Delegating RAM Computations with Adaptive Soundness and Privacy[⋆]

Prabhanjan Ananth[1], Yu-Chi Chen[2], Kai-Min Chung[2], Huijia Lin[3], and Wei-Kai Lin[4]

[1] University of California Los Angeles and Center for Encrypted Functionalities. prabhanjan@cs.ucla.edu.
[2] Academia Sinica, Taiwan. wycchen,kmchung@iis.sinica.edu.tw.
[3] University of California Santa Barbara. rachel.lin@cs.ucsb.edu.
[4] Cornell University. wklin@cs.cornell.edu.

Abstract. We consider the problem of delegating RAM computations over persistent databases. A user wishes to delegate a sequence of computations over a database to a server, where each computation may read and modify the database and the modifications persist between computations. Delegating RAM computations is important as it has the distinct feature that the run-time of computations maybe *sub-linear* in the size of the database.

We present the first RAM delegation scheme that provide both soundness and privacy guarantees in the *adaptive* setting, where the sequence of delegated RAM programs are chosen adaptively, depending potentially on the encodings of the database and previously chosen programs. Prior works either achieved only adaptive soundness without privacy [Kalai and Paneth, ePrint'15], or only security in the selective setting where all RAM programs are chosen statically [Chen et al. ITCS'16, Canetti and Holmgren ITCS'16].

Our scheme assumes the existence of indistinguishability obfuscation ($i\mathcal{O}$) for circuits and the decisional Diffie-Hellman (DDH) assumption. However, our techniques are quite general and in particular, might be applicable even in settings where iO is not used. We provide a *"security lifting technique"* that "lifts" any proof of selective security satisfying certain special properties into a proof of adaptive security, for arbitrary cryptographic schemes. We then apply this technique to the delegation scheme of Chen et al. and its selective security proof, obtaining that their scheme is essentially already adaptively secure. Because of the general approach, we can also easily extend to delegating parallel RAM (PRAM) computations. We believe that the security lifting technique can potentially find other applications and is of independent interest.

[⋆] This paper was presented jointly with "Adaptive Succinct Garbled RAM, or How To Delegate Your Database" by Ran Canetti, Yilei Chen, Justin Holmgren, and Mariana Raykova. The full version of this paper is available on ePrint [2]. Information about the grants supporting the authors can be found in "Acknowledgements" section.

1 Introduction

In the era of cloud computing, it is of growing popularity for users to outsource both their databases and computations to the cloud. When the databases are large, it is important that the delegated computations are modeled as RAM programs for efficiency, *as computations maybe sub-linear*, and that the state of a database is kept persistently across multiple (sequential) computations to support continuous updates to the database. In such a paradigm, it is imperative to address two security concerns: *Soundness* (a.k.a., integrity) – ensuring that the cloud performs the computations correctly, and *Privacy* – information of users' private databases and programs is hidden from the cloud. In this work, we design *RAM delegation schemes* with both soundness and privacy.

Private RAM Delegation. Consider the following setting. Initially, to outsource her database DB, a user encodes the database using a secret key sk, and sends the encoding $\hat{DB}$ to the cloud. Later, whenever the user wishes to delegate a computation over the database, represented as a RAM program M, it encodes M using sk, producing an encoded program $\hat{M}$. Given $\hat{DB}$ and $\hat{M}$, the cloud runs an evaluation algorithm to obtain an encoded output $\hat{y}$, on the way updating the encoded database; for the user to verify the correctness of the output, the server additionally generates a proof π. Finally, upon receiving the tuple $(\hat{y}, \pi)$, the user verifies the proof and recovers the output y in the clear. The user can continue to delegate multiple computations.
In order to leverage the efficiency of RAM computations, it is important that RAM delegation schemes are *efficient*: The user runs in time only proportional to the size of the database, or to each program, while the cloud runs in time proportional to the run-time of each computation.

Adaptive v.s. Selective Security. Two "levels" of security exist for delegation schemes: The, *weaker*, selective security provides guarantees only in the restricted setting where all delegated RAM programs and database are chosen statically, whereas, the, *stronger*, adaptive security allows these RAM programs to be chosen adaptively, each (potentially) depending on the encodings of the database and previously chosen programs. Clearly, adaptive security is more natural and desirable in the context of cloud computing, especially for these applications where a large database is processed and outsourced once and many computations over the database are delegated over time.

We present an adaptively secure RAM delegation scheme.

Theorem 1 (Informal Main Theorem). *Assuming DDH and* $\mathsf{i}\mathcal{O}$ *for circuits, there is an efficient RAM delegation scheme, with adaptive privacy and adaptive soundness.*

Our result closes the gaps left open by previous two lines of research on RAM delegation. In one line, Chen et al. [20] and Canetti and Holmgren [16] constructed the first RAM delegation schemes that achieve *selective privacy* and *selective soundness*, assuming $\mathsf{i}\mathcal{O}$ and one-way functions; their works, however, left open security in the adaptive setting. In another line, Kalai and Paneth [35], building upon the seminal result of [36], constructed a RAM delegation scheme with *adaptive soundness*, based on super-polynomial hardness of the LWE assumption, which, however, does not provide privacy at all.[1] Our RAM delegation scheme improves upon previous works — it simultaneously achieves adaptive soundness and privacy. Concurrent to our work, Canetti, Chen, Holmgren, and Raykova [15] also constructed such a RAM delegation scheme. Our construction and theirs are the first to achieve these properties.

1.1 Our Contributions in More Detail

Our RAM delegation scheme achieves the privacy guarantee that the encodings of a database and many RAM programs, chosen adaptively by a malicious server (i.e., the cloud), reveals nothing more than the outputs of the computations. This is captured via the simulation paradigm, where the encodings can be simulated by a simulator that receives only the outputs. On the other hand, soundness guarantees that no malicious server can convince an honest client (i.e., the user) to accept a wrong output of any delegated computation, even if the database and programs are chosen adaptively by the malicious server.

Efficiency. Our adaptively secure RAM delegation scheme achieves the same level of efficiency as previous selectively secure schemes [16, 20]. More specifically,

- Client delegation efficiency: To outsource a database DB of size n, the client encodes the database in time linear in the database size, $n\,\mathrm{poly}(\lambda)$ (where λ is the security parameter), and the server merely stores the encoded database. To delegate the computation of a RAM program M, with l-bit outputs and time and space complexity T and S, the client encodes the program in time linear in the output length and polynomial in the program description size $l \times \mathrm{poly}(|M|, \lambda)$, independent of the complexity of the RAM program.
- Server evaluation efficiency: The evaluation time and space complexity of the server, scales linearly with the complexity of the RAM programs, that is, $T\,\mathrm{poly}(\lambda)$ and $S\,\mathrm{poly}(\lambda)$ respectively.
- Client verification efficiency: Finally, the user verifies the proof from the server and recovers the output in time $l \times \mathrm{poly}(\lambda)$.

