Secure-In-Between-Accesses Database Security

Gregory Valiant
Stanford University
gvaliant@cs.stanford.edu

Paul Valiant
Brown University
pvaliant@gmail.com

Abstract

We introduce a definition of security that applies to databases that maintain dynamic information on users such as financial account information, medical records, etc. Such a database system is *secure in between accesses* provided 1) users can efficiently access their data, and 2) while a user is not accessing their data, the user's information is *information theoretically* secure to malicious agents. We propose a realization of such a database system and prove that a user's stored information, in between times when it is being legitimately accessed, is *information theoretically* secure both to adversaries who interact with the database in the prescribed manner, as well as to adversaries who have installed a virus that controls the entire internet-facing server that stores the database. We stress that the security guarantee is information theoretic and everlasting: it relies neither on unproved hardness assumptions, nor on the assumption that the adversary is computationally or storage bounded.

The central idea behind our design of an information theoretically secure database system is the construction of a "re-randomizing database" that periodically changes the internal representation of the information that is being stored. To ensure security, these remappings of the representation of the data must be made sufficiently often in comparison to the amount of information that is being communicated from the database between remappings and the amount of local memory on the database server that a virus may preserve during the remappings. While this changing representation provably foils the ability of an adversary to glean information, it can be accomplished in a manner transparent to the legitimate users, preserving how database users access their data.

The core of the proof of the security is a new analysis of a simple and explicit locally computable extractor, based on a hypercontractivity inequality, optimized for the relatively unexplored parameter regime of "nearly full min-entropy" where the min-entropy of the n-bit input string is $n - o(n)$.

Our analysis of this extractor can also be interpreted as establishing the following communication/data tradeoff for the problem of learning sparse parities from uniformly random n-bit examples. Fix a set $S \subset \{1, \ldots, n\}$ of size k: given access to examples $x_1, \ldots, x_t$ where $x_i \in \{0,1\}^n$ is chosen uniformly at random, conditioned on the XOR of the components of x indexed by set S equalling 0, any algorithm that learns the set S with probability at least p and extracts at most r bits of information from each example, must see at least $p \cdot \left(\frac{n}{r}\right)^{k/2} c_k$ examples, for $c_k \geq \frac{1}{4} \cdot \sqrt{\frac{(2e)^k}{k^{k+3}}}$. The r bits of information extracted from each example can be an arbitrary (adaptively chosen) function of the entire example, and need not be simply a subset of the bits of the example.

1 Introduction

A significant portion of our sensitive data, including bank account information, medical records, tax liabilities, and the contents of one's inbox, is both accessible online and *dynamic* in that the data may change over time, even without any direct knowledge or oversight of the user. The question of how to maintain access to this dynamic data, while preserving its security, is a challenge of central importance. Many of the most worrying and expensive hacks to date have not been due to failures in the transmission of encrypted data, but rather due to large-scale attacks on the database servers themselves. Stated differently, it is often not the case that a user requests his/her information, and that information is compromised in transmission; instead, in between a user's accesses, an adversary hacks into a database and downloads thousands or millions of sensitive entries. (See Figure 1 for two such examples.)

There are, perhaps inherently, many risks involved when a user actively accesses or uses sensitive data, ranging from submitting such information to an unintended website or recipient, insecurity in transmission (either via an insecurely managed wifi network, or further upstream), to someone watching your computer screen from the chair behind you. Indeed, it is the mantra of "leakage-resilient cryptography" that *every computation has observable side-effects* [31], and thus any time a user accesses sensitive data, they risk revealing aspects of it. Nevertheless, one might hope that, at least while one is *not* accessing or using sensitive information, such information is completely secure. The hope is that one should not need to worry about information being stolen from that highschool email account that you haven't checked in a decade, or from the three bank accounts that you forgot existed.

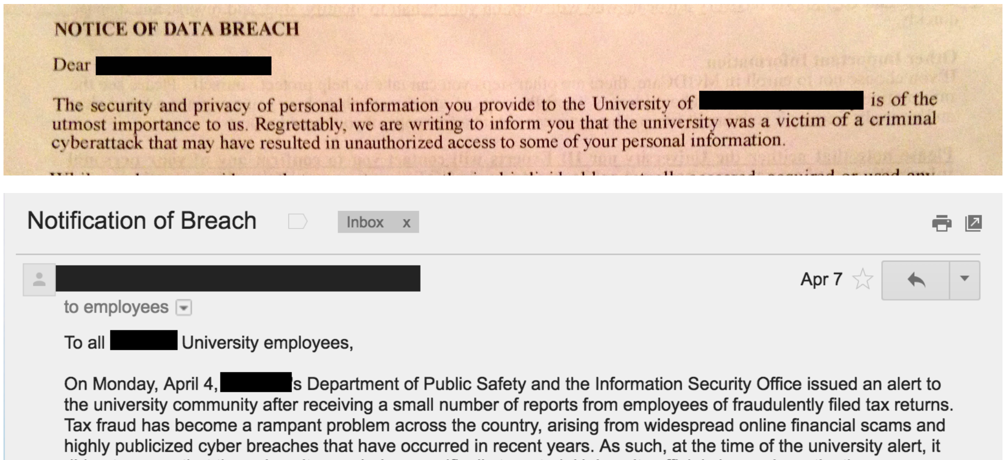
We introduce a notion of database security that applies to both static and dynamic data, which we term *secure in between accesses* ("SIBA security") that promises 1) users can easily and conveniently access their information, and 2) a user does not need to worry about the security of their information while they are not accessing it—as long as one is not accessing one's data, it is *information theoretically* secure.[1] And this security continues to apply even if an adversary has exogenously produced "hints" that are arbitrary functions of users' past secret data—for example even if an adversary knows the users' data at all previous times, the current data is still information theoretically secure (and, for example, the adversary cannot detect whether the data has changed).

Definition 1. *In a* ***secure in between accesses*** *database with security parameter ϵ, each user is initially provided with a secret key S, and can interact with an internet-facing server A that provides access to the user's time-varying data u_t, where time t is measured in discrete epochs.*

We say that a user's time-varying data is ϵ-secure-in-between-accesses if, for any adversary E that has complete control of the server A during each epoch when the user does not access her data, and where the adversary has complete access to all the other users in the system, there is an efficient simulator sim *(efficient, given black-box access to E), such that for any sequence of user data u_i and any sequence of hints H_i that are (possibly probabilistic) functions of the entire sequence $\{u_i\}$: letting V_t^E be the view of the adversary after t epochs receiving hint H_i after each previous epoch i, and letting V_t^{sim} be the view of the simulator who is given the same hints, but* ***never*** *interacts with the database, then the statistical distance between the pairs of distributions (S, V_t^E) and (S, V_t^{sim}) is at most $t \cdot \epsilon$.*

The closeness of these joint distributions immediately also implies that the marginal of the second components are close, meaning that the adversary cannot learn anything that they could not have figured out without access to the database. Further, the closeness of the two pairs of distributions (S, V_t^E) and (S, V_t^{sim}) also implies two different more nuanced kinds of security: if the adversary somehow learns something about the user's time-varying data stored in the database

[1] One simplistic analogy is to the role of a bank. There are many ways to be robbed or defrauded while spending money; the promise of a bank is that 1) you can easily and conveniently access your money, and 2) you do not need to worry about your money while you are not using it.


NOTICE OF DATA BREACH

Dear [redacted]

The security and privacy of personal information you provide to the University of [redacted] is of the utmost importance to us. Regrettably, we are writing to inform you that the university was a victim of a criminal cyberattack that may have resulted in unauthorized access to some of your personal information.

