIR protocol secure against $t$-semi-honest servers (up to a factor of $\omega(\log N)$). We refer to Section 5 for the formal statement and proof. In Table 1 we summarize the PIR protocols obtained from our results.

| | $2^t$ server PIR from [5] + [20] | $3^t$ server PIR from [5] + [22] | The $\ell$ server PIR, $\ell = O(1)$ [46] |
|---|---|---|---|
| $t$-private | $2^{\tilde{O}(t\sqrt{\log N})}$ | $\max\left\{t \cdot 2^{\tilde{O}(\sqrt{\log N})}, 2^{O(t)}\right\}$ | $N^{1/\lfloor(2\ell-1)/t\rfloor}$ |
| $t$-security-with-abort | $2^{\max\left\{\tilde{O}(t\sqrt{\log N}), t\log t\right\}}$ | $3^{2t} \cdot \max\left\{t2^{\tilde{O}(\sqrt{\log N})}, t3^t\right\}$ | $N^{\frac{1}{\lfloor(2\ell-1)/t\rfloor}+o(1)}$ |
| $t$-full security | $2^{t^2+\max\left\{\tilde{O}(t\sqrt{\log N}), t\log t\right\}}$ | $3^{t^2+2t} \cdot \max\left\{t2^{\tilde{O}(\sqrt{\log N})}, 2^{O(t)}\right\}$ | $N^{\frac{1}{\lfloor(2\ell-1)/t\rfloor-2}+o(1)}$ |

Table 1: A summary of the best PIR protocols tolerating $t$ corrupted servers for various parameters. The results in the second row are obtained by applying our compiler on the protocols from the first row. These results are also obtained by [23] for the weaker property-based security. The last row follows from [23] and Theorem 1.1.

### 1.2 Our Techniques

To illustrate our techniques, we next show that $t$-semi-honest security is equivalent to correctness and $t$-privacy. Since simulation-based security is clearly stronger than property-based security, we only show that correctness and $t$-privacy imply $t$-semi-honest security. We do it for statistical security (other cases are handled similarly). This is a special case of a known result that the property-based definition is equivalent to the simulation-based definition for deterministic functionalities with security against unbounded adversaries.

Fix a real-world adversary $\mathcal{B}$ corrupting a set $\mathcal{I}$ of at most $t$ servers. We define its simulator $\mathsf{Sim}$ running the client on index $i = 1$, and outputting the queries that correspond to the corrupted servers. We now show that the statistical distance between the real and ideal worlds is negligible. For a retrieval index $i \in [N]$ and a database $D \in \{0,1\}^N$ let $y_{D,i}$ denote the output of the client in the real world, and let $\mathbf{q}^i_{\mathcal{I}}$ denote the queries the corrupted server receives. By construction of the simulator and the fact that the client always outputs $D_i$ in the ideal world, we need to show that $(\mathbf{q}^i_{\mathcal{I}}, y_{D,i})$ and $(\mathbf{q}^1_{\mathcal{I}}, D_i)$ are statistically close. First, by the $t$-privacy of the protocol, $(\mathbf{q}^1_{\mathcal{I}}, D_i)$ and $(\mathbf{q}^i_{\mathcal{I}}, D_i)$ are statistically close. Second, observe that correctness implies that $(\mathbf{q}^i_{\mathcal{I}}, D_i)$ and $(\mathbf{q}^i_{\mathcal{I}}, y_{D,i})$ are statistically close. Thus, $(\mathbf{q}^i_{\mathcal{I}}, y_{D,i})$ and $(\mathbf{q}^1_{\mathcal{I}}, D_i)$ are statistically close.

### 1.3 Related Works

**1-private PIR.** We next state the best known PIR protocols. Efremenko [22] constructed a 3-server 1-private PIR protocol with query length $2^{\tilde{O}(\sqrt{\log N})}$ and $2^r$-server 1-private PIR proto-

cols with query length $2^{\tilde{O}(\sqrt[r]{\log N})}$; the answer length in these protocols is $O(1)$. Dvir and Gopi [20] constructed a PIR protocol with query and answer length $2^{\tilde{O}(\sqrt{\log N})}$. Very recently, Ghasemi, Kopparty, and Sudan [27] constructed a 3-server PIR protocol with query and answer length $2^{\tilde{O}(\sqrt[3]{\log N})}$. These constructions were simplified by Alon, Beimel, and Lasri [1]. The communication complexity of $\ell$-server PIR protocols for $\ell \geq 6$ was improved by Itoh and Suzuki [32, 33], Chee, Feng, Ling, Wang, and Zhang [14], Dvir and Gopi [20], and Ghasemi et al. [27]. For example, there is a 6-server PIR protocol with communication complexity $2^{\tilde{O}(\sqrt[4]{\log N})}$ [32, 3, 27, 1]. The best known lower bound on the total communication complexity of 2-server PIR protocols is $5 \log n$, proved by Wehner and de Wolf [45] (improving on [41, 36]).

**$t$-private PIR.** $t$-private PIR protocols were constructed in [16, 2, 30, 6, 8, 46]. In particular, Woodruff and Yekhanin [46] presented a $t$-private $\ell$-server PIR construction for general $\ell$ and $t$ with communication complexity $\frac{\ell^2}{t} \cdot \log \ell \cdot N^{1/\lfloor (2\ell-1)/t \rfloor}$. Barkol, Ishai, and Weinreb [5] presented a general transformation from 1-private PIR protocols to $t$-private PIR protocol; given a 1-private $\ell$-server PIR protocol with query length $m_q$ and answer length $m_a$, they constructed a $t$-private $\ell^t$-server PIR protocol with query length $O(tm_q)$ and answer length $O(m_a^t)$. When $t$ is small compared to $\ell$, this gives better protocols, e.g., a $t$-private $2^t$-server PIR protocol with communication complexity $2^{\tilde{O}(t\sqrt{\log(N)})}$ (using [20]) and a $t$-private $3^t$-server PIR protocol with query length $t \cdot 2^{\tilde{O}(\sqrt{\log(N)})}$ and answer length $2^{O(t)}$ (using [22]).

**Robust and Byzantine-robust PIR.** Beimel and Stahl [7] introduced robust and Byzantine robust PIR protocols. A PIR protocol is $t$-robust if the client can recover the correct value even if $t$ of the servers go offline. The generalized notion of $t$-Byzantine robust requires robustness to hold even if $t$ of the servers are malicious. This was further studied in subsequent works [46, 37, 24, 23]. In particular, Eriguchi et al. [23] showed that for a constant number of servers:

- The communication complexity of perfect $t$-Byzantine robust and $t$-private $\ell$-server PIR is equivalent to the communication complexity of $t$-private $(\ell - 2t)$-server PIR protocol;
- The communication complexity of statistical $t$-Byzantine robust and $t$-private $\ell$-server PIR is equivalent to $t$-private $(\ell - t)$-server PIR protocol.

Combining the above results and [46] yields a $t$-private and $t$-Byzantine robust $\ell$-server statistical PIR protocol with total communication $\omega(\log N) \cdot N^{\frac{1}{\lfloor (2\ell-1)/t \rfloor - 2}}$. When $t$ is relatively small compared to $\ell$, the results of [5, 20, 22] yield a $t$-private and a $t$-Byzantine robust $2^t$-server statistical PIR protocol with total communication complexity $2^{t^2 + \max\{\tilde{O}(t\sqrt{\log N}), t \log t\}}$ and a $t$-Byzantine robust $3^t$-server statistical PIR protocol with total communication complexity $3^{t^2+2t} \cdot \max\left\{t2^{\tilde{O}(\sqrt{\log N})}, t3^t\right\}$. Byzantine robust PIR protocols where the database entries are large were considered in [47, 43, 44, 4]. They measure efficiency compared to the size of the entries rather than the size of the database (as we do in our work).

Eriguchi et al. [24, 23] introduced error-detecting PIR, where the client may abort and it is required that it never outputs the wrong value. In [24], they showed that error-detecting with perfect security and communication that is sublinear in the database size is impossible. In particular, this shows that there is no non-trivial perfect security-with-abort PIR protocol. In [23] they constructed a compiler transforming any $t$-private $\ell$-server PIR protocol with communication $c_0$ to

a $t$-error-detecting one with total communication $\ell^2 \cdot \omega(\log N) \cdot (c_0 + \ell \log \ell)$. When $t$ is very small compared to $\ell$, this has significantly better communication complexity compared to the Byzantine robust protocols discussed above. For example, there is a $t$-error-detecting $2^t$-server PIR protocol with total communication complexity $2^{\max\left\{\tilde{O}(t\sqrt{\log N}), t \log t\right\}}$ and a $t$-error-detecting $3^t$-server PIR protocol with total communication complexity $3^{2t} \cdot \max\left\{t2^{\tilde{O}(\sqrt{\log N})}, t3^t\right\}$. Additional constructions of error-detecting PIR protocols appear in Eriguchi et al. [25].

Colombo et al. [17] also observed that standard PIR security definitions allow for selective abort attacks. This motivated them to define *authenticated PIR protocols*, where, similarly to error-detection, it is required that either the client aborts or its output is consistent with the honest server's database. Additionally, the privacy of the client's retrieval index is required to hold even if the adversary knows whether the client aborts or not, which prevents selective abort attacks. Unlike our definitions, Colombo et al. [17] defined security by listing the desired security properties.