The above level of efficiency is comparable to that of an *insecure* scheme (where the user simply sends the database and programs in the clear, and does not verify the correctness of the server computation), up to a multiplicative $\mathrm{poly}(\lambda)$ overhead at the server, and a $\mathrm{poly}(|M|, \lambda)$ overhead at the user.[2] In particular, if the run-time of a delegated RAM

[1] Note that here, privacy cannot be achieved for free using Fully Homomorphic Encryption (FHE), as FHE does not directly support computation with RAM programs, unless they are first transformed into oblivious Turing machines or circuits.

[2] We believe that the polynomial dependency on the program description size can be further reduced to linear dependency, using techniques in the recent work of [5].

program is sub-linear $o(n)$, the server evaluation time is also sub-linear $o(n)\,\mathrm{poly}(\lambda)$, which is crucial for server efficiency.

Technical contributions: Though our RAM delegation scheme relies on the existence of $\mathsf{i}\mathcal{O}$, the techniques that we introduce in this work are quite general and in particular, might be applicable in settings where $\mathsf{i}\mathcal{O}$ is not used at all.

Our main theorem is established by showing that the selectively secure RAM delegation scheme of [20] (CCC+ scheme henceforth) is, in fact, also adaptively secure (up to some modifications). However, proving its adaptive security is challenging, especially considering the heavy machinery already in the selective security proof (inherited from the line of works on succinct randomized encoding of Turing machines and RAMs [10,17]). Ideally, we would like to have a proof of adaptive security that uses the selective security property in a black-box way. A recent elegant example is the work of [1] that constructed an adaptively secure functional encryption from any selectively secure functional encryption without any additional assumptions.[3] However, such cases are rare: In most cases, adaptive security is treated independently, achieved using completely new constructions and/or new proofs (see examples, the adaptively secure functional encryption scheme by Waters [44], the adaptively secure garbled circuits by [34], and many others). In the context of RAM delegation, coming up with a proof of adaptive security from scratch requires at least repeating or rephrasing the proof of selective security and adding more details (unless the techniques behind the entire line of research [16,20,37] can be significantly simplified).

Instead of taking this daunting path, we follow a more principled and general approach. We provide an abstract proof that "lifts" any selective security proof satisfying certain properties — called a "nice" proof — into an adaptive security proof, for arbitrary cryptographic schemes. With the abstract proof, the task of showing adaptive security boils down to a mechanic (though possibly tedious) check whether the original selective security proof is nice. We proceed to do so for the CCC+ scheme, and show that when the CCC+ scheme is plugged in with a special kind of positional accummulator [37], called *history-less accummulator*, all niceness properties are satisfied; then its adaptive security follows immediately. At a very high-level, history-less accummulators can statistically bind the value at a particular position q irrespect of the history of read/write accesses, whereas positional accumulators of [37] binds the value at q after a specific sequence of read/write accesses.

Highlights of techniques used in the abstract proof includes a stronger version of complexity leveraging—called small-loss complexity leveraging—that have much smaller security loss than classical complexity leveraging, when the security game and its selective security proof satisfy certain "niceness" properties, as well as a way to apply small-loss complexity leveraging locally inside an involved security proof. We provide an overview of our techniques in more detail in Section 2.

[3] More generally, they use a 1-query adaptively secure functional encryption suffices which can be constructed from one-way functions by [32].

Parallel RAM (PRAM) Delegation As a benefit of our general approach, we can easily handle delegation of PRAM computations as well. Roughly speaking, PRAM programs are RAM programs that additionally support parallel (random) accesses to the database. Chen et al. [20] presented a delegation scheme for PRAM computations, with selective soundness and privacy. By applying our general technique, we can also lift the selective security of their PRAM delegation scheme to adaptive security, obtaining an adaptively secure PRAM delegation scheme.

Theorem 2 (informal — PRAM Delegation Scheme). *Assuming DDH and the existence of* $\mathsf{i}\mathcal{O}$ *for circuits, there exists an efficient PRAM delegation scheme, with adaptive privacy and adaptive soundness.*

1.2 Applications

In the context of cloud computing and big data, designing ways for delegating computation privately and efficiently is important. Different cryptographic tools, such as Fully Homomorphic Encryption (FHE) and Functional Encryption (FE), provide different solutions. However, so far, none supports delegation of *sub-linear* computation (for example, binary search over a large ordered data set, and testing combinatorial properties, like k-connectivity and bipartited-ness, of a large graph in sub-linear time). It is known that FHE does not support RAM computation, for the evaluator cannot decrypt the locations in the memory to be accessed. FE schemes for Turing machines constructed in [7] cannot be extended to support RAM, as the evaluation complexity is at least linear in the size of the encrypted database. This is due to a refreshing mechanism crucially employed in their work that "refreshes" the entire encrypted database in each evaluation, in order to ensure privacy. To the best of our knowledge, RAM delegation schemes are the only solution that supports sub-linear computations.

Apart from the relevance of RAM delegation in practice, it has also been quite useful to obtain theoretical applications. Recently, RAM delegation was also used in the context of patchable obfuscation by [6]. In particular, they crucially required that the RAM delegation satisfies adaptive privacy and only our work (and concurrently [15]) achieves this property.

1.3 On the Existence of IO

Our RAM delegation scheme assumes the existence of IO for circuits. So far, in the literature, many candidate IO schemes have been proposed (e.g., [9,14,26]) building upon the so called graded encoding schemes [23–25,29]. While the security of these candidates have come under scrutiny in light of two recent attacks [22,42] on specific candidates, there are still several IO candidates on which the current cryptanalytic attacks don't apply. Moreover, current multilinear map attacks do not apply to IO schemes obtained after applying bootstrapping techniques to candidate IO schemes for NC^1 [8,10,18,26,33] or special subclass of constant degree comptuations [38], or functional encryption schemes for NC^1 [4,5,11] or NC^0 [39]. We refer the reader to [3] for an extensive discussion of the state-of-affairs of attacks.