Notification of Breach Inbox x

[redacted] Apr 7

to employees

To all [redacted] University employees,

On Monday, April 4, [redacted]s Department of Public Safety and the Information Security Office issued an alert to the university community after receiving a small number of reports from employees of fraudulently filed tax returns. Tax fraud has become a rampant problem across the country, arising from widespread online financial scams and highly publicized cyber breaches that have occurred in recent years. As such, at the time of the university alert, it

Figure 1: Two notices that one of the authors received. In both cases, the personal information was not stolen in transmission—in fact, in the first case, the author has not attempted to access any information from that university's computer system during the past 5 years. A *Secure In Between Accesses* database provides strong defenses against such information theft.

(expressed via arbitrary hints H_i, for example, guessing the most significant bits of the user's bank balance), then even by comparing these hints to information extracted from the database itself, the adversary will learn at most $t \cdot \epsilon$ bits of information about the secret key S and therefore any future bits stored in the database will remain secure; second, if after the database shuts down, the adversary somehow finds the user's entire secret key S, this will still give at most $t \cdot \epsilon$ information about the bits that were formerly stored in the database—this is the notion of "everlasting security" [4].

We present an instantiation of a SIBA-secure database via the combination of a simple explicit protocol and a new physical assumption: a traffic counter on the internet connection of the server that cuts the connection once a certain number of bits, r, have been transmitted in an epoch, and signals a non-internet-connected back-end server to start a new epoch.

This proposal for a SIBA-secure database seems practically feasible for lower-throughput systems, such as bank information, medical records, school transcripts, tax documents, etc., which are some of the settings where the cost of data breaches vastly outweigh the potential cost of data transmission for reasonable security parameters in our proposed scheme. Thus the emphasis in this paper is not on high throughput, but rather, on achieving high security for the many extremely sensitive yet lower-throughput databases on the internet.

The new data protection regulations that have been passed by the European Commission, require that any entity storing personal data on users both makes that data available to the users, as well as ensures that such data be stored securely.[2] The requirements of these regulations offer one potentially widespread future use-case for the SIBA security definition. Indeed, one imagines that typical users might never request their data from many of these user-data-aggregator entities, or request it only very rarely. In such cases, the strong SIBA security guarantees would be appropriately robust reassurances.

The central idea behind our design of a SIBA-secure database system is to have the database's representation of the information it stores remap periodically in such a way that 1) honest database

[2] For more information on the data protection regulations, see `http://ec.europa.eu/justice/data-protection/`.

users can continue to transparently access their data by XORing together the information at a small (e.g. $k = 10$) unchanging set of addresses, but 2) adversaries are information theoretically unable to piece together information from the different representations in a sufficiently consistent manner to glean any knowledge. To ensure security, these remappings of the data must occur sufficiently often in comparison to the amount of information that is being communicated from the database. Our database's constantly-changing representation of (possibly unchanged) user data, combined with the strict monitoring of outgoing communication bandwidth from our server comprise the conceptual core of our proposal.

The physical hardware needed to implement our SIBA-secure scheme has three components:

- An internet-facing server A that may be vulnerable to being taken over by a virus;
- A back-end secure server B that has copies of the users' data bits and encryption keys, but is never connected to the internet;
- A secure but simple "traffic counter" that sits astride the cable connecting A to the internet, and when the transmitted traffic reaches a limit r, does the following four mechanical steps: a) it cuts the internet, b) shuts down server A, c) connects the back-end server B to A's hard drives, and d) instructs the back-end server B to begin a new epoch (which induces B to erase A's hard drives and upload a freshly generated database string dat_t).

The information theoretic security guarantees do not extend to the accessing and transmission of a user's data, or the viewing of the data at the user's end. At the time of access, a virus in the database may discover the user's secret addresses in the database and exploit them. Improving the security of the data during accessing and transmission can be attempted via standard oblivious transfer protocols [37, 34, 27] and encryption, though it will lack information theoretic guarantees. For these reasons, we say that our database is "secure in between accesses". As long as a user's data is not being accessed, its security is information theoretically guaranteed.

Theorem 1 (SIBA-security)**.** *We present a database system that stores d time-varying bits, each as the XOR of k secret locations. The database maintains a length $N \geq 2kd$ string, and has the following security guarantee: for each time-varying bit, b, and associated set of k indices, S, any algorithm that extracts at most r bits of information about the database between epochs can correctly guess S with probability at most $\binom{N/2}{k}^{-1} + t \cdot \left(\frac{2r}{N}\right)^{k/2} \cdot 4\sqrt{\frac{k^{k+3}}{(2e)^k}}$ after t epochs, even for an adversary that knows all $d-1$ other bits and secret keys. Furthermore, the security of bit b is "everlasting": suppose this bit takes values $b_1, \ldots, b_{t-1}$ for the first $t-1$ epochs, and these bits are known to the adversary ahead of time. If b_t is chosen at random from $\{0,1\}$, and the adversary extracts at most r bits of information from the database during each of the first t epochs, then even if the adversary is given secret set S after the $t+1$st epoch, the adversary can correctly guess b_t with probability at most $1/2 + t \cdot \left(\frac{2r}{N}\right)^{k/2} \cdot 4 \cdot \sqrt{\frac{k^{k+3}}{(2e)^k}}$.*

One reasonable setting of parameters would be to use a database string of size $N = 2 \cdot 10^{12}$ bits, where each bit stored in the database is represented as the XOR of $k = 10$ secret locations and thus the database has a capacity of $d = N/2k = 10^{11}$ bits of user information, with a new epoch starting every $r = 10^8$ bits transmitted. In this case Theorem 1 guarantees information theoretic security except with probability $< 3 \cdot 10^{-17}$, per epoch, for each bit stored in the database.

The above theorem, restricted to the case $d = 1$, can also be viewed as a communication/data tradeoff for the problem of learning k-sparse parities over n-bit examples:

Corollary 1. *Choose a uniformly random set* $S = \{s_1, \ldots, s_k\} \subset \{1, \ldots, n\}$ *of size* k*: given access to a stream of examples* $x_1, \ldots, x_t$ *where each* $x_i \in \{0,1\}^n$ *is chosen uniformly at random, conditioned on the XOR of the components of* x_i *with indices in* S *being* b_i*, any algorithm, given* $b_1, \ldots, b_t$*, that must (even adaptively) compress each example to at most* r *bits before seeing the next example can correctly guess the set* S *with probability at most* $\binom{n}{k}^{-1} + t \cdot \left(\frac{r}{n}\right)^{k/2} \cdot 4\sqrt{\frac{k^{k+3}}{(2e)^k}}$.

This result can be viewed as mapping the intermediate regime between the communication/data tradeoffs given by Ohad Shamir [41] for the case $k = 1$ (the "hide-and-seek" problem of detecting a biased index from otherwise uniformly random length n sequences), and the results in Steinhardt, Valiant, and Wager [44] for the case $k = \theta(n)$, which shows that any algorithm that extracts less than $n - c$ bits of information from each example, must see at least $2^{\theta(c)}$ examples.

2 Related Work

We briefly survey related work in a number of different areas, including work on information theoretically secure encryption in the "bounded storage model" [30] and connections with local extractors [45], work on "leakage-resistant" cryptography in both the "bounded retrieval model" [9, 15] and the continual leakage model [21, 11] and "forward secure storage" [16], and the related work from the complexity and learning theory communities on tradeoffs between the amount of information extracted from each data point and the number of data points necessary to learn certain functions.