**Computational PIR.** Computational PIR protocols consider computationally-bounded servers. This model was first considered by Chor and Gilboa [15]. In this setting, it was shown that single-server PIR protocols can be constructed with non-trivial communication complexity [38, 12, 26]. Computational PIR can be seen as a special case of fully homomorphic encryption; the result of Brakerski and Vaikuntanathan [11] used this to construct a single-server PIR protocol with total communication $\text{poly}(\kappa) + \log N$, where $\kappa$ is the security parameter.

**Spooky PIR.** Dwork et al. [21] studied two-round succinct arguments for NP by composing PCP proofs with computational single-server PIR protocols. They showed that such heuristics may be insecure. More generally, they argued that executing PIR protocols in parallel may introduce "spooky interactions": even though the queries made by the client are independent and the server knows nothing about the indexes of the client, the server may introduce correlations between the queries and outputs of the client. Dodis et al. [19] later showed that this holds even when using multi-prover interactive proofs instead of PCPs. These results show the necessity of defining the security of PIR protocols via the real vs. ideal world paradigm.

**Verifiable PIR.** Ben-David et al. [10] introduced verifiable PIR, where it is required from the (single) server to be able to prove that the database satisfies various properties. Although one of their definition follows the real vs. ideal world paradigm, it does not require privacy or security against selective abort attacks.

# 2 Preliminaries

## 2.1 Notations

We use bold characters to denote vectors. For $n \in \mathbb{N}$, let $[n] = \{1, 2 \ldots, n\}$. For a set $\mathcal{S}$, we write $s \leftarrow \mathcal{S}$ to indicate that $s$ is selected uniformly at random from $\mathcal{S}$. Given a random variable (or a distribution) $X$, we write $x \leftarrow X$ to indicate that $x$ is selected according to $X$. PPT stands for probabilistic polynomial time. A function $\varepsilon(\cdot)$ is called negligible if for every positive polynomial $p(\cdot)$ and all sufficiently large $n$, $\varepsilon(n) < 1/p(n)$. For a vector $\mathbf{v}$ of dimension $n$, we write $v_i$ for its $i^{\text{th}}$ coordinate, and for $\mathcal{S} \subseteq [n]$ we write $\mathbf{v}_{\mathcal{S}} = (v_i)_{i \in \mathcal{S}}$.

A *distribution ensemble* $X = \{X_{a,n}\}_{a\in\mathsf{D}_n,n\in\mathbb{N}}$ is an infinite sequence of random variables indexed by $a \in \mathsf{D}_n$ and $n \in \mathbb{N}$, where $\mathsf{D}_n$ is a domain that might depend on $n$. Computational indistinguishability is defined as follows.

**Definition 2.1.** *Let $X = \{X_{a,n}\}_{a\in\mathcal{D}_n,n\in\mathbb{N}}$ and $Y = \{Y_{a,n}\}_{a\in\mathcal{D}_n,n\in\mathbb{N}}$ be two ensembles. We say that $X$ and $Y$ are computationally indistinguishable, denoted $X \overset{\text{C}}{\equiv} Y$, if for every* PPT *distinguisher $\mathsf{D}$, there exists a negligible function $\varepsilon(\cdot)$, such that for all $n \in \mathbb{N}$ and $a \in \mathcal{D}_n$,*

$$|\Pr[\mathsf{D}(X_{a,n}) = 1] - \Pr[\mathsf{D}(Y_{a,n}) = 1]| \leq \varepsilon(n).$$

**Definition 2.2.** *The statistical distance between two finite random variables $X$ and $Y$ is defined as*

$$\mathrm{SD}(X, Y) = \max_{\mathcal{T}} (\Pr[X \in \mathcal{T}] - \Pr[Y \in \mathcal{T}]).$$

*Two ensembles $X = \{X_{a,n}\}_{a\in\mathcal{D}_n,n\in\mathbb{N}}$ and $Y = \{Y_{a,n}\}_{a\in\mathcal{D}_n,n\in\mathbb{N}}$ are said to be statistically close, denoted $X \overset{\text{S}}{\equiv} Y$, if there exists a negligible function $\varepsilon(\cdot)$, such that for all $n$ and $a \in \mathcal{D}_n$,*

$$\mathrm{SD}(X_{a,n}, Y_{a,n}) \leq \varepsilon(n).$$

*We say that $X$ and $Y$ are identically distributed, denoted $X \overset{\text{P}}{\equiv} Y$, if $\varepsilon(n) = 0$ for all $a \in \mathcal{D}_n$ and $n \in \mathbb{N}$.*

**Theorem 2.3** (Hoeffding's inequality for the hypergeometric distribution)**.** *Let $n \in \mathbb{N}$, $m, k \in [n]$, and $A \in \binom{[n]}{m}$. Then for every $t > 0$,*

$$\Pr_{X\leftarrow\binom{[n]}{k}}\left[|X \cap A| - \frac{km}{n} \geq t\right] \leq e^{-2t^2/k}$$

*and*

$$\Pr_{X\leftarrow\binom{[n]}{k}}\left[|X \cap A| - \frac{km}{n} \leq -t\right] \leq e^{-2t^2/k}.$$

## 2.2 Private Information Retrieval

In this section, we present the definition of private information retrieval (PIR) protocols [16]. In the next definition, we present the syntax of a *single-round PIR protocol*. We then define property-based security taken from previous papers [7, 24, 23]. To simplify the presentation, we present the definition for a constant number of servers (i.e., independent of the database size). The definition readily extends to a non-constant number of servers. Intuitively, the PIR protocol starts with the client running a query algorithm $\mathcal{Q}$ and sending the $j^{\text{th}}$ output to the $j^{\text{th}}$ server $S_j$. The $j^{\text{th}}$ server, holding the database $D$ and the query $q_j$, responds with the answer $\mathcal{A}(j, q_j, D)$. Finally, the client computes the output by applying the reconstruction algorithm $\mathcal{C}$. We next formalize this.

**Definition 2.4** (PIR protocols)**.** *Let $\ell \in \mathbb{N}$. An $\ell$-server PIR protocol is given by a 3-tuple $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ of algorithms with the following syntax.*

1. *The query algorithm $(q_1, q_2, \ldots, q_\ell, \mathsf{st}) \leftarrow \mathcal{Q}(1^N, i)$ is a randomized algorithm that is given the size of the database $N$ and a retrieval index $i \in [N]$. It outputs a query $q_j$ for every server $j \in [\ell]$ and a state $\mathsf{st} \in \{0,1\}^*$. We denote $\mathbf{q} = (q_1, \ldots, q_\ell)$.*

2. *The answer algorithm* $a_j = \mathcal{A}(j, q, D)$ *is a deterministic algorithm that is given an index* $j \in [\ell]$*, a query* $q$*, and a database* $D \in \{0,1\}^N$ *for some* $N \in \mathbb{N}$*. It outputs a response* $a_j$*. We denote* $\mathbf{a} = (a_1, \ldots, a_\ell)$*.*

3. *The reconstruction algorithm* $y = \mathcal{C}(1^N, \mathbf{a}, \mathsf{st})$ *is a deterministic algorithm that is given the size of the database* $N$*, the* $\ell$ *answers* $\mathbf{a} = (a_j)_{j\in[\ell]}$*, and the state* $\mathsf{st}$*. It outputs* $y \in \{0, 1, \perp\}$*.*

*The protocol is efficient if all three algorithms run in polynomial time in* $N$*. The total communication complexity is the maximum (taken over the randomness of the client and the databases* $D$*) of the total number of bits sent in the protocol, i.e.,* $\max_{r,D}\{\sum_{i=1}^{\ell} |q_i| + |a_i|\}$*, where* $r$ *is the randomness of* $\mathcal{Q}$*.*

### 2.3 Defining Security for PIR via Security Properties

We next define the security properties of PIR protocols as they appear in the literature. We first define correctness, which states that in an honest execution, the client outputs the correct value.

**Definition 2.5** (Correctness)**.** *Let* $\ell \in \mathbb{N}$ *and let* $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ *be an* $\ell$*-server PIR protocol. We say that* $\Pi$ *is statistically* correct *if for all sufficiently large* $N \in \mathbb{N}$*, any database* $D \in \{0,1\}^N$*, and any retrieval index* $i \in [N]$*,*

$$\Pr_{(\mathbf{q},\mathsf{st})\leftarrow\mathcal{Q}(1^N,i)}\left[\mathcal{C}\left(1^N, \left(\mathcal{A}\left(j, q_j, D\right)\right)_{j\in[\ell]}, \mathsf{st}\right) = D_i\right] \geq 1 - \varepsilon(N),$$

*for some negligible function* $\varepsilon(\cdot)$*, where the probability is over the randomness of* $\mathcal{Q}$*. If* $\varepsilon = 0$ *then we say that* $\Pi$ *is perfectly correct.*

We now define $t$-privacy, stating that no set of $t$ servers learns anything about the retrieval index of the client.