1.4 Concurrent and Related Works

Concurrent and independent work: A concurrent and independent work achieving the same result of obtaining adaptively secure RAM delegation scheme is by Canetti et. al. [15]. Their scheme extends the selectively secure RAM delegation scheme of [16], and uses a new primitive called adaptive accumulators, which is interesting and potentially useful for other applications. They give a proof of adaptive security from scratch, extending the selective security proof of [16] in a non-black-box way. In contrast, our approach is semi-generic. We isolate our key ideas in an abstract proof framework, and then instantiate the existing selective security proof of [20] in this framework. The main difference from [20] is that we use historyless accumulators (instead of using positional accumulators). Our notion of historyless accumulators is seemingly different from adaptive accumulators; its not immediately clear how to get one from the other. One concrete benefit our approach has is that the usage of $\mathsf{i}\mathcal{O}$ is falsifiable, whereas in their construction of adaptive accumulators, $\mathsf{i}\mathcal{O}$ is used in a non-falsifiable way. More specifically, they rely on the $\mathsf{i}\mathcal{O}$-to-differing-input obfuscation transformation of [13], which makes use of $\mathsf{i}\mathcal{O}$ in a non-falsifiable way.

Previous works on non-succinct garbled RAM: The notion of (one-time, non-succinct) garbled RAM was introduced by the work of Lu and Ostrovsky [40], and since then, a sequence of works [28, 30] have led to a black-box construction based on one-way functions, due to Garg, Lu, and Ostrovsky [27]. A black-box construction for *parallel* garbled RAM was later proposed by Lu and Ostrovsky [41] following the works of [12, 19]. However, the garbled program size here is proportional to the worst-case time complexity of the RAM program, so this notion does not imply a RAM delegation scheme. The work of Gentry, Halevi, Raykova, and Wichs [31] showed how to make such garbled RAMs reusable based on various notions of obfuscations (with efficiency trade-offs), and constructed the first RAM delegation schemes in a (weaker) offline/online setting, where in the offline phase, the delegator still needs to run in time proportional to the worst case time complexity of the RAM program.

Previous works on succinct garbled RAM: Succinct garbled RAM was first studied by [10, 17], where in their solutions, the garbled program size depends on the space complexity of the RAM program, but does not depend on its time complexity. This implies delegation for space-bounded RAM computations. Finally, as mentioned, the works of [16, 20] (following [37], which gives a Turing machine delegation scheme) constructed fully succinct garbled RAM, and [20] additionally gives the first fully succinct garbled PRAM. However, their schemes only achieve selective security. Lifting to adaptive security while keeping succinctness is the contribution of this work.

1.5 Organization

We first give an overview of our approach in Section 2. In Section 3, we present our abstract proof framework. The formal definition of adaptive delegation for RAMs is then presented in Section 4. Instantiation of this definition using our abstract proof framework is presented in the full version.

2 Overview

We now provide an overview of our abstract proof for lifting "nice" selective security proofs into adaptive security proofs. To the best of our knowledge, so far, the only general method going from selective to adaptive security is *complexity leveraging*, which however has (1) exponential security loss and (2) cannot be applied in RAM delegation setting for two reasons: (i) this will restrict the number of programs an adversary can choose and, (ii) the security parameter has to be scaled proportional to the number of program queries. This means that all the parameters grow proportional to the number of program queries.

Small-loss complexity leveraging: Nevertheless, we overcome the first limitation by showing a stronger version of complexity leveraging that has much smaller security loss, when the original selectively secure scheme (including its security game and security reduction) satisfy certain properties—we refer to the properties as *niceness* properties and the technique as *small-loss complexity leveraging.*

Local application: Still, many selectively secure schemes may not be *nice*, in particular, the CCC+ scheme. We broaden the scope of application of small-loss complexity leveraging using another idea: Instead of applying small-loss complexity leveraging to the scheme directly, we dissect its proof of selective security, and apply it to "smaller units" in the proof. Most commonly, proofs involve hybrid arguments; now, if every pair of neighboring hybrids with indistinguishability is *nice*, small-loss complexity leveraging can be applied *locally* to lift the indistinguishability to be resilient to adaptive adversaries, which then "sum up" to the global adaptive security of the scheme.

We capture the niceness properties abstractly and prove the above two steps abstractly. Interestingly, a challenging point is finding the right "language" (i.e. formalization) for describing selective and adaptive security games in a general way; we solve this by introducing *generalized security games.* With this language, the abstract proof follows with *simplicity* (completely disentangled from the complexity of specific schemes and their proofs, such as, the CCC+ scheme).

2.1 Classical Complexity Leveraging

Complexity leveraging says if a selective security game is $\mathsf{negl}(\lambda)2^{-L}$-secure, where λ is the security parameter and $L = L(\lambda)$ is the length of the information that selective adversaries choose statically (mostly at the beginning of the game), then the corresponding adaptive security game is $\mathsf{negl}(\lambda)$-secure. For example, the selective security of a public key encryption (PKE) scheme considers adversaries that choose two challenge messages v_0, v_1 of length n statically, whereas adaptive adversaries may choose v_0, v_1 adaptively depending on the public key. (See Figure 1.) By complexity leveraging, any PKE that is $\mathsf{negl}(\lambda)2^{-2n}$-selectively secure is also adaptively secure.

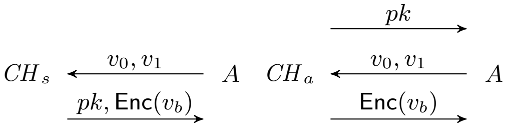


Fig. 1 Left: Selective security of PKE. Right: Adaptive security of PKE.

The idea of complexity leveraging is extremely simple. However, to extend it, we need a general way to formalize it. This turns out to be non-trivial, as the selective and adaptive security games are defined separately (e.g., the selective and adaptive security games of PKE have different challengers CH_s and CH_a), and vary case by case for different primitives (e.g., in the security games of RAM delegation, the adversaries choose multiple programs over time, as opposed to in one shot). To overcome this, we introduce generalize security games.

2.2 Generalized Security Games

Generalized security games, like classical games, are between a challenger CH and an adversary A, but are meant to separate the information A chooses statically from its interaction with CH. More specifically, we model A as a non-uniform Turing machine with an additional write-only *special output tape*, which can be written to only at the beginning of the execution (See Figure 1). The special output tape allows us to capture (fully) selective and (fully) adaptive adversaries naturally: The former write all messages to be sent in the interaction with CH on the tape (at the beginning of the execution), whereas the latter write arbitrary information. Now, selective and adaptive security are captured by running the same (generalized) security game, with different types of adversaries (e.g., see Figure 2 for the generalized security games of PKE).