Information Theoretic Security and Locally Computable Extractors. While most security and cryptographic schemes rely on the assumptions that adversaries are computationally bounded and that certain problems require super-polynomial amounts of computation to solve, there has been significant attention on devising stronger, unconditional security guarantees based on information theory. This direction, in some sense, began with the proposal of the "one-time-pad" in the late 1880's [32] (which was later re-discovered and patented in the 1920's by Vernam [46]), and Shannon's "Communication Theory of Secrecy Systems" [42] in the 1940's. More recently, Maurer introduced *bounded-storage cryptography*, that considered cryptographic protocols with information theoretic security, under the assumption that the adversary has limited memory [30] (also see the survey [29]). Bounded memory cryptography typically considers a setting where all have access to some enormous stream of random bits (for example, random bits that are being continuously broadcast from a satellite), and that the users interact with the stream by selectively storing certain bits. There has been a great deal of work in this bounded-storage model, including more recent work on "everlasting" encryption in this model, in which information theoretic security persists even if the secret key is divulged at some later point in time (assuming the adversary has bounded storage) [4, 10, 28].

The "everlasting security" analysis was later reframed by Lu in terms of *extractors*, where the fact that an extractor "cleans up" a distribution until it is very nearly uniform is recast as implying that, even if some data about a uniformly-chosen key has leaked to an adversary making the posterior distribution non-uniform, then an extractor-based encryption scheme will "clean" this up and yield an output that is exponentially close to uniform, and hence secure [28]. Our work can be thought of as giving improved guarantees for this style of analysis in the regime where a sub-constant fraction of the data has leaked.

Our secure database scheme can also be viewed as a new extractor analysis with significantly improved parameters for the regime of n-bit strings with min-entropy $n - o(n)$. Our extractor is simple and explicit, consisting of sampling a random set of k locations in the length-n string and returning their XOR. This extractor falls into the "sample-then-extract" category introduced by Vadhan [45] which identified a general framework for constructing new extractors based on separately analyzing the processes of a) sampling k random locations and b) applying a smaller extractor $f : \{0,1\}^k \to \{0,1\}$ to compute the final output. However, our extractors have significantly better properties than are implied by this more general analysis—for a min-entropy of $n - r$, we show via a new application of hypercontractivity inequalities that our extractor output's distance from uniform is roughly $(\frac{r}{n})^{k/2}$, whereas the analysis of [45] (and also previous works including [28]) yields a bound of only $2^{-O(k)}$, with somewhat worse constants and, crucially, no improvement as r becomes sublinear. We note that here, the emphasis is on minimizing the distance to uniformity, and maintaining small locality, k, since the distance corresponds to the security guarantee and k translates to the multiplicative communication overhead in retrieving information in the secure database scheme. In particular, in our locally computable extractor construction, we are not trying to minimize the seed length, which is one of the main focuses in [45] and other works on extractors.

At a higher level, our proposal, in which a database monitors the amount of information communicated from the database, and refreshes its representation of the data, can be thought of as replacing the enormous stream of random bits of the bounded storage setting via an interaction model in which a database may measure the number of bits communicated, and refresh itself accordingly. From a practical perspective, this is a significantly different interaction model from previous approaches to information theoretic security and cryptography.

Leakage-Resistant Cryptography and Storage There is a large body of recent work on security and cryptography in several models that admit some "leakage" of information to the adversary. In the "Bounded Retrieval Model" [9, 15], the assumption is that an adversary has the ability to leak at most some fixed amount of information, ℓ, and the question is typically how one can enlarge the representation of a secret key in such a way that the encryption security is preserved given the adversary has at most ℓ bits of leaked information about the key. The challenge is how to enlarge the secret key without inconveniencing the honest user—specifically, the user would like to be able to use the key by only querying a small number of indices [2, 1, 12] (we also refer the reader to the survey [3]). This line of work has typically focused on encryption and secret sharing, with the emphasis on preserving computational hardness guarantees, rather than the information theoretic emphasis of this paper.

There has been work in the "continual leakage model", where it is assumed that an adversary is capable of leaking at most ℓ bits every time period. The question then becomes how the honest party can periodically refresh the secret keys (or key shares in a secret-sharing setting) so as to ensure that security guarantees persist even as the adversary obtains more and more information [21, 11, 13, 26]. Earlier work on "proactive" secret sharing [22] seeks to tackle a similar setting. These settings are related to the model we consider, in the sense that both settings assume that we are able to update the internal representations periodically, ensuring that the amount of information leaked between updates is bounded. The more recent results on cryptography in the continual leakage model are generally concerned with refreshing representations of secret keys (as opposed to the cyphertext), and the security guarantees are based on computational hardness (instead of information theoretic hardness as in this paper).

We also point out the related work of Micali and Reyzin [31] that considers a different model of information leakage, according to the assumption that *only computation leaks information.* This model is designed to capture side-channel attacks such as detailed measurements of the power drain during a computation or other physical properties of the machine, as opposed to cyberattacks. Work in this line considers signature schemes and encryption schemes, and generally leverages this restriction on the manner in which information may be leaked, rather than assumptions on the quantity of leaked information [36, 17].

Perhaps most relevant to our work is Dziembowski's work on "Forward Secure Storage" [16], in which they consider the problem of securely storing a ciphertext—by inflating its size—in such a way that if a bounded amount of information about the ciphertext is leaked, and then later the adversary guesses/learns the secret key, then the message is still secure. While the emphasis in our SIBA security definition is different than that of forward secure storage, our notion can be thought of as an extension of the latter to the setting where multiple different users are simultaneously storing information in a single database (ideally without inflating the size of the database linearly with the number of users), and the database must behave in a forward secure manner during every time epoch. Dziembowski constructs both a computationally-secure protocol, as well as observes that information theoretic forward secure storage can be obtained from the everlasting security guarantees in the bounded storage model. As discussed above, our simple explicit extractors and analysis in the high $n - o(n)$ min-entropy regime yield improved security parameters in this regime for the bounded storage model, implying stronger security guarantees in the forward-secure storage model in the regime in which the amount of information leaked is $o(n)$ where n denotes the storage size.

Communication/Data Tradeoffs for Learning. Our analysis of the explicit extractor that is the core of our proof of the information theoretic security of the database system we propose can be interpreted as a result on the tradeoff between communication and the number of examples necessary to learn a parity function over uniformly random examples: given uniformly random n-bit examples subject to the XOR of an unknown subset of k indices is 0, to learn the subset either one must communicate at least r bits of information about some examples, or observe at least $O\left((n/r)^{k/2}\right)$ examples.

There has been significant attention from the learning theory community over the past few decades on understanding such tradeoffs, perhaps beginning with the work of Ben-David and Dichterman [7]. In the distributed setting, there has been significant recent work analyzing how much data is required to learn certain classes of function in the setting in which data is partitioned across servers, and there are restrictions on the communication between the servers, and/or privacy constraints (e.g. [5, 14, 47, 18, 8, 44, 41]). In many cases, these results on communication bounded learning immediately yield analogous statements in the memory bounded streaming setting, where the learner has limited memory, and is given access to a stream of examples.

The proof of correctness for the everlasting security scheme of Ding and Rabin [10] can be interpreted as proving a communication/data tradeoff for the problem of learning parities of size k from random length n examples: either the learner must extract at least $n/6$ bits of information from some example, or observe at least $exp(k)$ examples. In [41], Shamir considers learning with memory and communication constraints, and, among other results, shows that: for the problem of identifying one significantly biased bit from otherwise uniformly random length n examples, any learning algorithm with r bits of memory requires $O(n/r)$ examples to correctly identify the biased index. This corresponds to the problem of learning parities of size $k = 1$, from random examples. In [44], the authors establish a correspondence between the problems that are learnable via algorithms that communicate/extract few bits of information from each example, and those problems learnable in Kearns' *statistical query* model [24]. Additionally, they show that in the distributed setting in which parties are given examples from a length n instance of parity, either some parties must communicate $\theta(n)$ bits of information about their example, or an exponential number of examples/parties are required to learn the parity set. They also conjectured a stronger result, that for any positive $\epsilon > 0$ and sufficiently large n, any algorithm for learning a random parity function over length n examples either requires memory at least $n^2(\frac{1}{4} - \epsilon)$, or must see an exponential number of examples to recover the parity set with high probability. This conjecture was proved by Raz, with the slightly weaker constant of $1/25$ instead of $1/4 - \epsilon$ [39]. Raz also observes that such a result immediately provides an example of a bounded-storage cryptographic scheme where a single bit can be communicated given a length n private key, and time n to encrypt/decrypt, in such a way that it is secure against any adversary with memory less than $n^2/25$. There has been a significant subsequent push to develop a tight understanding of memory/sample tradeoffs for other learning settings (see e.g. [19, 25, 38, 6, 33, 43]).