**Definition 2.6** (Privacy)**.** *Let* $\ell \in \mathbb{N}$*, let* $t \in [\ell]$*, and let* $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ *be an* $\ell$*-server PIR protocol. We say that* $\Pi$ *is* statistically $t$-private *if for any set of* $t$ *servers* $\mathcal{I} \in \binom{[\ell]}{t}$*, and any two sequences of indices* $\{i_N\}_{N\in\mathbb{N}}$ *and* $\{i'_N\}_{N\in\mathbb{N}}$*,*

$$\{\mathbf{q}_\mathcal{I}\}_{N\in\mathbb{N}, D\in\{0,1\}^N} \overset{\mathrm{S}}{\equiv} \{\mathbf{q}'_\mathcal{I}\}_{N\in\mathbb{N}, D\in\{0,1\}^N}$$

*where* $(\mathbf{q}, \mathsf{st}) \leftarrow \mathcal{Q}(1^N, i_N)$ *and* $(\mathbf{q}', \mathsf{st}') \leftarrow \mathcal{Q}(1^N, i'_M)$*. We define computationally/perfectly* $t$*-privacy by replacing* $\overset{\mathrm{S}}{\equiv}$ *with* $\overset{\mathrm{C}}{\equiv}$ *and* $\overset{\mathrm{P}}{\equiv}$*, respectively, in the above equation.*

We next define security properties that should hold against malicious behavior by a subset of the servers. Following [24, 23], we first define tampering algorithms;[3] this is an adversary that can modify some of the answers.

**Definition 2.7** (A tampering algorithm [24, 23])**.** *Let* $\ell \in \mathbb{N}$*,* $\mathcal{I} \subseteq [\ell]$*, and let* $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ *be an* $\ell$*-server PIR protocol. A tampering algorithm* $\mathcal{B}$ *for* $\Pi$ *is a randomized algorithm* $\mathbf{a}' = (a'_j)_{j\in[\ell]} \leftarrow \mathcal{B}(\mathcal{I}, \mathbf{q}, D)$ *that is given a set* $\mathcal{I} \subseteq [\ell]$*, a set of* $\ell$ *queries, and a database* $D \in \{0,1\}^N$ *for some* $N \in \mathbb{N}$*. It outputs answers* $(a'_j)_{j\in[\ell]}$ *such that* $a'_j = \mathcal{A}\left(j, q_j, D\right)$ *for all* $j \in [\ell] \setminus \mathcal{I}$*, and* $a'_j$ *depends on* $q_\mathcal{I}$ *and* $D$ *for all* $j \in \mathcal{I}$*.*

[3] Eriguchi et al. [24, 23] defined tampering functions rather than algorithms.

We now define $t$-error-detecting. Roughly, it means that no set of $t$ cheating servers can force the client to output the wrong value. Note, however, that the client is allowed to output $\perp$ (even with probability 1).

**Definition 2.8** (Error-detecting PIR [24, 23])**.** *Let $\ell \in \mathbb{N}$, $t \in [\ell]$, and let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be an $\ell$-server PIR protocol. We say that $\Pi$ is* statistical $t$-error-detecting *if for any $N \in \mathbb{N}$, for any database $D \in \{0,1\}^N$, for any retrieval index $i \in [N]$, any subset $\mathcal{I} \in \binom{[\ell]}{t}$ of $t$ servers, and any tampering algorithm $\mathcal{B}$ for $\Pi$,*

$$\Pr_{(\mathbf{q},\mathsf{st})\leftarrow\mathcal{Q}(1^N,i)}\left[\mathcal{C}\left(1^N, \mathcal{B}(\mathcal{I},\mathbf{q},D), \mathsf{st}\right) \in \{D_i, \perp\}\right] \geq 1 - \varepsilon(N),$$

*for some negligible function $\varepsilon(\cdot)$, where the probability is over the randomness of $\mathcal{Q}$ and $\mathcal{B}$. We define perfect $t$-error-detecting by requiring the above to hold for $\varepsilon(N) = 0$ for all $N \in \mathbb{N}$. We define computational $t$-error-detecting by requiring the above to hold only for* PPT *algorithms $\mathcal{B}$.*

Finally, we define $t$-Byzantine robustness, which roughly means that the client outputs the correct value (i.e., $D_i$) even if at most $t$ servers cheat. Note that any Byzantine robust PIR protocol is also correct, since we can consider a tampering algorithm that behaves honestly.

**Definition 2.9** (Byzantine robust PIR [7])**.** *Let $\ell \in \mathbb{N}$, let $t \in [\ell]$, and let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be an $\ell$-server PIR protocol. We say that $\Pi$ is* statistical $t$-Byzantine robust *if for any $N \in \mathbb{N}$, any database $D \in \{0,1\}^N$, any retrieval index $i \in [N]$, any subset $\mathcal{I} \in \binom{[\ell]}{t}$ of $t$ servers, and any tampering algorithm $\mathcal{B}$ for $\Pi$,*

$$\Pr_{(\mathbf{q},\mathsf{st})\leftarrow\mathcal{Q}(1^N,i)}\left[\mathcal{C}\left(1^N, \mathcal{B}(\mathcal{I},\mathbf{q},D), \mathsf{st}\right) = D_i\right] \geq 1 - \varepsilon(N),$$

*for some negligible function $\varepsilon(\cdot)$, where the probability is over the randomness of $\mathcal{Q}$ and $\mathcal{B}$. We define perfect $t$-Byzantine robust by requiring the above to hold for $\varepsilon(N) = 0$ for all $N \in \mathbb{N}$. We define computational $t$-Byzantine robust by requiring the above to hold only for* PPT *algorithms $\mathcal{B}$.*

# 3 Defining Security via The Real vs. Ideal World Paradigm

In this section, we present the security definition for PIR via the real vs. ideal world paradigm. We present the definition for both full security (i.e., with guaranteed output delivery) and for security-with-abort. We first define the real world execution, followed by the ideal worlds for full security and security-with-abort. Then, we present the security definitions. Our main contribution in the definition is defining the appropriate functionalities that are computed in the ideal world.

### The real world

Let $\ell \in \mathbb{N}$ and let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be an $\ell$-server PIR protocol. Toward defining security, we first describe an execution of $\Pi$ in the presence of an adversary $\mathcal{B}$. The adversary controls a subset $\mathcal{I} \subseteq [\ell]$ of the servers. It receives the database $D \in \{0,1\}^N$ and the queries the corrupted servers receive from the client, and instructs each server how to respond. We consider *malicious* adversaries that can instruct the corrupted servers to respond to the client in an arbitrary way. At the end of the execution, the adversary outputs some function of its view (i.e., its randomness, the database, and the queries it received from the client). For a database size $N \in \mathbb{N}$, a database $D \in \{0,1\}^N$,

and a retrieval index $i \in [N]$, we let $\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N, D, i)$ and $\text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N, D, i)$ denote the outputs of $\mathcal{B}$ and the client respectively in an execution of $\Pi$. Finally, let

$$\text{REAL}_{\Pi,\mathcal{B}}(N, D, i) = \left(\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N, D, i),\ \text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N, D, i)\right).$$

**The ideal world – full security**

We next describe an ideal computation with *guaranteed output delivery* (also referred to as full security) for PIR, where a trusted party $\mathsf{T}$ performs the computation on behalf of the parties, and the ideal model adversary cannot abort the computation. Note that we only consider adversaries that *do not* corrupt the client and corrupt a minority of the servers. The input of the adversary is the input of the servers it corrupts, i.e., the database $D$. An ideal computation with database size $N \in \mathbb{N}$, on input a database $D \in \{0,1\}^N$ for the servers and retrieval index $i \in [N]$ for the client, in the presence of an adversary (a simulator) $\mathsf{Sim}$ controlling a set $\mathcal{I} \subseteq [\ell]$, proceeds as follows.

**Parties send inputs to the trusted party:** Each honest server $S_j$, i.e., $j \notin \mathcal{I}$, sends $D^j := D$ to the trusted party $\mathsf{T}$. For each corrupted server $S_j$, i.e., $j \in \mathcal{I}$, the adversary $\mathsf{Sim}$ sends to $\mathsf{T}$ a database $D^j \in \{0,1\}^N$ of its choice. If the server does not send any database then $\mathsf{T}$ sets $D^j = 0^N$. The client sends $i$ to $\mathsf{T}$.

**The trusted party performs the computation:** If more than $1/2$ of the databases equal some $D'$, then the trusted party sends $D'_i$ to the client. Otherwise, send $D^1_i$ to the client.

**Output:** The client outputs whatever it received from $\mathsf{T}$ and the adversary outputs some function of its view (i.e., its random string and the database $D$).

For a database size $N \in \mathbb{N}$, a database $D \in \{0,1\}^N$, and a retrieval index $i \in [N]$, let $\text{VIEW}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i)$ and $\text{OUT}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i)$ denote the outputs of $\mathsf{Sim}$ and the client respectively in an execution of the above ideal world. Finally, let

$$\text{IDEAL}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i) = \left(\text{VIEW}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i),\ \text{OUT}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i)\right).$$

Note that the client always outputs $D_i$ if there is an honest majority.

**The ideal world – security-with-abort**

We next describe an ideal computation with *security-with-abort* for PIR. Unlike the ideal model for full security, here the adversary can abort the computation. An ideal computation with database size $N \in \mathbb{N}$, on input a database $D \in \{0,1\}^N$ and index $i \in [N]$ for the client, in the presence of an adversary (a simulator) $\mathsf{Sim}$ controlling a set $\mathcal{I} \subseteq [\ell]$, proceeds as follows.

**Parties send inputs to the trusted party:** Each honest server $S_j$ sends $D^j := D$ to the trusted party $\mathsf{T}$. For each corrupted server $S_j$, the adversary $\mathsf{Sim}$ sends to $\mathsf{T}$ a database $D^j \in \{0,1\}^N$ of its choice. If the server does not send any database then $\mathsf{T}$ sets $D^j = 0^N$. The client sends $i$ to $\mathsf{T}$.