Now, complexity leveraging can be proven abstractly: If there is an adaptive adversary A that wins against CH with advantage $\mathsf{negl}(\lambda)$, there is a selective adversary A' that wins with advantage $\mathsf{negl}(\lambda)/2^L$, as A' simply writes on its tape a random guess ρ of A's messages, which is correct with probability $1/2^L$.

With this formalization, we can further generalize the security games in two aspects. First, we consider the natural class of semi-selective adversaries that choose only partial information statically, as opposed to its entire transcript of messages (e.g., in the selective security game of functional encryption in [26] only the challenge messages are chosen selectively, whereas all functions are chosen adaptively). More precisely, an adversary is *F-semi-selective* if the initial choice ρ it writes to the special output tape is always consistent with its messages $m_1, \cdots, m_k$ w.r.t.

the output of F, $F(\rho) = F(m_1, \cdots, m_k)$. Clearly, complexity leveraging w.r.t. F-semi-selective adversaries incurs a 2^{L_F}-security loss, where $L_F = |F(\rho)|$.

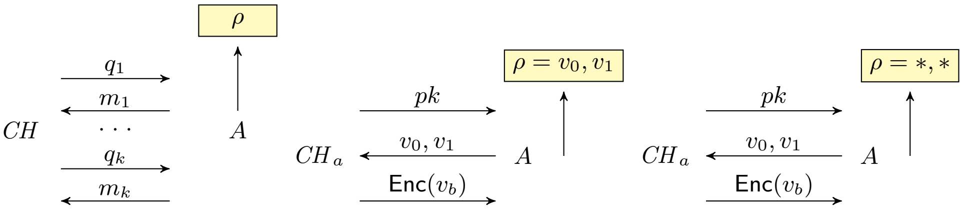
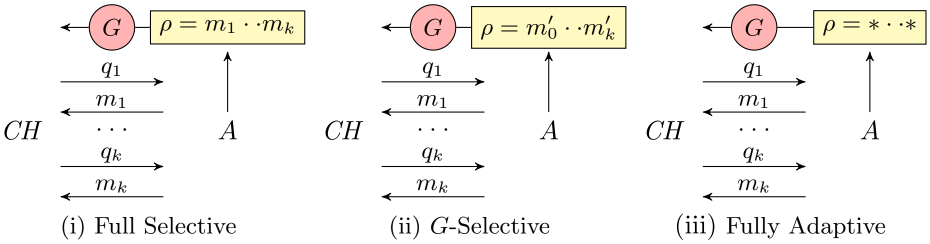


Fig. 2 Left: A generalized game. Middle and Right: Selective and adaptive security of PKE described using generalized games.

Second, we allow the challenger to depend on some partial information $G(\rho)$ of the adversary's initial choice ρ, by sending $G(\rho)$ to CH, after A writes to its special output tape (See Figure 3)—we say such a game is G-*dependent*. At a first glance, this extension seems strange; few primitives have security games of this form, and it is unnatural to think of running such a game with a fully adaptive adversary (who does not commit to $G(\rho)$ at all). However, such games are prevalent *inside* selective security proofs, which leverage the fact that adversaries are selective (e.g., the selective security proof of the functional encryption of [26] considers an intermediate hybrid where the challenger uses the challenge messages v_0, v_1 from the adversary to program the public key). Hence, this extension is essential to our eventual goal of applying small-loss complexity leveraging to neighboring hybrids, inside selective security proofs.



(i) Full Selective (ii) G-Selective (iii) Fully Adaptive

Fig. 3 Three levels of adaptivity. In (ii) G-selective means $G(m_1 \cdot \cdot m_k) = G(m'_1 \cdot \cdot m'_k)$.

2.3 Small-loss Complexity Leveraging

In a G-dependent generalized game CH, *ideally*, we want a statement that $\mathsf{negl}(\lambda)2^{-L_G}$-selective security (i.e., against (fully) selective adversaries) implies $\mathsf{negl}(\lambda)$-adaptively security (i.e., against (fully) adaptive adversaries). We stress that the security loss we aim for is 2^{L_G}, related to the length of the information $L_G = G(\rho)$ that the challenger depends on,[4] as opposed to 2^L as in classical complexity leveraging (where L is the total length of messages selective adversaries choose statically). When $L \gg L_G$, the saving in security loss is significant. However, this ideal statement is clearly false in general.

1. For one, consider the special case where G always outputs the empty string, the statement means $\mathsf{negl}(\lambda)$-selective security implies $\mathsf{negl}(\lambda)$-adaptive security. We cannot hope to improve complexity leveraging unconditionally.
2. For two, even if the game is 2^{-L}-selectively secure, complexity leveraging does not apply to *generalized* security games. To see this, recall that complexity leveraging turns an adaptive adversary A with advantage δ, into a selective one B with advantage $\delta/2^L$, who guesses A's messages at the beginning. It relies on the fact that the challenger is oblivious of B's guess ρ to argue that messages to and from A are information theoretically independent of ρ, and hence ρ matches A's messages with probability $1/2^L$ (see Figure 3 again). However, in generalized games, the challenger does depend on some partial information $G(\rho)$ of B's guess ρ, breaking this argument.

To circumvent the above issues, we strengthen the premise with two niceness properties (introduced shortly). Importantly, both niceness properties still only provide $\mathsf{negl}(\lambda)2^{-L_G}$-security guarantees, and hence the security loss remains 2^{L_G}.

Lemma 1 (Informal, Small Loss Complexity Leveraging). *Any G-dependent generalized security games with the following two properties for $\delta = \mathsf{negl}(\lambda)2^{-L_G}$ are adaptively secure.*

- *The game is δ-G-*hiding*.*
- *The game has a security reduction with δ-*statistical emulation property *to a δ-secure cryptographic assumption.*

We define δ-G-hiding and δ-statistical emulation properties shortly. We prove the above lemma in a modular way, by first showing the following semi-selective security property, and then adaptive security. In each step, we use one niceness property.