3 SIBA–Security via Re-Randomizing Databases

Our proposed system works as follows: the database can be regarded as an n bit string x, and each user has a specified set of k indices for each bit to be stored. The user's bit can be accessed by computing the XOR of the database values at the specified k indices. Periodically, the database will replace x with a uniformly random string, subject to the condition that the XOR of the specified indices is still the desired value. The security of the system rests on the ability of the database to ensure that a limited number of bits of information have been communicated to the outside world between these "re-randomizations" of x.

For clarity, we formally describe the proposed system in the case of a single user, "Alice", who wishes to access a single fixed bit of information $b \in \{0, 1\}$.

Box 1: An Information Theoretically Secure Database

The database will consist of n bits, the parameter k denotes the key size, the parameter r denotes the *database refresh rate*, and Alice wishes to store bit $b \in \{0, 1\}$.

Initialization:

- Alice and the database agree on a uniformly random secret, $S = \{s_1, \ldots, s_k\} \subset \{1, \ldots, n\}$ with $|S| = k$.
- The database is initialized to a uniformly random bit string $x \in \{0, 1\}^n$ such that $b = \sum_{i \in S} x_i \mod 2$.

Access:

- To access Alice's bit, b, she requests the values of x at locations $i \in S$, and computes their XOR.

Maintenance:

- The database counts the total number of bits communicated from the database (in aggregate across all users.)
- Before this count reaches r, the count resets to 0 and the database "re-randomizes": it replaces x with a new $x' \in \{0, 1\}^n$ chosen uniformly at random conditioned on $b = \sum_{i \in S} x_i \mod 2$. This starts the next epoch.

The crux of the above proposal—that the database "re-randomizes" before too much information is transmitted about any given instantiation of the database's memory—can be easily guaranteed in practice by ensuring that two properties hold. First, that the database is connected to the world via a communication channel with some known bandwidth, and ensuring that the database re-randomizes conservatively, assuming that the channel is permanently transmitting at full capacity. And second, that there is no large tract of database memory that is skipped in the re-randomizations. Specifically, to ensure that no adversary extracts too much information about the state of the database at a given time, the database must ensure that any virus that might be present (on the database itself) is restricted to a limited amount of memory between re-randomizations (as such memory could be leveraged at a future time to communicate extra information about the current state of the database).

3.1 Time-Varying Data

The above basic design easily extends to support the setting where Alice's bit changes over time. Let b_t denote the bit as a function of time, t, and assume that time is delimited in discrete intervals. The above protocol can be trivially adapted to allow Alice access to the time-dependent bit b_t, by simply ensuring that the database re-randomizes both at each time increment, and when the communication reaches the database refresh rate parameter, r. With a re-randomization that occurs at time t, the database string x is chosen uniformly at random conditioned on the XOR of the indices of x in Alice's secret set equalling b_t. This protocol also extends naturally to the multi-user setting, and the setting where each user stores multiple bits, with security guarantees essentially unchanged, provided the length of the database string is greater than $2k$ times the total number of bits to be stored across all users. We describe these extensions in Section 4.

3.2 Security Guarantees

The following observation characterizes the security guarantees of the above system in the setting in which an adversary can only interact with the database by requesting the database values at specified indices:

Observation 1. *After at most t "re-randomizations" of the database, an adversary that only interacts with the database by requesting the database values at r specified indices per re-randomization can correctly guess Alice's secret subset S with probability at most $t\,(r/n)^k$, even if the adversary knows Alice's sequence of bits $b_1, \ldots, b_t$* a priori. *Furthermore, an adversary who knows Alice's bit values at times $1, \ldots, i-1, i+1, i+2, \ldots, t$, can distinguish the case that $b_i = 0$ from the case that $b_i = 1$ with probability at most $t\,(r/n)^k$.*

The above observation follows from noting that for any subset of indices $Q \subset \{1, \ldots, n\}$ that does not contain the entire set S, the bits of the database at indices in Q will be a uniformly random $|Q|$-length bitstring, and in particular, is independent of Alice's current bit b_i and all of her past and future bits.

The power of the above observation is that even an adversary who knows Alice's data ahead of time cannot leverage this knowledge to any advantage. In practical terms, for a bank implementing such a system, this would mean that even if an adversary steals a paper copy of Alice's bank account balance and all her historical banking information, the adversary cannot leverage this information to glean any additional information about Alice's account, and, for example, will not be able to detect a change to the account balance, or recover any more historic data than what the adversary already has. Further, this simple setting has the "everlasting security" property [4] that, if after the database is shut down, the adversary later learns the locations of Alice's secret bit locations, the adversary will not be able to recover any of Alice's secrets (unless, as happens with probability

$t(r/n)^k$, during one of the t periods the adversary had previously got lucky and simultaneously observed all k of the bits at Alice's locations).

The following theorem, which is our main result, shows that similar information theoretic security persists even in the presence of significantly more pernicious attacks on the database. Suppose an adversary hacks into the database and installs a virus that allows the adversary to interact with the database in a more general fashion, rather than simply querying the database value at specified indices. Even if the virus can compute arbitrary functions of the *entire* database and transmit these function evaluations to the adversary, the database will be secure with essentially the same bounds provided at most r bits have been transmitted between "re-randomizations". This security holds even if the adversary has infinite memory and computational power, and can communicate arbitrary amounts of information to the virus—for example, even in the setting where the adversary is able to upload a new virus with every re-randomization. Alternatively, instead of assuming a bound of r communication to the outside world, the same results hold when bounding the communication of the virus to its future self: the assumption that the database can ensure that no virus preserves more than r local memory on the database between re-randomizations is practically feasible as the database simply needs to ensure that there is no very-large tract of memory that is left untouched during the "re-randomizations".

Theorem 2 (SIBA-security)**.** *Given the database system described in Box 1, with key size k that maintains an n-bit string, any algorithm that extracts at most r bits of information about the database between epochs can correctly guess Alice's secret set, S, with probability at most $\binom{n}{k}^{-1} + t \cdot \left(\frac{r}{n}\right)^{k/2} \cdot 4\sqrt{\frac{k^{k+3}}{(2e)^k}}$ after t epochs. Furthermore, the security of Alice's data is "everlasting": suppose Alice stores bits $b_1, \ldots, b_{t-1}$ in the database for the first $t-1$ epochs, and these bits are known to the adversary ahead of time. If Alice then chooses bit b_t at random from $\{0,1\}$, and the adversary extracts at most r bits of information from the database during each of the first t epochs, then even if the adversary is given Alice's secret set S after the $t+1$st epoch, the adversary can correctly guess Alice's bit with probability at most $1/2 + t \cdot \left(\frac{r}{n}\right)^{k/2} \cdot 4 \cdot \sqrt{\frac{k^{k+3}}{(2e)^k}}$.*

The proof of Theorem 2 is given in Section 5.