**The trusted party performs the computation:** If the databases that $\mathsf{T}$ received are not identical, then $\mathsf{T}$ sends $\perp$ to the client. Otherwise, it sends it $D^1_i$.

**Output:** The client outputs whatever it received from $\mathsf{T}$ and the adversary outputs some function of its view (i.e., its random string and the database $D$).

For a database size $N \in \mathbb{N}$, a database $D \in \{0,1\}^N$, and a retrieval index $i \in [N]$, let $\text{VIEW}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i)$ and $\text{OUT}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i)$ denote the outputs of $\mathsf{Sim}$ and the client respectively in an execution of the above ideal world. Finally, let

$$\text{IDEAL}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i) = \left(\text{VIEW}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i),\ \text{OUT}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i)\right).$$

### Defining Security

Next, we define the security of a PIR protocol via the real vs. ideal world paradigm. We define full security and security-with-abort.

**Definition 3.1** (Security via real vs. ideal paradigm)**.** *Let $\ell \in \mathbb{N}$, $t \in [\ell]$, and let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be an $\ell$-server PIR protocol. We say that $\Pi$ is* statistically $t$-fully secure *if for every adversary $\mathcal{B}$ controlling a set $\mathcal{I} \subseteq [\ell]$ of at most $t$ servers, there exists a simulator $\mathsf{Sim}$ controlling the same subset $\mathcal{I}$ in the ideal world, such that*

$$\left\{\text{IDEAL}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i)\right\}_{N \in \mathbb{N}, D \in \{0,1\}^N, i \in [N]} \overset{\text{S}}{\equiv} \left\{\text{REAL}_{\Pi, \mathcal{B}}(N, D, i)\right\}_{N \in \mathbb{N}, D \in \{0,1\}^N, i \in [N]}.$$

*We say that $\Pi$ is* statistically $t$-secure-with-abort *if the above holds with $\text{IDEAL}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i)$ replacing $\text{IDEAL}^{\mathsf{god}}_{\mathsf{Sim}}(N, D, i)$.*

*Perfect $t$-security is defined by replacing $\overset{\text{S}}{\equiv}$ with $\overset{\text{P}}{\equiv}$ in the above equations. Computational $t$-security is defined by replacing $\overset{\text{S}}{\equiv}$ with $\overset{\text{C}}{\equiv}$ and limiting both the real-world adversary $\mathcal{B}$ and the ideal-world simulator $\mathsf{Sim}$ to be* PPT *algorithms.*

**Remark 3.2.** *Standard security definitions following the real vs. ideal world paradigm consider adversaries that receive auxiliary information. The same auxiliary information is given to the real-world adversary and its simulator. This is necessary to argue that the composition of secure protocols remains secure. If we were to add this to the security definition, then in order to argue that full security is equivalent to the privacy and Byzantine robustness properties (see Theorem 4.1 below), we would have had to add the auxiliary information to the privacy requirement (the statement and proof of equivalence remain the same). We decided not to include the auxiliary information in the privacy definition since this is not a common definition in the literature.*

## 4 Comparing the Definitions

In this section, we compare the definitions given in the previous section. We start by showing the equivalence between full security and the property-based security definition. Specifically, we show that a PIR protocol is fully secure if and only if it is both private and Byzantine robust (recall that Byzantine robustness implies correctness). The interesting direction is showing the property-based security definition implies the real vs. ideal definition.

**Theorem 4.1.** *Let $\ell, t \in \mathbb{N}$ be such that $t < \ell/2$, let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be an $\ell$-server PIR protocol, and let $\mathsf{type} \in \{$computationally, statistically, perfectly$\}$. Then, $\Pi$ is $\mathsf{type}$ $t$-fully secure if and only if it is $\mathsf{type}$ $t$-private and $\mathsf{type}$ $t$-Byzantine robust.*

*Proof.* We prove the results for $\mathsf{type}$ = statistically. The other cases are similar. We first prove that if $\Pi$ is both statistically $t$-private and statistically $t$-Byzantine robust, then it is statistically $t$-fully secure. Fix a real-world adversary $\mathcal{B}$ corrupting a set $\mathcal{I}$ of at most $t$ servers. We define its simulator $\mathsf{Sim}$ as follows. First, it sends to the trusted party the database $D$ for every corrupted party. It then computes $(\mathbf{q}^1, \mathsf{st}^1) \leftarrow \mathcal{Q}(1^N, 1)$ (i.e., queries for the index 1), sends $\mathbf{q}^1_{\mathcal{I}}$ to the adversary, and outputs whatever it outputs. Note that if $\mathcal{B}$ and $\mathcal{Q}$ are PPT algorithms, then $\mathsf{Sim}$ is a PPT algorithm.

We now show that the statistical difference between the real and ideal worlds is negligible. For a retrieval index $i \in [N]$ and a database $D \in \{0,1\}^N$ let $y_{D,i}$ denote the output of the client in the real world. Note that it suffices to show that

$$\left\{\left(\mathbf{q}^i_{\mathcal{I}}, y_{D,i}\right)\right\}_{N\in\mathbb{N}, D\in\{0,1\}^N, i\in[N]} \stackrel{\text{S}}{\equiv} \left\{\left(\mathbf{q}^1_{\mathcal{I}}, D_i\right)\right\}_{N\in\mathbb{N}, D\in\{0,1\}^N, i\in[N]}, \tag{1}$$

where $(\mathbf{q}^i, \mathsf{st}^i) \leftarrow \mathcal{Q}(1^N, i)$.

Fix an event $\mathcal{T}$ and let

$$\Delta = \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, y_{D,i}\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^1_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right],$$

where the probabilities are over the execution of the real/ideal world. Then

$$\begin{aligned}\Delta &= \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, y_{D,i}\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^1_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] \\ &= \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, y_{D,i}\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^1_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right].\end{aligned}$$

Observe that by the $t$-privacy of $\Pi$,

$$\Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^1_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] = \mathrm{neg}(N).$$

Indeed, if this was not the case, then

$$\Pr\left[\mathbf{q}^i_{\mathcal{I}} \in \mathcal{T}_{D_i}\right] - \Pr\left[\mathbf{q}^1_{\mathcal{I}} \in \mathcal{T}_{D_i}\right]$$

is non-negligible, where $\mathcal{T}_{D_i} = \{\mathbf{q} : (\mathbf{q}, D_i) \in \mathcal{T}\}$. Therefore,

$$\begin{aligned}\Delta &\le \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, y_{D,i}\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \mathrm{neg}(N) \\ &= \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] \cdot \Pr\left[y_{D,i} = D_i\right] + \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, 1-D_i\right) \in \mathcal{T}\right] \cdot \Pr\left[y_{D,i} = 1 - D_i\right] \\ &\quad - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \mathrm{neg}(N).\end{aligned}$$

By the $t$-Byzantine robustness of $\Pi$,

$$\Pr\left[y_{D,i} = 1 - D_i\right] = \mathrm{neg}(N).$$

Thus,

$$\begin{aligned}\Delta &\le \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] \cdot \Pr\left[y_{D,i} = D_i\right] + \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, 1-D_i\right) \in \mathcal{T}\right] \cdot \Pr\left[y_{D,i} = 1 - D_i\right] \\ &\quad - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \mathrm{neg}(N) \\ &= \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] \cdot (1 - \mathrm{neg}(N)) + \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, 1-D_i\right) \in \mathcal{T}\right] \cdot \mathrm{neg}(N) \\ &\quad - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \mathrm{neg}(N) \\ &= \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] - \Pr\left[\left(\mathbf{q}^i_{\mathcal{I}}, D_i\right) \in \mathcal{T}\right] + \mathrm{neg}(N) \\ &= \mathrm{neg}(N).\end{aligned}$$

We now prove the second direction. Assume that $\Pi$ is statistically $t$-fully secure. We first prove that $\Pi$ is statistically $t$-private. Fix a set $\mathcal{I}$ of size $t$ and consider the real-world adversary $\mathcal{B}$ controlling the servers in $\mathcal{I}$ that outputs the queries it received from the client. Then by assumption, there exists an ideal-world simulator $\mathsf{Sim}$ such that

$$\left\{\text{IDEAL}^{\mathsf{god}}_{\mathsf{Sim}}(N,D,i)\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i\in[N]} \stackrel{\text{S}}{\equiv} \left\{\text{REAL}_{\Pi,\mathcal{B}}(N,D,i)\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i\in[N]}. \tag{2}$$

In particular,

$$\left\{\text{VIEW}^{\mathsf{god}}_{\mathsf{Sim}}(N,D,i)\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i\in[N]} \stackrel{\text{S}}{\equiv} \left\{\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i\in[N]}.$$

Now, as $\mathsf{Sim}$ does not receive any message from the trusted party, it follows that its output is independent of $i$. Thus,

$$\text{VIEW}^{\mathsf{god}}_{\mathsf{Sim}}(D,i) \equiv \text{VIEW}^{\mathsf{god}}_{\mathsf{Sim}}(D,i')$$

for all $D\in\{0,1\}^N$ and $i,i'\in[N]$. Therefore,

$$\left\{\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i,i'\in[N]} \stackrel{\text{C}}{\equiv} \left\{\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i')\right\}_{N\in\mathbb{N},D\in\{0,1\}^N,i,i'\in[N]}.$$

Privacy follows from the fact that $\text{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i) = \mathbf{q}_{\mathcal{I}}$, where $(\mathbf{q},\mathsf{st}) \leftarrow \mathcal{Q}(1^N,i)$.