> **δ-semi-selective security:** We say that a G-dependent generalized security game *CH* is δ-semi-selective secure, if the winning advantage of any *G-semi-selective adversary* is bounded by $\delta = \mathsf{negl}(\lambda)2^{-L_G}$. Recall that such an adversary writes ρ to the special output tape at the beginning, and later choose adaptively any messages $m_1, \cdots, m_k$ consistent with $G(\rho)$, that is, $G(m_1, \cdots, m_k) = G(\rho)$ or $\perp$ (i.e., the output of G is undefined for $m_1, \cdots, m_k$).

[4] Because the challenger *CH* depends on L_G-bit of partial information $G(\rho)$ of the adversary's initial choice ρ, we do not expect to go below 2^{-L_G}-security loss unless requiring very strong properties to start with.

Step 1 – From selective to G-semi-selective security This step encounters the same problem as in the first issue above: We cannot expect to go from $\mathsf{negl}(\lambda)2^{-L_G}$-selective to $\mathsf{negl}(\lambda)2^{-L_G}$-semi-selective security unconditionally, since the latter is dealing with much more adaptive adversaries. Rather, we consider only cases where the selective security of the game with CH is proven using a *black-box straight-line* security reduction R to a game-based intractability assumption with challenger CH' (c.f. falsifiable assumption [43]). We identify the following sufficient conditions on R and CH' under which semi-selective security follows.

Recall that a reduction R simultaneously interacts with an adversary A (on the right), and leverages A's winning advantage to win against the challenger CH' (on the left). It is convenient to think of R and CH' as a compound machine $CH' \leftrightarrow R$ that interacts with A, and outputs what CH' outputs. Our condition requires that $CH' \leftrightarrow R$ emulates statistically every next message and output of CH. More precisely,

> **δ-statistical emulation property**: For every possible $G(\rho)$ and partial transcript $\tau = (q_1, m_1, \cdots, q_k, m_k)$ consistent with $G(\rho)$ (i.e., $G(m_1, \cdots, m_k) = G(\rho)$ or $\perp$), condition on them $(G(\rho), \tau)$ appearing in interactions with CH or $CH' \leftrightarrow R$, the distributions of the next message or output from CH or $CH' \leftrightarrow R$ are δ-statistically close.

We show that this condition implies that for any G-semi-selective adversary, its interactions with CH and $CH' \leftrightarrow R$ are $\text{poly}(\lambda)\delta$-statistically close (as the total number of messages is $\text{poly}(\lambda)$), as well as the output of CH and CH'. Hence, if the assumption CH' is $\mathsf{negl}(\lambda)2^{-L_G}$-secure against arbitrary adversaries, so is CH against G-semi-selective adversaries.[5]

Further discussion: We remark that the statistical emulation property is a strong condition that is sufficient but not necessary. A weaker requirement would be requiring the game to be G-semi-selective secure directly. However, we choose to formulate the statistical emulation property because it is a typical way how reductions are built, by emulating perfectly the messages and output of the challenger in the honest games. Furthermore, given R and CH', the statistical emulation property is easy to check, as from the description of R and CH', it is usually clear whether they emulate CH statistically close or not.

Step 2 – From G-semi-selective to adaptive security we would like to apply complexity leveraging to go from $\mathsf{negl}(\lambda)2^{-L_G}$-semi-selective security to adaptive security. However, we encounter the same problem as in the second issue above. To overcome it, we require the security game to be G-hiding, that is, the challenger's messages computationally hides $G(\rho)$.

> **δ-G-hiding:** For any ρ and ρ', interactions with CH after receiving $G(\rho)$ or $G(\rho')$ are indistinguishable to any polynomial-time adversaries, except from a δ distinguishing gap.

[5] Technically, we also require that CH and CH' have the same winning threshold, like both 1/2 or 0.

Let's see how complexity leveraging can be applied now. Consider again using an adaptive adversary A with advantage $1/\operatorname{poly}(\lambda)$ to build a semi-selective adversary B with advantage $1/\operatorname{poly}(\lambda)2^{L_G}$, who guesses A's choice of $G(m_1, \cdots, m_k)$ later. As mentioned before, since the challenger in the generalized game depends on B's guess τ, classical complexity leveraging argument does not apply. However, by the δ-G-hiding property, B's advantage differ by at most δ, when moving to a hybrid game where the challenger generates its messages using $G(\rho)$, where ρ is what A writes to its special output tape at the beginning, instead of τ. In this hybrid, the challenger is oblivious of B's guess τ, and hence the classical complexity leveraging argument applies, giving that B's advantage is at least $1/\operatorname{poly}(\lambda)2^{L_G}$. Thus by G-hiding, B's advantage in the original generalized game is at least $1/\operatorname{poly}(\lambda)2^{L_G} - \delta = 1/\operatorname{poly}(\lambda)2^{L_G}$. This gives a contradiction, and concludes the adaptive security of the game. Summarizing the above two steps, we obtain our informal lemma on small-loss complexity leveraging.

2.4 Local Application

In many cases, small-loss complexity leveraging may not directly apply, since either the security game is not G-hiding, or the selective security proof does not admit a reduction with the statistical emulation property. We can broaden the application of small-loss complexity leveraging by looking into the selective security proofs and apply small loss complexity leveraging on smaller "steps" inside the proof. For our purpose of getting adaptively secure RAM delegation, we focus on the following common proof paradigm for showing indistinguishability based security. But the same principle of local application could be applied to other types of proofs.

A common proof paradigm for showing the indistinguishability of two games Real_0 and Real_1 against selective adversaries is the following:

- First, construct a sequence of hybrid experiments $H_0, \cdots, H_\ell$, that starts from one real experiment (i.e., $H_0 = \mathit{Real}_0$), and gradually morphs through intermediate hybrids H_i's into the other (i.e., $H_\ell = \mathit{Real}_1$).
- Second, show that every pair of neighboring hybrids H_i, H_{i+1} is indistinguishable to selective adversaries.

Then, by standard hybrid arguments, the real games are selectively indistinguishable.

To lift such a selective security proof into an adaptive security proof, we first cast all real and hybrids games into our framework of generalized games, which can be run with both selective and adaptive adversaries. If we can obtain that neighboring hybrids games are also indistinguishable to adaptive adversaries, then the adaptive indistinguishability of the two real games follow simply from hybrid arguments. Towards this, we apply small-loss complexity leveraging on neighboring hybrids. More specifically, H_i and H_{i+1} are adaptively indistinguishable, if they satisfy the following properties:

- H_i and H_{i+1} are respectively G_i and G_{i+1}-dependent, as well as δ-$(G_i||G_{i+1})$-hiding, where $G_i||G_{i+1}$ outputs the concatenation of the outputs of G_i and G_{i+1} and $\delta = \mathsf{negl}(\lambda)2^{-L_{G_i}-L_{G_{i+1}}}$.
- The selective indistinguishability of H_i and H_{i+1} is shown via a reduction R to a δ-secure game-based assumption and the reduction has δ-statistical emulation property.