4 Multiple Users and Multiple Bits

We now describe how to incorporate multiple users, each storing multiple bits, in our database protocol; we do this in such a way that the security guarantees and the total database size are essentially independent of the number of bits stored. Each (possibly time-varying) bit to be stored will have its own associated secret set of k indices, disjoint from the sets corresponding to all other bits (whether from the same user or a different user). To construct a length-N string that stores d bits collectively, across various users: for each bit $b_{i,t}$, to be stored in locations $S_i = (h_{i,1}, \ldots, h_{i,k})$, the database independently chooses a random set of k bits of parity $b_{i,t}$ and assigns them to the locations $h_{i,1}, \ldots, h_{i,k}$ in the string; the remaining locations in the string are chosen to be independent coin flips. The security guarantees on a length-N database storing d bits collectively across all users result from the following observation: for each secret bit of a user, even assuming the adversary has complete information about all aspects of all $d-1$ remaining bits (from this user and the other users), then the database setup for the remaining secret bit is effectively identical to the standard setup of Theorem 2 for a database with string length $n = N - (d-1)k$.

Thus, provided the size of the database representation, N, is at least $n - k$ bits larger than k times the number of bits being stored, the security guarantees of Theorem 2 persist. Even in the event that an adversary has prior knowledge of some of the secret sets and bits, with all but negligible probability, the adversary cannot leverage this knowledge to gain any additional knowledge:

Corollary 2. *Consider a database system as described above that maintains a length N string, has key size k (per bit stored), and stores a total of d bits $b_{1,t}, \ldots, b_{d,t}$, with the ith bit corresponding to the XOR of k indices $h_i \subset \{1, \ldots, n\}$, where the sets h_i are disjoint. The security guarantees of Theorem 2 hold for $n = N - (d-1)k$, for each given bit $b_{i,t}$ and secret set S_i, even if the adversary knows partial or complete information about the remaining $d-1$ bits $b_{j,t}$ and secret key sets S_j.*

Proof. We consider the case when the adversary has *complete* information about the remaining $d-1$ bits and secret key sets, as such an adversary can accomplish at least as much as one with only partial information. Consider ignoring all the bits in the $(d-1)k$ known secret key locations $h_{2,1}, \ldots, h_{d,k}$ from the database string of length N, the (joint) distribution of what remains is identical to the construction of a single bit in a database of size $n = N - (d-1)k$, since each of the secret key locations is chosen independently and disjointly, and for each k-tuple of secret locations the k bits at these locations are chosen independently of the rest of the database, and each bit not at a secret location is chosen by an independent coin flip. □

The above proof can alternatively be viewed as follows: given a database of size n securely storing a single bit, the adversary could easily simulate having access to a database of size $N = n+(d-1)k$ securely storing d bits, where the adversary knows everything about the $d-1$ simulated secrets (because the adversary simulated adding them to the database). If there were any way to extract information about one of the bits in the d-bit setting by leveraging partial or complete information about the other $d-1$ bits, then an adversary could simulate this attack in the single-bit setting, contradicting Theorem 2. Theorem 2 and Corollary 2 are stated in combined form in the introduction as Theorem 1.

4.1 Decreasing the Key Size

In this multiple-bit setting, as described above, a user will store sk secret indices for every s bits of information that she wishes to store securely. There are many natural approaches to improving this scaling of parameters, including analogs of the pseudorandom generators induced by random walks that were used in a related setting to substantially decrease the key size for a user, so as to be sublinear in the number of bits she wants to store [28].

5 Proof of Theorem 2

We begin with a high level overview of the proof of Theorem 2. Because our proof relies on properties of polynomials of random ± 1 variables, which lets one express the parity of k bits as a degree k monomial, for the entirety of this section we will refer to bits as being ± 1 valued rather than 0/1 valued (as was done in the rest of the paper). Given an n-bit database dat from which an adversary runs an arbitrary computation returning an r-bit output OUT_{dat}, the challenge is to show that OUT_{dat} gives essentially no information about either of the two aspects of the database we wish to keep secret: the user's secret key, specified by k locations $h_1, \ldots, h_k$; and the user's secret itself, which is stored as the XOR of these k locations in the database. Consider the portion of the hypercube of possible databases $dat \leftarrow \{-1, 1\}^n$ that induces a particular output OUT—because there are only 2^r possible r-bit outputs, a typical output OUT must be induced by a relatively large, 2^{-r} fraction of the hypercube. The main technical step is arguing why, for any large subset of the hypercube, most k-bit parities will return almost exactly as many 1's as -1's on this set. In other words, the only subsets of the hypercube that are biased for many parities are those subsets that are very small. A singleton set is biased for *all* parities but consists of a 2^{-n} fraction of the cube; the set of points whose first k coordinates are 0 is a fairly large fraction of the hypercube, but is only strongly biased for the k-way parity consisting of exactly the first k bits; in general, large subsets of the hypercube are very close to unbiased, on a typical parity. We analyze this situation in Proposition 1.

Given this tool, the single time-step ($t = 1$) special instance of Theorem 2 follows by the straightforward argument that, even given r bits of output from the database, the joint conditional distribution of the k secret locations, and XOR of the values in these locations is very close to uniform. Following the intuition of [28], this implies both that 1) If the adversary has r bits of output and somehow knows the user's secret data, then the secret key is still close to independent of this information, and thus remains secure; 2) If in addition to the r bits of output, the adversary somehow, after the database has closed down, learns the user's secret key, then the user's secret data remains independent of this information, and thus has "everlasting" security [4]–namely, with high probability it is information theoretically impossible to learn Alice's bit. To obtain the proof of Theorem 2 for general t, we proceed by induction on t, leveraging the single time-step special case. The details of this proof overview are given below.

5.1 Large Sets Have Few Biases

In this section we show that for any sufficiently large section of the hypercube, relatively few sized k parities may have a significant bias. We begin by formalizing the notion of "bias" in a slightly more general setting.

Definition 2. *Given a function $f : \{-1,1\}^n \to [0, 2^{-n}]$ and a k-tuple of indices $h \subset \{1, \dots, n\}$, the* bias *of f with respect to h is the average value of the degree k monomial induced by h on the conditional distribution induced by f. Formally,*

$$bias(h, f) = \frac{1}{|f|} \sum_{x \in \{-1,1\}^n} f(x) \prod_{i \in h} x_i,$$

where $|f| = \sum_{x \in \{-1,1\}^n} f(x)$.

The following proposition shows that no function $f : \{-1,1\}^n \to [0, 2^{-n}]$ can have a significant bias with respect to too many k-tuples.

Proposition 1. *Let S denote the set of all k-tuples of indices in $\{1, \dots, n\}$ (hence $|S| = \binom{n}{k}$). For an even integer k, given a function $f : \{-1,1\}^n \to [0, 2^{-n}]$, the sum over all $h \in S$ of the square of the bias of f with respect to h is bounded as:*

$$\sum_{h \in S} bias(h, f)^2 \leq \frac{4k^{k+3}(2 - \log |f|)^k}{(2e)^k},$$

where $|f| = \sum_{x \in \{-1,1\}^n} f(x)$.