We now show that $\Pi$ is statistically $t$-Byzantine robust. Fix a tampering algorithm $\mathcal{B}'$ and consider the adversary $\mathcal{B}$ that instructs the corrupted server to respond by computing $\mathcal{B}'$. Then by Equation (2),

$$\left|\Pr\left[\text{OUT}^{\mathsf{god}}_{\mathsf{Sim}}(N,D,i)=D_i\right] - \Pr\left[\text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)=D_i\right]\right| \le \varepsilon(N),$$

for some negligible function $\varepsilon(\cdot)$. Since $t<\ell/2$, in the ideal world, the client always outputs $D_i$. Thus, $\Pr\left[\text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)=D_i\right] \ge 1-\varepsilon(N)$. □

**Remark 4.2.** *Note that the result holds for PIR protocols with more than a single round. To see this, let the simulator run the client on input 1 and simulate the entire execution of the protocol to generate the adversary's view. The same analysis shows that privacy and Byzantine robustness suffice to argue that the simulator succeeds.*

For security-with-abort, we show that the real vs. ideal definition is *strictly stronger* than the property-based security. The next theorem states that security-with-abort implies the corresponding property-based security.

**Theorem 4.3.** *Let $\ell,t\in\mathbb{N}$, let $\Pi=(\mathcal{Q},\mathcal{A},\mathcal{C})$ be an $\ell$-server PIR protocol, and let* $\mathsf{type}\in\{$*computationally, statistically, perfectly*$\}$*. If $\Pi$ is* $\mathsf{type}$ *$t$-secure-with-abort, then it is* $\mathsf{type}$ *correct,* $\mathsf{type}$ *$t$-private, and* $\mathsf{type}$ *$t$-error-detecting.*

*Proof.* We assume that $\mathsf{type}$ = statistically since the other cases are handled similarly. For correctness, consider an adversary that does not corrupt any server. Then the output of the client in the ideal world is $D_i$. Therefore, in the real world, the client outputs $D_i$ except with negligible probability. The proof that $\Pi$ is statistically $t$-private is identical to the proof done in Theorem 4.1

and is therefore omitted. We now show that $\Pi$ is statistically $t$-error-detecting. Fix a tampering algorithm $\mathcal{B}'$ and consider the adversary $\mathcal{B}$ that instructs the corrupted server to respond by computing $\mathcal{B}'$. Then there is a simulator $\mathsf{Sim}$ such that

$$\left|\Pr\left[\text{OUT}^{\mathsf{swa}}_{\mathsf{Sim}}(N,D,i)=y\right]-\Pr\left[\text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)=y\right]\right|\leq\varepsilon(N),$$

for all $y\in\{0,1,\perp\}$, where $\varepsilon(\cdot)$ is a negligible function. Since in the ideal world the client never outputs $1-D_i$, it follows that $\Pr\left[\text{OUT}^{\mathsf{real}}_{\Pi,\mathcal{B}}(N,D,i)\in\{D_i,\perp\}\right]\geq 1-\varepsilon(N)$. □

We next show that the converse does not hold: there exists an $\ell$-server PIR protocol $\Pi$ that is correct, 1-private, and 1-error-detecting but is not computationally 1-secure-with-abort. Intuitively, we construct a protocol where a malicious server can force a "selective abort" of the client, that is, the adversary forces the client to abort if and only if it holds certain indices $i$ (e.g., $i=1$). This is impossible to simulate since the simulation is independent of $i$. We next formalize this intuition.

**Example 4.4.** *Let $\Pi$ be a perfectly 1-secure-with-abort PIR protocol (e.g., each server sends $D$ to the client; if all databases are the same then the client outputs $D_i$, else it aborts). Consider the following PIR protocol $\Pi'$. The client computes the same queries as in $\Pi$, all servers respond with the same messages, and additionally, the first server sends an additional bit $b\in\{0,1\}$. An honest server sends $b=0$. If the client has input $i=1$ and it received $b=1$ then it outputs $\perp$. Otherwise, it proceeds as in $\Pi$. Clearly, $\Pi'$ is correct, 1-private, and 1-error-detecting.*

*We next show that $\Pi'$ is* not *1-secure-with-abort. Intuitively, this is because the adversary can force a probability for abort on $i=1$ that is different from the probability on $i\neq 1$ (although each server does not know $i$). Fix an adversary $\mathcal{B}$ that corrupts $S_1$ and sends $b=1$, and assume towards contradiction that there exists a simulator $\mathsf{Sim}$ for it. Consider an execution of $\Pi'$ with $i\leftarrow\{1,2\}$ (i.e., $i=1$ with probability 1/2 and $i=2$ with probability 1/2) and $D=1^N$. Then in the real world,*

$$\Pr_{i\leftarrow\{1,2\}}\left[i=1,\text{OUT}^{\mathsf{real}}_{\Pi',\mathcal{B}}(N,1^N,i)=\perp\right]=\Pr_{i\leftarrow\{1,2\}}\left[i=2,\text{OUT}^{\mathsf{real}}_{\Pi',\mathcal{B}}(N,1^N,i)=1\right]=\frac{1}{2}.$$

*Let us analyze the ideal world. Let $p$ denote the probability that $\mathsf{Sim}$ sends $D=1^N$ to the trusted party (and thus the client outputs 1). Since in the ideal world the simulator is independent of $i$,*

$$\Pr_{i\leftarrow\{1,2\}}\left[i=1,\text{OUT}^{\mathsf{swa}}_{\mathsf{Sim}}(N,1^N,i)=\perp\right]=\frac{1}{2}\cdot(1-p)$$

*and*

$$\Pr_{i\leftarrow\{1,2\}}\left[i=2,\text{OUT}^{\mathsf{swa}}_{\mathsf{Sim}}(N,1^N,i)=1\right]=\frac{1}{2}\cdot p.$$

*Clearly, it cannot be the case where both values are 1/2 as in the real world, thus the two worlds can be easily distinguished.*

One might expect that the reason the above example worked is because the client aborts with different probabilities for different inputs. We next show that adding such a requirement is still insufficient to argue for simulation-based security. Roughly, this is done by letting the client send to the first server a random index $i'$ that will inform the server whether the client will abort on $i'$ (if the additional bit $b$ from the previous example is 1). Thus, in the real world, the adversary can choose for which inputs the client will abort, which cannot be simulated in the ideal world. We next formalize this.

**Example 4.5.** *Let $\Pi$ be a perfectly 1-secure-with-abort PIR protocol (e.g., the servers send $D$ to the client who then proceeds like the trusted party). Consider the following PIR protocol $\Pi'$. The client computes the same queries as in $\Pi$ and appends the first query a random $i' \leftarrow [N]$ sampled uniformly at random. All servers respond with the same messages, and additionally, the first server sends an additional bit $b \in \{0,1\}$. An honest server sends $b = 0$. If the client has input $i = i'$ and it received $b = 1$, then it outputs $\perp$. Otherwise, it proceeds as in $\Pi$. Clearly, $\Pi'$ is correct, 1-private, 1-error-detecting, and the probability the client aborts is $1/N$ if the first server cheats (and 0 otherwise). The formal argument that $\Pi'$ is not secure-with-abort is similar to the previous example and is therefore omitted.*

# 5 A Generic Transformation From Private PIR to Secure-With-Abort PIR

In this section, we show a generic transformation of a private PIR protocol to a secure-with-abort protocol, based on Eriguchi et al. [23]. The original construction of [23] is *insecure* according to Definition 3.1 (see Remark 5.7 below). However, we prove that a modification of the construction is secure. We prove the following.

**Theorem 5.1.** *Let $\ell \in \mathbb{N}$, $t \in [\ell]$, and $\mathsf{type} \in \{$statistically, computationally$\}$. There exists a compiler that transforms any statistically correct and $\mathsf{type}$ $t$-private $\ell$-server PIR protocol $\Pi$ to a $\mathsf{type}$ $t$-secure-with-abort protocol. Moreover, if the total communication complexity of $\Pi$ is $c$, then the new protocol has total communication complexity $\ell^2 \cdot \omega(\log N) \cdot (c + \ell \log \ell)$.*

*Proof.* We prove the result for $\mathsf{type} =$ statistically. The case where $\mathsf{type} =$ computationally follows from the fact that the simulator we construct is efficient (if the adversary is) and from simple hybrid arguments. Let $\Pi = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$ be a statistically correct and statistically $t$-private PIR protocol. The idea of the transformation is the following. First, the client generates the queries $\mathbf{q}$ as in $\Pi$. Then, with probability $1/2$, the client samples two indices $m_1, m_2 \in [\ell]$ at random such that $m_1 \neq m_2$, and sends $q_j$ to the $j^{\text{th}}$ server $S_j$ for every $j \neq m_1$, and sends $(m_2, q_{m_2})$ to server $S_{m_1}$ (i.e., query $q_{m_2}$ is a duplicate query). With probability $1/2$, the client sends $q_j$ to $S_j$ for every $j \in [\ell]$ (without duplicating any of them). We refer to the first kind of execution, where the client duplicated a query, as a *test execution*, and we refer to the second kind of execution as a *real execution*. The $j^{\text{th}}$ server, upon receiving $q_j$, responds as $S_j$ does in $\Pi$, and upon receiving $(m_2, q_{m_2})$, $S_{m_1}$ responds as $S_{m_2}$ does in $\Pi$.