Thus, applying small-loss complexity leveraging on every neighboring hybrids, the maximum security loss is $2^{2L_{max}}$, where $L_{max} = \max(L_{G_i})$. Crucially, if every hybrid H_i have small L_{G_i}, the maximum security loss is small. In particular, we say that a selective security proof is "*nice*" if it falls into the above framework and all G_i's have only *logarithmic length* outputs — such "nice" proofs can be lifted to proofs of adaptive indistinguishability with only polynomial security loss. This is exactly the case for the CCC+ scheme, which we explain next.

2.5 The CCC+ Scheme and Its Nice Proof

CCC+ proposed a selectively secure RAM delegation scheme in the persistent database setting. We now show how CCC+ scheme can be used to instantiate the abstract framework discussed earlier in this Section. We only provide with relevant details of CCC+ and refer the reader to the full version for a thorough discussion.

There are two main components in CCC+. The first component is *storage* that maintains information about the database and the second component is the *machine* component that involves executing instructions of the delegated RAM. For every RAM delegated, there will be a separate *machine component*. Both the storage and the machine components are built on heavy machinery. We highlight below two important building blocks relevant to our discussion. Additional tools such as iterators and splittable signatures are also employed in their construction.

- *Positional Accumulators*: This primitive offers a mechanism of producing a short value, called *accumulator*, that commits to a large storage. Further, accumulators should also be updatable - if a small portion of storage changes then only a correspondingly small change is required to update the accumulator value. In the security proof, accumulators allow for programming the parameters with respect to a particular location in such a way that the accumulator uniquely determines the value at that location. However, such programming requires to know ahead of time all the changes the storage undergoes since its initialization. Henceforth, we refer to the hybrids to be in Enforce-mode when the accumulator parameters are programmed and the setting when it is not programmed to be Real-mode.
- *"Puncturable" Oblivious RAM*: Oblivious RAM (ORAM) is a randomized compiler that compiles any RAM program into one with a fixed distribution of random access pattern to hide its actual (logic) access pattern. CCC+ relies on stronger "puncturable" property of specific ORAM construction of [21], which roughly says the compiled access pattern of a particular logic memory access can be simulated if certain local ORAM randomness is information theoretically "punctured out," and this local randomness is determined at the time the

logic memory location is last accessed. Henceforth, we refer to the hybrids to be in Puncturing-mode when the ORAM randomness is punctured out.

We show that the security proof of CCC+ has a nice proof. We denote the set of hybrids in CCC+ to be $H_1, \ldots, H_\ell$. Correspondingly, we denote the reductions that argue indistinguishability of H_i and H_{i+1} to be R_i. We consider the following three cases depending on the type of neighboring hybrids H_i and H_{i+1}:

1. ORAM is in Puncturing-mode in one or both of the neighboring hybrids: In this case, the hybrid challenger needs to know which ORAM local randomness to puncture out to hide the logic memory access to location q at a particular time point t. As mentioned, this local randomness appears for the first time at the last time point t' that location q is accessed, possibly by a previous machine. As a result, in the proof, some machine components need to be programmed depending on the memory access of later machines. In this case, G_i or G_{i+1} need to contain information about q, t and t', which can be described in $\mathrm{poly}(\lambda)$ bits.
2. Positional Accumulator is in Enforce-mode in one or both of the neighboring hybrids: Here, the adversary is supposed to declare all its inputs in the beginning of experiment. The reason being that in the enforce-mode, the accumulator parameters need to be programmed. As remarked earlier, programming the parameters is possible only with the knowledge of the entire computation.
3. Remaining cases: In remaining cases, the indistinguishability of neighboring hybrids reduces to the security of other cryptographic primitives, such as, iterators, splittable signatures, indistinguishability obfuscation and others. We note that in these cases, we simply have $G_i = G_{i+1} = \mathsf{null}$, which outputs an empty string.

As seen from the above description, only the second case is problematic for us since the information to be declared by the adversary in the beginning of the experiment is too long. Hence, we need to think of alternate variants to positional accumulators where the enforce-mode can be implemented without the knowledge of the computation history.

History-less Accumulators. To this end, we introduce a primitive called *history-less accumulators*. As the name is suggestive, in this primitive, programming the parameters requires only the location being information-theoretically bound to be known ahead of time. And note that the location can be represented using only logarithmic bits and satisfies the size requirements. That is, the output length of G_i is now short. By plugging this into the CCC+ scheme, we obtain a "nice" security proof.

All that remains is to construct history-less accumulators. The construction of this primitive can be found in the full version.

3 Abstract Proof

In this section, we present our abstract proof that turns "nice" selective security proofs, to adaptive security proofs. As discussed in the introduction, we use generalized security experiments and games to describe our transformation. We present small-loss complexity leveraging in Section 3.3 and how to locally apply it in Section 3.4. In the latter, we focus our attention on proofs of indistinguishability against selective adversaries, as opposed to proofs of arbitrary security properties.

3.1 Cryptographic Experiments and Games

We recall standard cryptographic experiments and games between two parties, a challenger CH and an adversary A. The challenger defines the procedure and output of the experiment (or game), whereas the adversary can be any probabilistic interactive machine.

Definition 1 (Canonical Experiments). *A canonical experiment between two probabilistic interactive machines, the* challenger CH *and the* adversary A*, with security parameter* $\lambda \in \mathbb{N}$*, denoted as* $\mathsf{Exp}(\lambda, CH, A)$*, has the following form:*

- *CH and A receive common input 1^λ, and interact with each other.*
- *After the interaction, A writes an output γ on its output tape. In case A aborts before writing to its output tape, its output is set to $\perp$.*
- *CH additionally receives the output of A (receiving $\perp$ if A aborts), and outputs a bit b indicating accept or reject. (CH never aborts.)*

We say A wins *whenever CH outputs 1 in the above experiment.*
A canonical game (CH, τ) has additionally a threshold $\tau \in [0, 1)$. We say A has advantage γ if A wins with probability $\tau + \gamma$ in $\mathsf{Exp}(\lambda, CH, A)$*.*

For machine $\star \in \{CH, A\}$, we denote by $\mathsf{Out}_\star(\lambda, CH, A)$ and $\mathsf{View}_\star(\lambda, CH, A)$ the random variables describing the output and view of machine $\star$ in $\mathsf{Exp}(\lambda, CH, A)$.