Our proof will leverage the following hypercontractivity concentration inequality from [40] (see [35, 23] for details):

Theorem 3 (Thm 1.10 from [40])**.** *For any degree k polynomial $P(x) = P(x_1, \ldots, x_n)$, where the x_i are independently chosen to be ± 1,*

$$\Pr_{x \in \{-1,1\}^n} [|P(x) - \boldsymbol{E}[P(x)]| \geq \lambda] \leq e^2 \cdot e^{-\left(\frac{\lambda^2}{e^2\, \boldsymbol{Var}[P(x)]}\right)^{1/k}},$$

Additionally, we will leverage the following standard fact about the upper incomplete gamma function:

Fact 1. *Letting $\Gamma(s,\alpha) = \int_{t=\alpha}^{\infty} t^{s-1}e^{-t}dt$ denote the upper incomplete gamma function, for any positive integer s,*

$$\Gamma(s,\alpha) = (s-1)!e^{-\alpha}\sum_{i=0}^{s-1}\frac{\alpha^i}{i!}.$$

Proof of Proposition 1. Define $P(x) = \sum_{h \in S} bias(h,f) \prod_{i \in h} x_i$ to be the degree k polynomial with $|S| = \binom{n}{k}$ monomials, with coefficients equal to the biases of the corresponding monomials/sets. Let $s = \sum_{h \in S} bias(h,f)^2$ denote the quantity we are trying to bound, and note that

$$s|f| = \sum_{x \in \{-1,1\}^n} f(x) \cdot P(x).$$

To bound this sum, given the polynomial P, consider the function $f^* : \{-1,1\}^n \to [0, 2^{-n}]$ with $|f^*| = |f|$ that maximizes the above quantity. Such an f^* can be constructed by simply sorting the points of the hypercube $x_1, x_2, \ldots, x_{2^n}$ s.t. $P(x_i) \geq P(x_{i+1})$, and then setting $\frac{1}{2^n} = f^*(x_1) = f^*(x_2) = \ldots = f^*(x_j)$ for $j = |f|/2^n$, and $f(x_i) = 0$ for all $i > |f|/2^n$. (For simplicity, assume $|f|$ is a multiple of $1/2^n$; if this is not the case, then we set $j = \lfloor |f|/2^n \rfloor$ and $f^*(x_{j+1}) = |f| - j/2^n$ and the argument proceeds analogously.) We now bound

$$\sum_{x \in \{-1,1\}^n} f(x)P(x) \leq \sum_{x \in \{-1,1\}^n} f^*(x)P(x) = \sum_{\lambda : \exists j \leq |f|/2^n \text{ with } P(x_j) = \lambda} \lambda \cdot \Pr_{x \leftarrow \{-1,1\}^n}[P(x) = \lambda],$$

where the probability is with respect to the uniform distribution over $x \in \{-1,1\}^n$. Given any differentiable function $g(\lambda)$ that satisfies $\Pr_{x \leftarrow \{-1,1\}^n}[P(x) \geq \lambda] \leq g(\lambda)$, we have

$$\sum_{\lambda : \exists j \leq |f|/2^n \text{ with } P(x_j) = \lambda} \lambda \cdot \Pr_{x \leftarrow \{-1,1\}^n}[P(x) = \lambda] \leq \int_{\lambda_0}^{\infty} \lambda \cdot \left| \frac{d}{d\lambda} g(\lambda) \right| d\lambda, \tag{1}$$

where λ_0 is chosen to be the largest value that λ can take, such that $g(\lambda_0) \geq |f|$. By Theorem 3, we may take

$$g(\lambda) = e^2 \cdot e^{-\left(\frac{\lambda^2}{e^2 \mathbf{Var}[P(x)]}\right)^{1/k}} = e^2 \cdot e^{-\left(\frac{\lambda^2}{e^2 \cdot s}\right)^{1/k}}.$$

Hence taking λ_0 so as to satisfy $|f| = e^2 \cdot e^{-\left(\frac{\lambda_0^2}{e^2 \cdot s}\right)^{1/k}}$, yields

$$\lambda_0 = (2 - \log|f|)^{k/2} \left(e^2 \cdot s\right)^{1/2}.$$

Plugging this into Equation 1 and noting that $\lambda|\frac{d}{d\lambda}g(\lambda)| = \frac{2}{k}\left(\frac{\lambda^2}{e^2\cdot s}\right)^{1/k} e^{-\left(\frac{\lambda^2}{e^2\cdot s}\right)^{1/k}}$, we get the following:

$$\begin{aligned}
\sum_x f(x)P(x) &\leq \frac{2e^2}{k}\int_{\lambda_0}^{\infty}\left(\frac{\lambda^2}{e^2\cdot s}\right)^{1/k} e^{-\left(\frac{\lambda^2}{e^2\cdot s}\right)^{1/k}} d\lambda \qquad \text{making the substitution } u = \left(\frac{\lambda^2}{e^2\cdot s}\right)^{1/k} \\
&= e^2\left(e^2\cdot s\right)^{1/2}\int_{u_0}^{\infty} u^{k/2}e^{-u}du \qquad \text{for } u_0 = \left(\frac{\lambda_0^2}{e^2\cdot s}\right)^{1/k} = 2-\log|f| \\
&= e^3\sqrt{s}\Gamma(k/2+1,u_0) \\
&= e^3\sqrt{s}(k/2)!e^{-u_0}\sum_{i=0}^{k/2}\frac{u_0^i}{i!} \\
&\leq e\sqrt{s}|f|(k/2)!(k/4)(2-\log|f|)^{k/2}.
\end{aligned}$$

The above establishes that $s|f| \leq e\sqrt{s}|f|(k/2)!(k/4)(2-\log|f|)^{k/2} \leq 2\sqrt{s}|f|(2e)^{-k/2}k^{k/2+3/2}(2-\log|f|)^{k/2}$, which implies that $s \leq 4(2e)^{-k}k^{k+3}(2-\log|f|)^k$, as desired. □

5.2 Completing the Proof

Equipped with Proposition 1, we now analyze the overall behavior of our secure database. We begin by proving that the security holds for a single re-randomization of the database, and then leverage that result via a basic induction argument to show that the security guarantees degrade linearly with the number of re-randomizations. The argument of this section closely follow the proof approach of [28].

We begin by considering an adversary that, given the n bits contained in the database, conducts an arbitrary computation to produce an output OUT that is r bits long, and show that, over the random choice of the k locations $h_1,\ldots,h_k \in [n]$ and the random choice of the database $dat \in \{-1,1\}^n$, even given OUT, the joint distribution of 1) the k locations $h_1,\ldots,h_k$ and 2) the parity of these k locations, is very close to being jointly uniform and independent.

Using the notation $\langle OUT_{dat}, h_1\ldots h_k, dat_{h_1}\oplus\cdots\oplus dat_{h_k}\rangle$ to represent the joint distribution of these three random variables, and letting U^h and $U^{\pm 1}$ denote the uniform distribution over the set $S = \{h_1,\ldots,h_k\} \subset [n]^k$ and the uniform distribution on ± 1, respectively, we have the following immediate corollary of Proposition 1, which shows the joint distributions $\langle OUT_{dat}, h_1\ldots h_k, dat_{h_1}\oplus\cdots\oplus dat_{h_k}\rangle$ and $\langle OUT_{dat}, U^h, U^{\pm 1}\rangle$ are exponentially close. This implies that, even with the hints provided by r bits of output OUT, 1) knowing the user's secret data $dat_{h_1}\oplus\cdots\oplus dat_{h_k}$ gives exponentially little information about the secret key $h_1\ldots h_k$ implying that the key can be securely reused an exponential number of times; and 2) if after the database closes, the secret key $h_1\ldots h_k$ is revealed, everlasting security still holds and the adversary has exponentially little information about the user's secret data $dat_{h_1}\oplus\cdots\oplus dat_{h_k}$, which implies the main results of this paper.

Lemma 1. *The statistical distance between the distributions* $\langle OUT_{dat}, h_1\ldots h_k, dat_{h_1}\oplus\cdots\oplus dat_{h_k}\rangle$ *and* $\langle OUT_{dat}, U^h, U^{\pm 1}\rangle$ *induced by randomly drawing* $dat \leftarrow \{-1,1\}^n$ *is at most* $\left(\frac{r}{n}\right)^{k/2}\cdot\frac{2k^{k/2+3/2}}{(2e)^{k/2}}$.