Notice that if in a test execution, $S_{m_1}$ is honest, then the view of the adversary in this case is identical to its view in a real execution. Thus, if the adversary cheats in the test execution, then it cheats in the real execution.

We then let the parties repeat this process in parallel sufficiently many times. The output of the client is defined as follows. If in one of the test executions, the client receives from the servers $S_{m_1}$ and $S_{m_2}$ different answers, then one of them is malicious and so the client aborts. Otherwise, the client computes the output of all of the real executions and outputs the majority. Intuitively, if we repeat this sufficiently many times, then the adversary cannot distinguish the real executions from most of the test executions. Therefore, if it cheats on too many real executions, then the client will catch it in one of the test executions. Otherwise, it will cheat on a minority of the real executions; hence, the correctness of the underlying PIR protocol implies that the client will compute the correct value most of the time.

For convenience, instead of deciding at random and independently which execution is a test and which is a real execution, we let the client randomly sample exactly half of the executions to be real.[4] We next formalize this.

**Protocol 5.2** ($\Pi_{\mathsf{swa}} = (\mathcal{Q}_{\mathsf{swa}}, \mathcal{A}_{\mathsf{swa}}, \mathcal{C}_{\mathsf{swa}})$).
**Inputs:** *The client holds the database length* $1^N$ *and a retrieval index* $i \in [N]$. *Each server holds a database* $D \in \{0,1\}^N$.
*Let* $\lambda \in \mathbb{N}$ *be an integer such that* $\lambda/2$ *is odd to be determined by the analysis below.*

**The query algorithm:** *The client samples a set* $\mathcal{K} \in \binom{[\lambda]}{\lambda/2}$ *and computes* $(\mathbf{q}^1, \mathsf{st}^1), \ldots, (\mathbf{q}^\lambda, \mathsf{st}^\lambda) \leftarrow \mathcal{Q}(1^N, i)$ *independently. It then does the following for all* $k \in [\lambda]$.

1. *Sample a pair* $m_1^k, m_2^k \in [\ell]$ *of distinct elements independently and uniformly at random.*
2. *If* $k \in \mathcal{K}$ *then set* $\hat{q}_j^k = q_j^k$ *for all* $j \in [\ell]$. *Otherwise, for every* $j \in [\ell] \setminus \{m_1^k\}$ *set* $\hat{q}_j^k = q_j^k$, *and set* $\hat{q}_{m_1^k}^k = (m_2^k, q_{m_2^k}^k)$.

*Let* $\mathsf{st} = \mathcal{K} || ((m_1^k, m_2^k, \mathsf{st}^k))_{k \in [\lambda]}$. *The client sends* $(\hat{q}_j^1, \ldots, \hat{q}_j^\lambda)$ *to the* $j^{th}$ *server.*

**The answer algorithm:** *For every* $k \in [\lambda]$ *and* $j \in [\ell]$, *if server* $S_j$ *receives* $\hat{q}_j^k = q_j^k$ *from the client, it computes* $a_j^k = \mathcal{A}(j, \hat{q}_j^k, D)$. *Else, it receives* $\hat{q}_j^k = (m_2^k, q_{m_2^k}^k)$ *and computes* $a_j^k = \mathcal{A}(m_2^k, q_{m_2^k}^k, D)$. *It sends* $(a_j^1, \ldots, a_j^\lambda)$ *to the client.*

**The reconstruction algorithm:** *The client computes* $y^k = \mathcal{C}(1^N, a_1^k, \ldots, a_\ell^k, \mathsf{st}^k)$ *for all* $k \in \mathcal{K}$. *Output* $\perp$ *if there exists* $k \in [\lambda] \setminus \mathcal{K}$ *such that and* $a_{m_1^k}^k \neq a_{m_2^k}^k$. *Otherwise, the client outputs* $\mathsf{maj}_{k \in \mathcal{K}}\{y^k\}$.

We now show the security of $\Pi_{\mathsf{swa}}$. Fix an adversary $\mathcal{B}$ corrupting a set $\mathcal{I} \subseteq [\ell]$ of at most $t$ servers. We define its simulator $\mathsf{Sim}$ whose input is $D$ as follows. Sample $(\hat{\mathbf{q}}, \mathsf{st}) \leftarrow \mathcal{Q}_{\mathsf{swa}}(1^N, 1)$ and send $\hat{\mathbf{q}}_{\mathcal{I}}$ to $\mathcal{B}$. For every $j \in \mathcal{I}$ let $a_j$ denote the response $\mathcal{B}$ made for server $S_j$, and for every $j \in [\ell] \setminus \mathcal{I}$ let $a_j = \mathcal{A}_{\mathsf{swa}}(j, \hat{q}_j, D)$. Write $\mathsf{st} = \mathcal{K} || ((m_1^k, m_2^k, \mathsf{st}^k))_{k \in [\lambda]}$. If there exists $k \in [\lambda] \setminus \mathcal{K}$ such that $a_{m_1^k}^k \neq a_{m_2^k}^k$, then send to the trusted party a different database for the corrupted servers. Otherwise, send $D$ as the input of every corrupted server. Finally, output whatever $\mathcal{B}$ outputs and halt. We show that

$$\{\text{IDEAL}^{\mathsf{swa}}_{\mathsf{Sim}}(N, D, i)\}_{N \in \mathbb{N}, D \in \{0,1\}^N, i \in [N]} \stackrel{\text{S}}{\equiv} \{\text{REAL}_{\Pi_{\mathsf{swa}}, \mathcal{B}}(N, D, i)\}_{N \in \mathbb{N}, D \in \{0,1\}^N, i \in [N]}. \tag{3}$$

To prove this, we first prove the following lemma, stating that $\mathcal{B}$ cannot force the client to output $1 - D_i$ with noticeable probability.

**Lemma 5.3.** *Let* noAbort *be the event that* $a_{m_1^k}^k = a_{m_2^k}^k$ *for every* $k \in [\lambda] \setminus \mathcal{K}$ *(i.e., the client did not output* $\perp$*), and let* Fail *denote the event that* $\mathsf{maj}_{k \in \mathcal{K}}\{y^k\} = 1 - D_i$ *(i.e., the client output the wrong value). Then*

$$\Pr[\mathsf{noAbort} \wedge \mathsf{Fail}] \leq 2e^{-\lambda/64} + e^{-\frac{\lambda}{8\ell(\ell-1)}} + \text{neg}(N), \tag{4}$$

[4] Note that if $\Pi$ is perfectly correct, our protocol achieves only statistical correctness since it could be the case where the client duplicates a query in every execution of $\Pi$.

*where the probability is over the random coins of the client and the adversary.*

*Proof.* We prove that the bound holds for any fixed randomness of the adversary. In the following, we assume the randomness of $\mathcal{B}$ and the queries of the client before duplication (i.e., $\mathbf{q}^1, \ldots, \mathbf{q}^\lambda$) are fixed. Observe that for every $k \in [\lambda]$, the adversary cannot distinguish the case where $k \in \mathcal{K}$ and the case where $k \notin \mathcal{K}$ and $m_1^k \notin \mathcal{I}$. Therefore, if it instructs a server to cheat in one case, then it will instruct it to cheat in the other case.

To this end, we say that a server $S_j$ *cheated in execution* $k$ if its answer is different from $\mathcal{A}(j, q_j^k, D)$. That is, if it were to receive its original query before duplicating, then it would send a different answer. Let

$$A = \{k \in [\lambda] : \exists j \in \mathcal{I} \text{ s.t. } S_j \text{ cheated in execution } k\},$$

i.e., $A$ denotes the set of all executions where the adversary cheated. As we fixed the randomness of the adversary and $\mathbf{q}^1, \ldots, \mathbf{q}^\lambda$, the set $A$ is properly defined. We show that if $|A|$ is too large, then the client will catch the adversary with overwhelming probability in one of the test executions, and if it is too small, then it is unlikely that the majority of the outputs computed for the real execution will be incorrect. We separate the proof into two cases.