Definition 2 (Cryptographic Experiments and Games). *A cryptographic experiment is defined by an ensemble of PPT challengers $\mathcal{CH} = \{CH_\lambda\}$. And a* cryptographic game $(\mathcal{CH}, \tau)$ *has additionally a threshold $\tau \in [0, 1)$. We say that a non-uniform adversary $\mathcal{A} = \{A_\lambda\}$ wins the cryptographic game with* advantage $\mathsf{Advt}(\star)$*, if for every $\lambda \in \mathbb{N}$, its advantage in* $\mathsf{Exp}(\lambda, CH_\lambda, A_\lambda)$ *is* $\tau + \mathsf{Advt}(\lambda)$*.*

Definition 3 (Intractability Assumptions). *An intractability assumption $(\mathcal{CH}, \tau)$ is the same as a cryptographic game, but with* potentially unbounded *challengers. It states that the advantage of every non-uniform* PPT *adversary $\mathcal{A}$ is negligible.*

3.2 Generalized Cryptographic Games

In the literature, experiments (or games) for selective security and adaptive security are often defined separately: In the former, the challenger requires the adversary to choose certain information at the beginning of the interaction, whereas in the latter, the challenger does not require such information.

We generalize standard cryptographic experiments so that the same experiment can work with both selective and adaptive adversaries. This is achieved by separating information necessary for the execution of the challenger and information an adversary chooses statically, which can be viewed as a property of the adversary. More specifically, we consider adversaries that have a *special output tape*, and write information α it chooses statically at the beginning of the execution on it; and only the necessary information specified by a function, $G(\alpha)$, is sent to the challenger. (See Figure 3.)

Definition 4 (Generalized Experiments). *A* generalized experiment *between a challenger CH and an adversary A* with respect to a function *G, with security parameter $\lambda \in \mathbb{N}$, denoted as $\mathsf{Exp}(\lambda, CH, G, A)$, has the following form:*

1. *The adversary A on input 1^λ writes on its special output tape string α at the beginning of its execution, called the initial choice of A, and then proceeds as a normal probabilistic interactive machine. (α is set to the empty string ε if A does not write on the special output tape at the beginning.)*
2. *Let $A[G]$ denote the adversary that on input 1^λ runs A with the same security parameter internally; upon A writing α on its special output tape, it sends out message $m_1 = G(\alpha)$, and later forwards messages A sends, $m_2, m_3, \cdots$*
3. *The generalized experiment proceeds as a standard experiment between CH and $A[G]$, $\mathsf{Exp}(\lambda, CH, A[G])$.*

We say that A wins *whenever CH outputs* 1.
Furthermore, for any function $F : \{0,1\}^* \to \{0,1\}^*$, we say that A is F-selective in $\mathsf{Exp}(\lambda, CH, G, A)$, if it holds with probability 1 that either A aborts or its initial choice α and messages it sends satisfy that $F(\alpha) = F(m_2, m_3, \cdots)$. We say that A is adaptive, in the case that F is a constant function.

Similar to before, we denote by $\mathsf{Out}_\star(\lambda, CH, G, A)$ and $\mathsf{View}_\star(\lambda, CH, G, A)$ the random variables describing the output and view of machine $\star \in \{CH, A\}$ in $\mathsf{Exp}(\lambda, CH, G, A)$. In this work, we restrict our attention to all the functions G that are efficiently computable, as well as, *reversely computable*, meaning that given a value y in the domain of G, there is an efficient procedure that can output an input x such that $G(x) = y$.

Definition 5 (Generalized Cryptographic Experiments and $\mathcal{F}$-Selective Adversaries). *A generalized cryptographic experiment is a tuple $(\mathcal{CH}, \mathcal{G})$, where $\mathcal{CH}$ is an ensemble of PPT challengers $\{CH_\lambda\}$ and $\mathcal{G}$ is an ensemble of efficiently computable functions $\{G_\lambda\}$. Furthermore, for any ensemble of functions $\mathcal{F} = \{F_\lambda\}$ mapping $\{0,1\}^*$ to $\{0,1\}^*$, we say that a non-uniform adversary $\mathcal{A}$ is $\mathcal{F}$-selective in cryptographic experiments $(\mathcal{CH}, \mathcal{G})$ if for every $\lambda \in \mathbb{N}$, A_λ is F_λ-selective in experiment $\mathsf{Exp}(\lambda, CH_\lambda, G_\lambda, A_\lambda)$.*

Similar to Definition 2, a generalized cryptographic experiment can be extended to a *generalized cryptographic game* $(\mathcal{CH}, \mathcal{G}, \tau)$ by adding an additional threshold $\tau \in [0, 1)$, where the advantage of any non-uniform probabilistic adversary $\mathcal{A}$ is defined identically as before.
We can now quantify the level of selective/adaptive security of a generalized cryptographic game.

Definition 6 ($\mathcal{F}$-Selective Security). *A generalized cryptographic game $(\mathcal{CH}, \mathcal{G}, \tau)$ is* $\mathcal{F}$-selective secure *if the advantage of every non-uniform* PPT *$\mathcal{F}$-selective adversary $\mathcal{A}$ is negligible.*

3.3 Small-loss Complexity Leveraging

In this section, we present our small-loss complexity leveraging technique to lift fully selective security to fully adaptive security for a generalized cryptographic game $\Pi = (\mathcal{CH}, \mathcal{G}, \tau)$, provided that the game and its (selective) security proof satisfies certain niceness properties. We will focus on the following class of *guessing* games, which captures indistinguishability security. We remark that our technique also applies to generalized cryptographic games with arbitrary threshold (See Remark 1).

Definition 7 (Guessing Games). *A generalized game (CH, G, τ) (for a security parameter λ) is a* guessing game *if it has the following structure.*

- *At beginning of the game, CH samples a uniform bit $b \leftarrow \{0, 1\}$.*
- *At the end of the game, the adversary guesses a bit $b' \in \{0, 1\}$, and he wins if $b = b'$.*
- *When the adversary aborts, his guess is a uniform bit $b' \leftarrow \{0, 1\}$.*
- *The threshold $\tau = 1/2$.*

The definition extends naturally to a sequence of games $\Pi = (\mathcal{CH}, \mathcal{G}, 1/2)$. Our technique consists of two modular steps: First reach $\mathcal{G}$-selective security, and then adaptive security, where the first step applies to any generalized cryptographic game.