Proof. For a fixed r-bit string OUT_{dat}, consider the function $f_{OUT}: \{-1,1\}^n \to [0,2^{-n}]$ that on each string $x \in \{-1,1\}^n$ takes value equal to the joint probability that x is the chosen n-bit string *and* that the r bit output string equals OUT_{dat}. Proposition 1 yields that,

$$\sum_{h\in S} bias(h, f_{OUT})^2 \leq c_k(2-\log|f_{OUT}|)^k,$$

where $c_k = \frac{4k^{k+3}}{(2e)^k}$, and $|f_{OUT}| = \sum_{x \in \{-1,1\}^n} f_{OUT}(x)$.

Combining this result with the Cauchy-Schwarz inequality relating the sum of the elements of a vector to the sum of the squares of its elements, we have

$$\sum_{h \in S} bias(h, f_{OUT}) \leq \sqrt{\binom{n}{k} \cdot c_k (2 - \log |f_{OUT}|)^k}.$$

We observe that $|f_{OUT}|$, by definition, equals the probability that the particular value of OUT is chosen from among all r-bit strings; further, for this fixed OUT, the statistical distance between the joint distribution $\langle h_1 \dots h_k, dat_{h_1} \oplus \dots \oplus dat_{h_k} \rangle$ and the corresponding uniform distribution $\langle U^h, U^{\pm 1} \rangle$ equals $\binom{n}{k}^{-1} \cdot \sum_{h \in S} bias(h, f_{OUT})$.

Thus the desired statistical distance between the distributions $\langle OUT, h_1 \dots h_k, dat_{h_1} \oplus \dots \oplus dat_{h_k} \rangle$ and $\langle OUT, U^h, U^{\pm 1} \rangle$ is bounded by

$$\mathop{\mathbb{E}}_{OUT} \left[\sqrt{\binom{n}{k}^{-1} \cdot c_k (2 - \log |f_{OUT}|)^k} \right] = \sum_{OUT} |f_{OUT}| \cdot \sqrt{\binom{n}{k}^{-1} \cdot c_k (2 - \log |f_{OUT}|)^k},$$

subject to the constraint that $\sum_{OUT} |f_{OUT}| = 1$. Since $x(2 - \log x)^{k/2}$ is a concave function of x for $x \in [0, 1]$, the statistical distance is thus maximized when for each of the 2^r possible outputs OUT, the probabilities are all equal: $|f_{OUT}| = 2^{-r}$. Plugging this in to the above equation gives the desired bound on the statistical distance:

$$\begin{aligned} &\left| \langle OUT_{dat}, h_1 \dots h_k, dat_{h_1} \oplus \dots \oplus dat_{h_k} \rangle - \langle OUT_{dat}, U^h, U^{\pm 1} \rangle \right| \\ &\qquad \leq \sqrt{\binom{n}{k}^{-1} \cdot c_k (2 - \log 2^{-r})^k} = \left(\frac{r}{n} \right)^{k/2} \cdot \frac{2k^{k/2+3/2}}{(2e)^{k/2}}. \end{aligned}$$

□

We now complete the proof of our main security guarantee, Theorem 2, which we restate below in the above terminology. We use the notation $[t]$ for an integer t to denote the set $\{1, \dots, t\}$. We show that an adversary repeatedly hijacking the database essentially learns nothing beyond what the adversary could have learned by staying home and simulating the whole process. This guarantee holds even if the adversary finds out about all of Alice's previously stored bits, or, more generally, receives arbitrary "hints" from an outside source about Alice's past, present, and future bits. We proceed to show the information theoretic security of our database scheme by showing that for any adversary extracting information from the database, there is an analogous *simulator* that the adversary could run without any access to the database, whose results are identical with all but negligible probability. Such simulator constructions were originally developed and employed in the context of semantic security [20].

Theorem 2 (SIBA-security, restated)**.** *For any adversary, there is an efficient simulator S such that for any sequence of bits b_i to be stored at a succession of rerandomization times in the database, and any sequence of (possibly probabilistic) "hints" H_i that the adversary receives about the (previous, current, or future) bits in the sequence, then, averaged over the all $\binom{n}{k}$ secret k-tuples of locations $h_{[k]}$, the statistical distance between the distribution of the view of the adversary after running on the database for t rounds, receiving hint H_i after each round i versus the view of the simulator who is given hints $H_{[t]}$ but* **never** *interacts with the database, is less than $2t \cdot \epsilon_{r,n,k}$, where $\epsilon_{r,n,k} = \left(\frac{r}{n}\right)^{k/2} \cdot \sqrt{\frac{4k^{k+3}}{(2e)^k}}$ is the bound given in Lemma 1 for a single re-randomization.*

This theorem has the following immediate interpretations:

1. If at the end of t database rerandomizations an adversary is told Alice's bits $b_1, \ldots, b_t$, then it still cannot guess Alice's secret indices correctly with probability any better than $2t\epsilon_{r,n,k}$ more than random guessing.

2. If the database represents a uniformly random bit $b_t \in \{-1, 1\}$ during the tth re-randomization, then even if an adversary is told (at the very beginning) the $t-1$ bits, $b_1, \ldots, b_{t-1}$, that Alice is storing during the first $t-1$ database rerandomizations, and even if, subsequent to the $t+1$st rerandomization, the adversary is told Alice's secret set of indices, then the adversary can guess b_t correctly with probability at most $2t\epsilon_{r,n,k}$ better than random guessing. This is the "everlasting security" property.

Proof. We prove the theorem by induction on the number of rerandomizations t, where the $t = 0$ case corresponds to 0 rounds of the database, where the theorem trivially holds since neither the real nor simulated adversary has any information.

Assume, by the induction hypothesis, that there is an efficient simulator S that on input $H_{[t-1]}$ can probabilistically construct a sequence of outputs $OUT'_{[t-1]}$ that is (averaged over all $\binom{n}{k}$ choices of secret bit locations $h_{[k]}$) within statistical distance $2(t-1)\epsilon_{r,n,k}$ of the distribution of outputs $OUT_{[t-1]}$ produced by an adversary running for t rerandomizations on the actual database that encodes Alice's secret bits $b_{[t]}$ in secret locations $h_{[k]}$. We couple the random variables $OUT_{[t-1]}$ and $OUT'_{[t-1]}$ together so that they differ with probability $\leq 2(t-1)\epsilon_{r,n,k}$.

When the adversary is running on the database during the tth rerandomization, it calculates the tth output via some function $OUT_t = f(dat_t, OUT_{[t-1]}, H_t)$, in terms of the current database (which was randomly drawn so as to encode Alice's bit b_t as the XOR of locations $h_{[k]}$), the previous outputs, and whatever "hint" H_t it receives about Alice's bits. We change the distribution of OUT_t with probability $\leq 2(t-1)\epsilon_{r,n,k}$ if we modify it to a "primed" version $OUT'_t = f(dat_t, OUT'_{[t-1]}, H_t)$.

Since $OUT'_{[t-1]}$ is constructed by the simulator, it is independent of the locations $h_{[k]}$, though possibly dependent on Alice's current secret bit b_t (through hints the adversary received). Thus the output $OUT'_t = f(dat_t, OUT'_{[t-1]}, H_t)$ is a function of dat_t, independent of the locations $h_{[k]}$, possibly dependent on bit b_t (and also possibly dependent on previous bits $b_1, \ldots, b_{t-1}$ and future bits $b_{t+1}, \ldots$, though these do not matter here); we thus denote $OUT'_t = f_{b_t}(dat_t)$, where the function f_{b_t} is possibly stochastic. We thus apply Lemma 1 to both $f_{b_t=-1}$ and $f_{b_t=1}$: we interpret here interpret Lemma 1 as saying that for any function f that outputs r bits, the average over all choices of secret locations $h_{[k]}$ and both choices of the bit b_t of the statistical distance between the output of f applied to a database generated from $h_{[k]}$ and b_t versus the output of f when applied to a uniformly random string $dat \leftarrow \{-1, 1\}^n$ is at most $\epsilon_{r,n,k}$.