We first analyze the probability in (4) assuming $|A| < 3\lambda/8$. We show that it is unlikely that the client outputs the wrong value. That is, we show $\mathsf{Fail}$ occurs with low probability. Observe that the expectation of $|A \cap \mathcal{K}|$ is at most $|A|/2 < 3\lambda/16$. By Hoeffding's inequality (Theorem 2.3), it follows that

$$\Pr_{\mathcal{K} \leftarrow \binom{[\lambda]}{\lambda/2}}\left[|A \cap \mathcal{K}| \geq \frac{\lambda}{4}\right] \leq \Pr_{\mathcal{K} \leftarrow \binom{[\lambda]}{\lambda/2}}\left[|A \cap \mathcal{K}| - \frac{3\lambda}{16} \geq \frac{\lambda}{16}\right] \leq e^{-\lambda/64}.$$

Let $k \in \mathcal{K} \setminus A$, i.e., the $k^{\text{th}}$ execution is an honest real execution. By the statistical correctness of the original PIR protocol, the probability that the client outputs $1 - D_i$ in the $k^{\text{th}}$ execution of the protocol is negligible. Therefore,

$$\begin{aligned}
\Pr[\mathsf{Fail}] &= \Pr\left[\mathsf{maj}_{k \in \mathcal{K}}\left\{y^k\right\} = 1 - D_i\right] \\
&= \Pr\left[|A \cap \mathcal{K}| \geq \frac{\lambda}{4}\right] \cdot \Pr\left[\mathsf{maj}_{k \in \mathcal{K}}\left\{y^k\right\} = 1 - D_i \middle| |A \cap \mathcal{K}| \geq \frac{\lambda}{4}\right] \\
&\quad + \Pr\left[|A \cap \mathcal{K}| < \frac{\lambda}{4}\right] \cdot \Pr\left[\mathsf{maj}_{k \in \mathcal{K}}\left\{y^k\right\} = 1 - D_i \middle| |A \cap \mathcal{K}| < \frac{\lambda}{4}\right] \\
&\leq \Pr\left[|A \cap \mathcal{K}| \geq \frac{\lambda}{4}\right] + \Pr\left[\mathsf{maj}_{k \in \mathcal{K}}\left\{y^k\right\} = 1 - D_i \middle| |A \cap \mathcal{K}| < \frac{\lambda}{4}\right] \\
&\leq e^{-\lambda/64} + \Pr\left[\exists_{k \in \mathcal{K} \setminus A}\, y^k = 1 - D_i \middle| |A \cap \mathcal{K}| < \frac{\lambda}{4}\right] \\
&\leq e^{-\lambda 64} + \mathrm{neg}(N).
\end{aligned}$$

We now analyze the probability in (4) assuming $|A| \geq 3\lambda/8$. We show that in this case, the client will catch the adversary cheating with high probability. Since the expectation of $|A \cap \overline{\mathcal{K}}|$ is at least $|A|/2 \geq 3\lambda/16$, by Hoeffding's inequality, it follows that

$$\Pr_{\mathcal{K} \leftarrow \binom{[\lambda]}{\lambda/2}}\left[|A \cap \overline{\mathcal{K}}| \leq \frac{\lambda}{8}\right] \leq \Pr_{\mathcal{K} \leftarrow \binom{[\lambda]}{\lambda/2}}\left[|A \cap \overline{\mathcal{K}}| - \frac{3\lambda}{16} \leq -\frac{\lambda}{16}\right] \leq e^{-\lambda/64}.$$

Next, for every $k \in A \cap \overline{\mathcal{K}}$ let $\mathsf{Caught}_k$ be the event that $m_1^k \notin \mathcal{I}$ and $m_2^k$ cheated in execution $k$ (that is, it is the event the adversary got caught cheating in the $k^{\text{th}}$ execution). Fix an honest server $h \in [\ell] \setminus \mathcal{I}$, and for every $k \in A \cap \overline{\mathcal{K}}$ fix a server $c_k \in \mathcal{I}$ that cheated execution $k$. Observe that for every $k \in A \cap \overline{\mathcal{K}}$ the client will catch the adversary with probability at least

$$\Pr\left[\mathsf{Caught}_k \middle| k \in A\right] \geq \Pr\left[m_1^k = h, m_2^k = c_k \middle| k \in A\right] \geq \frac{1}{\ell(\ell-1)}.$$

Now, observe that for every $k \in [\lambda] \setminus \mathcal{K}$, if $a^k_{m_1^k} = a^k_{m_2^k}$ then either the adversary did not cheat on the $k^{\text{th}}$ execution or $\mathsf{Caught}_k$ did not occur. Therefore,

$$\begin{aligned}
\Pr\left[\mathsf{noAbort}\right] &= \Pr\left[\forall_{k\in[\lambda]\setminus\mathcal{K}}\ a^k_{m_1^k} = a^k_{m_2^k}\right] \\
&\leq \Pr\left[\forall_{k\in([\lambda]\setminus\mathcal{K})\cap A}\ \neg\mathsf{Caught}_k\right] \\
&= \Pr\left[|A\cap\overline{\mathcal{K}}| \leq \frac{\lambda}{8}\right] \cdot \Pr\left[\forall_{k\in([\lambda]\setminus\mathcal{K})\cap A}\ \neg\mathsf{Caught}_k \middle| |A\cap\overline{\mathcal{K}}| \leq \frac{\lambda}{8}\right] \\
&\quad + \Pr\left[|A\cap\overline{\mathcal{K}}| > \frac{\lambda}{8}\right] \cdot \Pr\left[\forall_{k\in([\lambda]\setminus\mathcal{K})\cap A}\ \neg\mathsf{Caught}_k \middle| |A\cap\overline{\mathcal{K}}| > \frac{\lambda}{8}\right] \\
&\leq e^{-\lambda/64} + \Pr\left[\forall_{k\in A\cap\overline{\mathcal{K}}}\ \neg\mathsf{Caught}_k \middle| |A\cap\overline{\mathcal{K}}| > \frac{\lambda}{8}\right] \\
&\leq e^{-\lambda/64} + \left(1 - \frac{1}{\ell(\ell-1)}\right)^{\lambda/8} \\
&\leq e^{-\lambda/64} + e^{-\frac{\lambda}{8\ell(\ell-1)}}.
\end{aligned}$$

We conclude that

$$\Pr\left[\mathsf{noAbort} \wedge \mathsf{Fail}\right] \leq 2e^{-\lambda/64} + e^{-\frac{\lambda}{8\ell(\ell-1)}} + \mathrm{neg}(N).$$

□

We now use Lemma 5.3 to show that Equation (3) holds. Specifically, we show that conditioned that the event $\mathsf{noAbort} \wedge \mathsf{Fail}$ does *not* occur, the two ensembles are statistically close. First, note that by the $t$-privacy of $\Pi$, the queries that $\mathcal{B}$ receives in the real world are statistically close to the queries it receives from $\mathsf{Sim}$ in the ideal world. Therefore, its responses are statistically close. Now, if the client aborts in the real world, then there exists a test execution where it gets different responses. Thus, the same holds in the ideal world (except with negligible probability), hence the simulator changes its database. This means that in both worlds, the client outputs $\perp$. Otherwise, since we condition on $\mathsf{noAbort} \wedge \mathsf{Fail}$ not occurring, the client outputs $D_i$ in both worlds. Finally, setting $\lambda = \omega(\ell^2 \cdot \log N)$ implies that $\mathsf{noAbort} \wedge \mathsf{Fail}$ occurs with negligible probability, which completes the proof. □

Using the $t$-private $\ell$-server PIR protocol of Woodruff and Yekhanin [46] we obtain the following.

**Corollary 5.4.** *For every $\ell \in \mathbb{N}$ and $t \in [\ell - 1]$ there exists a $t$-secure-with-abort $\ell$-server PIR protocol with communication complexity*

$$\omega(\log N) \cdot \frac{\ell^4}{t} \cdot \log \ell \cdot N^{1/\lfloor(2\ell-1)/t\rfloor}.$$

Applying Theorem 5.1 to the protocol resulting from the compiler of Barkol et al. [5] applied to the 2-server protocol of Dvir and Gopi [20] yields the following.

**Corollary 5.5.** *For every $t \in \mathbb{N}$, there exists a $t$-secure-with-abort $2^t$-server PIR protocol with communication complexity*

$$2^{\max\left\{\tilde{O}\left(\sqrt{\log N}\right), t\log t\right\}}.$$

Finally, for $3^t$-server PIR protocols, Barkol et al. [5] and Efremenko [22] gives the following.

**Corollary 5.6.** *For every $t \in \mathbb{N}$, there exists a $t$-secure-with-abort $3^t$-server PIR protocol with communication complexity*

$$3^{2t} \cdot \max\left\{t2^{\tilde{O}\left(\sqrt{\log N}\right)}, t \cdot 3^t\right\}.$$

**Remark 5.7** (On the insecurity of the compiler of [23])**.** *In the original construction of Eriguchi et al. [23], instead of computing $\mathsf{maj}_{k\in\mathcal{K}}\{y^k\}$, the client also tests whether all of these values are equal. If not, it outputs $\perp$, and if they are equal, the client outputs this value. Although their protocol is correct, $t$-private, and $t$-error-detecting, it is not $t$-secure-with-abort. Intuitively, this is because in the underlying protocol $\Pi$ the adversary can cause the client to output an incorrect value for different inputs $i$ (recall that $\Pi$ is not guaranteed to satisfy any robustness property). This causes the client to abort only on certain inputs; thus it has the same selective abort issue presented in Examples 4.4 and 4.5. We next give details.*

*Consider the protocol $\Pi$, where each server sends $D$ to the client, and the client computes the output as follows. If $i = 1$ then output the $i^{th}$ entry of the database sent by $S_1$, and otherwise, output the $i^{th}$ entry of the database sent by $S_2$. Clearly this protocol is correct and private. However, the compiled protocol of Eriguchi et al. [23] is not 1-secure-with-abort since an adversary corrupting $S_1$ can force the client to output an incorrect value only on $i = 1$. Therefore, the client will abort only when $i = 1$.*

## Bibliography


[1] B. Alon, A. Beimel, and O. Lasri. Simplified PIR and CDS protocols and improved linear secret-sharing schemes. *IACR Cryptol. ePrint Arch.*, page 1599, 2024. URL https://eprint.iacr.org/2024/1599.

[2] A. Ambainis. Upper bound on the communication complexity of private information retrieval. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proc. of the 24th International Colloquium on Automata, Languages and Programming*, volume 1256 of *Lecture Notes in Computer Science*, pages 401–407, 1997.

[3] H. Amran. Constructing multi-servers private information retrieval protocols. Master's thesis, Ben-Gurion University, 2016.

[4] K. A. Banawan and S. Ulukus. The capacity of private information retrieval from byzantine and colluding databases. *IEEE Trans. Inf. Theory*, 65(2):1206–1219, 2019. doi: 10.1109/TIT.2018.2869154. URL https://doi.org/10.1109/TIT.2018.2869154.

[5] O. Barkol, Y. Ishai, and E. Weinreb. On locally decodable codes, self-correctable codes, and $t$ -private PIR. In M. Charikar, K. Jansen, O. Reingold, and J. D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, volume 4627 of *Lecture Notes in Computer Science*, pages 311–325. Springer, 2007.

[6] A. Beimel and Y. Ishai. Information-theoretic private information retrieval: A unified construction. In F. Orejas, P. G. Spirakis, and J. van Leeuwen, editors, *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 912–926. Springer, 2001.

[7] A. Beimel and Y. Stahl. Robust information-theoretic private information retrieval. In S. Cimato, C. Galdi, and G. Persiano, editors, *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, volume 2576 of *Lecture Notes in Computer Science*, pages 326–341. Springer, 2002.

[8] A. Beimel, Y. Ishai, and E. Kushilevitz. General constructions for information-theoretic private information retrieval. *Journal of Computer and System Sciences*, 71(2):213–247, 2005. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2005.03.002.

[9] A. Beimel, Y. Ishai, R. Kumaresan, and E. Kushilevitz. On the cryptographic complexity of the worst functions. In Y. Lindell, editor, *TCC 2014*, volume 8349 of *Lecture Notes in Computer Science*, pages 317–342. Springer, 2014.

[10] S. Ben-David, Y. T. Kalai, and O. Paneth. Verifiable private information retrieval. In E. Kiltz and V. Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part III*, volume 13749 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2022.

[11] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In R. Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 97–106. IEEE Computer Society, 2011. doi: 10.1109/FOCS.2011.12. URL https://doi.org/10.1109/FOCS.2011.12.

[12] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT '99*. Springer Berlin Heidelberg, 1999.

[13] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[14] Y. M. Chee, T. Feng, S. Ling, H. Wang, and L. F. Zhang. Query-efficient locally decodable codes of subexponential length. *Computational Complexity*, 22(1):159–189, 2013. doi: 10.1007/s00037-011-0017-1. URL https://doi.org/10.1007/s00037-011-0017-1.

[15] B. Chor and N. Gilboa. Computationally private information retrieval (extended abstract). In F. T. Leighton and P. W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 304–313. ACM, 1997.

[16] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*, pages 41–50. IEEE Computer Society, 1995.

[17] S. Colombo, K. Nikitin, H. Corrigan-Gibbs, D. J. Wu, and B. Ford. Authenticated private information retrieval. In J. A. Calandrino and C. Troncoso, editors, *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023*, pages 3835–3851. USENIX Association, 2023.

[18] C. Devet, I. Goldberg, and N. Heninger. Optimally robust private information retrieval. In T. Kohno, editor, *Proceedings of the 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012*, pages 269–283. USENIX Association, 2012.

[19] Y. Dodis, S. Halevi, R. D. Rothblum, and D. Wichs. Spooky encryption and its applications. In M. Robshaw and J. Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 93–122. Springer, 2016. doi: 10.1007/978-3-662-53015-3_4. URL https://doi.org/10.1007/978-3-662-53015-3_4.

[20] Z. Dvir and S. Gopi. 2-server PIR with subpolynomial communication. *J. ACM*, 63(4):39:1–39:15, 2016. doi: 10.1145/2968443. URL https://doi.org/10.1145/2968443.

[21] C. Dwork, M. Langberg, M. Naor, K. Nissim, and O. Reingold. Succinct proofs for np and spooky interactions. *Unpublished manuscript, available at http://www. cs. bgu. ac. il/˜ kobbi/papers/spooky_ sub_crypto. pdf*, 2004.

[22] K. Efremenko. 3-query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012.

[23] R. Eriguchi, K. Kurosawa, and K. Nuida. On the optimal communication complexity of error-correcting multi-server PIR. In E. Kiltz and V. Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part III*, volume 13749 of *Lecture Notes in Computer Science*, pages 60–88. Springer, 2022.

[24] R. Eriguchi, K. Kurosawa, and K. Nuida. Multi-server PIR with full error detection and limited error correction. In D. Dachman-Soled, editor, *3rd Conference on Information-Theoretic Cryptography, ITC 2022, July 5-7, 2022, Cambridge, MA, USA*, volume 230 of *LIPIcs*, pages 1:1–1:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[25] R. Eriguchi, K. Kurosawa, and K. Nuida. Efficient and generic methods to achieve active security in private information retrieval and more advanced database search. In M. Joye and G. Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland,*

*May 26-30, 2024, Proceedings, Part V*, volume 14655 of *Lecture Notes in Computer Science*, pages 92–121. Springer, 2024.

[26] C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 803–815. Springer, 2005.

[27] F. Ghasemi, S. Kopparty, and M. Sudan. Improved pir schemes using matching vectors and derivatives. Technical Report 2411.11611, Cryptology ePrint Archive, 2024. URL https://arxiv.org/abs/2411.11611.

[28] I. Goldberg. Improving the robustness of private information retrieval. In *2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA*, pages 131–148. IEEE Computer Society, 2007.

[29] O. Goldreich. *Foundations of Cryptography – VOLUME 2: Basic Applications*. Cambridge University Press, 2004.

[30] Y. Ishai and E. Kushilevitz. Improved upper bounds on information-theoretic private information retrieval (extended abstract). In J. S. Vitter, L. L. Larmore, and F. T. Leighton, editors, *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 79–88. ACM, 1999.

[31] T. Itoh. Efficient private information retrieval. *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, E82-A(1):11–20, 1999.

[32] T. Itoh and Y. Suzuki. New constructions for query-efficient locally decodable codes of subexponential length. *CoRR*, abs/0810.4576, 2008.

[33] T. Itoh and Y. Suzuki. Improved constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions on Information and Systems*, E93-D(2):263–270, 2010. ISSN 1745-1361. doi: 10.1587/transinf.e93.d.263. URL http://dx.doi.org/10.1587/transinf.E93.D.263.

[34] J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In F. F. Yao and E. M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 80–86. ACM, 2000.

[35] K. S. Kedlaya and S. Yekhanin. Locally decodable codes from nice subsets of finite fields and prime factors of Mersenne numbers. Technical Report TR07-040, Electronic Colloquium on Computational Complexity, 2007.

[36] I. Kerenidis and R. de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences*, 69(3):395–420, 2004.

[37] K. Kurosawa. How to correct errors in multi-server PIR. In S. D. Galbraith and S. Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part II*, volume 11922 of *Lecture Notes in Computer Science*, pages 564–574. Springer, 2019. doi: 10.1007/978-3-030-34621-8_20. URL https://doi.org/10.1007/978-3-030-34621-8_20.

[38] E. Kushilevitz and R. Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 364–373. IEEE Computer Society, 1997.

[39] T. Liu, V. Vaikuntanathan, and H. Wee. Conditional disclosure of secrets via non-linear reconstruction. In J. Katz and H. Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 758–790. Springer, 2017.

[40] T. Liu, V. Vaikuntanathan, and H. Wee. Towards breaking the exponential barrier for general secret sharing. In *EUROCRYPT 2018*, volume 10820 of *LNCS*, pages 567–596, 2018.

[41] E. Mann. Private access to distributed information. Master's thesis, Technion – Israel Institute of Technology, Haifa, 1998.

[42] S. Micali and P. Rogaway. Secure computation (abstract). In J. Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 392–404. Springer, 1991.

[43] H. Sun and S. A. Jafar. The capacity of private information retrieval. *IEEE Trans. Inf. Theory*, 63(7):4075–4088, 2017. doi: 10.1109/TIT.2017.2689028. URL https://doi.org/10.1109/TIT.2017.2689028.

[44] H. Sun and S. A. Jafar. The capacity of robust private information retrieval with colluding databases. *IEEE Trans. Inf. Theory*, 64(4):2361–2370, 2018. doi: 10.1109/TIT.2017.2777490. URL https://doi.org/10.1109/TIT.2017.2777490.

[45] S. Wehner and R. de Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *Proc. of the 32nd International Colloquium on Automata, Languages and Programming*, volume 3580 of *Lecture Notes in Computer Science*, pages 1424–1436, 2005.

[46] D. P. Woodruff and S. Yekhanin. A geometric approach to information-theoretic private information retrieval. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 275–284. IEEE Computer Society, 2005. doi: 10.1109/CCC.2005.2. URL https://doi.org/10.1109/CCC.2005.2.

[47] E. Yang, J. Xu, and K. Bennett. Private information retrieval in the presence of malicious failures. In *Proceedings 26th Annual International Computer Software and Applications*, pages 805–810, 2002. doi: 10.1109/CMPSAC.2002.1045104.

[48] S. Yekhanin. Towards 3-query locally decodable codes of subexponential length. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 266–274, 2007.

[49] L. F. Zhang and R. Safavi-Naini. Verifiable multi-server private information retrieval. In I. Boureanu, P. Owesarski, and S. Vaudenay, editors, *Applied Cryptography and Network Security - 12th International Conference, ACNS 2014, Lausanne, Switzerland, June 10-13, 2014. Proceedings*, volume 8479 of *Lecture Notes in Computer Science*, pages 62–79. Springer, 2014.