Since this bound of $\epsilon_{r,n,k}$ is averaged over both choices of the bit b_t, we bound the statistical distance for *either* choice by twice this, $2\epsilon_{r,n,k}$. Thus, for both $b_t = -1$ and $b_t = 1$ we have that, averaged over all choices of secret locations $h_{[k]}$, the statistical distance between f_{b_t} when evaluated on a database generated from the secrets $h_{[k]}$ and b_t versus when f_{b_t} is evaluated on a uniformly random string $dat \leftarrow \{-1, 1\}^n$ is at most $2\epsilon_{r,n,k}$

Thus, our simulator, after having already simulated $OUT'_{[t-1]}$ (by the induction hypothesis), next simply draws a random string $dat_t \leftarrow \{-1, 1\}^n$ and lets $OUT'_t = f(dat_t, OUT'_{[t-1]}, H_t)$. The coupling argument shows that the first $t-1$ outputs are accurately simulated except with probability $\leq 2(t-1)\epsilon_{r,n,k}$, and provided the first $t-1$ outputs are accurately simulated, the previous paragraph shows that the tth output has the desired distribution, up to statistical distance error $\leq 2\epsilon_{r,n,k}$ (in both the case $b_t = -1$ and the case $b_t = 1$); summing these bounds yields the induction: that our simulator accurately emulates the first t outputs up to statistical distance error $\leq 2t\epsilon_{r,n,k}$, as desired. □

References

[1] Joël Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 113–134. Springer, 2010.

[2] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In *Advances in Cryptology-CRYPTO 2009*, pages 36–54. Springer, 2009.

[3] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Survey: Leakage resilience and the bounded retrieval model. In *International Conference on Information Theoretic Security*, pages 1–18. Springer, 2009.

[4] Yonatan Aumann, Yan Zong Ding, and Michael O Rabin. Everlasting security in the bounded storage model. *Information Theory, IEEE Transactions on*, 48(6):1668–1680, 2002.

[5] M.F. Balcan, A. Blum, S. Fine, and Y. Mansour. Distributed learning, communication complexity and privacy. In *Conference on Learning Theory (COLT)*, 2012.

[6] Paul Beame, Shayan Oveis Gharan, and Xin Yang. Time-space tradeoffs for learning from small test spaces: Learning low degree polynomial functions. *arXiv preprint arXiv:1708.02640*, 2017.

[7] Shai Ben-David and Eli Dichterman. Learning with restricted focus of attention. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 287–296. ACM, 1993.

[8] M. Braverman, A. Garg, T. Ma, H.L. Nguyen, and D.P. Woodruff. Communication lower bounds for statistical estimation problems via a distributed data processing inequality. *arXiv preprint arXiv:1506.07216*, 2015.

[9] Giovanni Di Crescenzo, Richard Lipton, and Shabsi Walfish. Perfectly secure password protocols in the bounded retrieval model. In *Theory of Cryptography Conference*, pages 225–244. Springer, 2006.

[10] Yan Zong Ding and Michael O Rabin. Hyper-encryption and everlasting security. In *STACS 2002*, pages 1–26. Springer, 2002.

[11] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 511–520. IEEE, 2010.

[12] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 613–631. Springer, 2010.

[13] Yevgeniy Dodis, Allison Lewko, Brent Waters, and Daniel Wichs. Storing secrets on continually leaky devices. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 688–697. IEEE, 2011.

[14] J. Duchi, M. Jordan, and M. Wainwright. Local privacy and statistical minimax rates. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

[15] Stefan Dziembowski. Intrusion-resilience via the bounded-storage model. In *Theory of Cryptography Conference*, pages 207–224. Springer, 2006.

[16] Stefan Dziembowski. On forward-secure storage. In *Annual International Cryptology Conference*, pages 251–270. Springer, 2006.

[17] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 293–302. IEEE, 2008.

[18] A. Garg, T. Ma, and H. Nguyen. On communication cost of distributed statistical estimation and dimensionality. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[19] Sumegha Garg, Ran Raz, and Avishay Tal. Extractor-based time-space lower bounds for learning. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 990–1002. ACM, 2018.

[20] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM.

[21] Shafi Goldwasser and Guy N Rothblum. Securing computation against continuous leakage. In *Annual Cryptology Conference*, pages 59–79. Springer, 2010.

[22] Amir Herzberg, Stanisław Jarecki, Hugo Krawczyk, and Moti Yung. Proactive secret sharing or: How to cope with perpetual leakage. In *Annual International Cryptology Conference*, pages 339–352. Springer, 1995.

[23] Svante Janson. *Gaussian Hilbert Spaces*. Cambridge University Press, 1997. Cambridge Books Online.

[24] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998.

[25] Gillat Kol, Ran Raz, and Avishay Tal. Time-space hardness of learning sparse parities. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1067–1080. ACM, 2017.

[26] Allison Lewko, Mark Lewko, and Brent Waters. How to leak on key updates. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 725–734. ACM, 2011.

[27] Helger Lipmaa. An oblivious transfer protocol with log-squared communication. In *Information Security*, pages 314–328. Springer, 2005.

[28] Chi-Jen Lu. Hyper-encryption against space-bounded adversaries from on-line strong extractors. In *Advances in Cryptology (CRYPTO2002)*, pages 257–271. Springer, 2002.

[29] Ueli Maurer. Information-theoretic cryptography. In *Advances in Cryptology (CRYPTO99)*, pages 47–65. Springer, 1999.

[30] Ueli M Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[31] Silvio Micali and Leonid Reyzin. Physically observable cryptography. In *Theory of Cryptography Conference*, pages 278–296. Springer, 2004.

[32] Frank Miller. *Telegraphic code to insure privacy and secrecy in the transmission of telegrams.* CM Cornwell, 1882.

[33] Dana Moshkovitz and Michal Moshkovitz. Mixing implies lower bounds for space bounded learning. In *Conference on Learning Theory*, pages 1516–1566, 2017.

[34] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 448–457. Society for Industrial and Applied Mathematics, 2001.

[35] Ryan O'Donnell. Analysis of boolean functions, lecture notes (lecture 16). *http://www.cs.cmu.edu/ odonnell/boolean-analysis/.*

[36] Krzysztof Pietrzak. A leakage-resilient mode of operation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 462–482. Springer, 2009.

[37] Michael O Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptology ePrint Archive*, 2005:187, 2005.

[38] Ran Raz. A time-space lower bound for a large class of learning problems. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 732–742. IEEE, 2017.

[39] Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. *J. ACM*, 66(1):3:1–3:18, December 2018.

[40] Warren Schudy and Maxim Sviridenko. Concentration and moment inequalities for polynomials of independent random variables. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 437–446. SIAM, 2012.

[41] Ohad Shamir. Fundamental limits of online and distributed algorithms for statistical learning and estimation. In *Advances in Neural Information Processing Systems*, pages 163–171, 2014.

[42] Claude E Shannon. Communication theory of secrecy systems*. *Bell system technical journal*, 28(4):656–715, 1949.

[43] Vatsal Sharan, Aaron Sidford, and Gregory Valiant. Memory-sample tradeoffs for linear regression with small error. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 890–901, 2019.

[44] Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In *Conference on Learning Theory (COLT)*, 2016.

[45] Salil P Vadhan. On constructing locally computable extractors and cryptosystems in the bounded storage model. In *Annual International Cryptology Conference*, pages 61–77. Springer, 2003.

[46] Gilbert S Vernam. Cipher printing telegraph systems: For secret wire and radio telegraphic communications. *AIEE, Journal of the*, 45(2):109–115, 1926.

[47] Y. Zhang, J. Duchi, M. Jordan, and M. Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.