Step 1: $\mathcal{G}$-Selective Security In general, a fully selectively secure Π may not be $\mathcal{F}$-selective secure for $\mathcal{F} \neq \mathcal{F}_{\text{id}}$, where $\mathcal{F}_{\text{id}}$ denotes the identity function. We restrict our attention to the following case: The security is proved by a straight-line black-box security reduction from Π to an intractability assumption $(\mathcal{CH}', \tau')$, where the reduction is an ensemble of PPT machines $\mathcal{R} = \{R_\lambda\}$ that interacts simultaneously with an adversary for Π and $\mathcal{CH}'$, the reduction is syntactically well-defined with respect to any class of $\mathcal{F}$-selective adversary. This, however, does not imply that R is a correct reduction to prove $\mathcal{F}$-selective security of Π. Here, we identify a sufficient condition on the "niceness" of reduction that implies $\mathcal{G}$-selective security of Π. We start by defining the syntax of a straight-line black-box security reduction.

Standard straight-line black-box security reduction from a cryptographic game to an intractability assumption is a PPT machine R that interacts simultaneously with an adversary and the challenger of the assumption. Since our generalized cryptographic games can be viewed as standard cryptographic games with adversaries of the form $\mathcal{A}[\mathcal{G}] = \{A_\lambda[G_\lambda]\}$, the standard notion of reductions extends naturally, by letting the reductions interact with adversaries of the form $\mathcal{A}[\mathcal{G}]$.

Definition 8 (Reductions). *A probabilistic interactive machine R is a (*straight-line black-box*) reduction from a generalized game (CH, G, τ) to a (canonical) game (CH', τ') for security parameter λ, if it has the following syntax:*

- *Syntax: On common input 1^λ, R interacts with CH' and an adversary $A[G]$ simultaneously in a straight-line—referred to as "left" and "right" interactions respectively. The left interaction proceeds identically to the experiment* $\mathsf{Exp}(\lambda, CH', R{\leftrightarrow}A[G])$*, and the right to experiment* $\mathsf{Exp}(\lambda, CH'{\leftrightarrow}R, A[G])$*.*

*A (*straight-line black-box*) reduction from an ensemble of generalized cryptographic game $(\mathcal{CH}, \mathcal{G}, \tau)$ to an intractability assumption $(\mathcal{CH}', \tau')$ is an ensemble of PPT reductions $\mathcal{R} = \{R_\lambda\}$ from game $(CH_\lambda, G_\lambda, \tau)$ to (CH'_λ, τ') (for security parameter λ).*

At a high-level, we say that a reduction is μ-nice, where μ is a function, if it satisfies the following syntactical property: R (together with the challenger CH of the assumption) generates messages and output that are statistically close to the messages and output of the challenger CH' of the game, *at every step*.

More precisely, let $\rho = (m_1, a_1, m_2, a_2, \cdots, m_t, a_t)$ denote a transcript of messages and outputs in the interaction between CH and an adversary (or in the interaction between $CH'{\leftrightarrow}R$ and an adversary) where $\boldsymbol{m} = m_1, m_2, \cdots, m_{t-1}$ and m_t correspond to the messages and output of the adversary ($m_t = \perp$ if the adversary aborts) and $\boldsymbol{a} = a_1, a_2, \cdots, a_{t-1}$ and a_t corresponds to the messages and output of CH (or $CH'{\leftrightarrow}R$). A transcript ρ possibly appears in an interaction with CH (or $CH'{\leftrightarrow}R$) if when receiving $\boldsymbol{m}$, CH (or $CH'{\leftrightarrow}R$) generates $\boldsymbol{a}$ with non-zero probability. The syntactical property requires that for every prefix of a transcript that possibly appear in both interaction with CH and interaction with $CH'{\leftrightarrow}R$, the distributions of the next message or output generated by CH and $CH'{\leftrightarrow}R$ are statistically close. In fact, for our purpose later, it suffices to consider the prefixes of transcripts that are *G-consistent*: A transcript ρ is G-consistent if $\boldsymbol{m}$ satisfies that either $m_t = \perp$ or $m_1 = G(m_2, m_3, \cdots, m_{t-1})$; in other words, ρ could be generated by a G-selective adversary.

Definition 9 (Nice Reductions). *We say that a reduction R from a generalized game (CH, G, τ) to a (canonical) game (CH', τ) (with the same threshold) for security parameter λ is μ-*nice*, if it satisfies the following property:*

- *$\mu(\lambda)$-statistical emulation for G-consistent transcripts: For every prefix $\rho = (m_1, a_1, m_2, a_2, \cdots, m_{\ell-1}, a_{\ell-1}, m_\ell)$ of a G-consistent transcript of messages that possibly appears in interaction with both CH and $CH'{\leftrightarrow}R$, the following two distributions are $\mu(\lambda)$-close:*

$$\mathit{\Delta}(\mathsf{D}_{CH'\leftrightarrow R}(\lambda, \rho),\ \mathsf{D}_{CH}(\lambda, \rho)) \leq \mu(\lambda)$$

where $\mathsf{D}_M(\lambda, \rho)$ for $M = CH'{\leftrightarrow}R$ or CH is the distribution of the next message or output *a_ℓ generated by $M(1^\lambda)$ after receiving messages $\boldsymbol{m}$ in ρ, and conditioned on $M(1^\lambda)$ having generated $\boldsymbol{a}$ in ρ.*

Moreover, we say that a reduction $\mathcal{R} = \{R_\lambda\}$ from a generalized cryptographic game $(\mathcal{CH}, \mathcal{G}, \tau)$ to a intractability assumption $(\mathcal{CH}', \tau)$ is nice *if there is a negligible function μ, such that, R_λ is $\mu(\lambda)$-nice for every λ.*

When a reduction is μ-nice with negligible μ, it is sufficient to imply $\mathcal{G}$-selective security of the corresponding generalized cryptographic game. We defer the proofs to the full version.

Lemma 2. *Suppose R is a μ-nice reduction from (CH, G, τ) to (CH', τ) for security parameter λ, and A is a deterministic G-semi-selective adversary that wins (CH, G, τ) with advantage $\gamma(\lambda)$, then $R \leftrightarrow A[G]$ is an adversary for (CH', τ) with advantage $\gamma(\lambda) - t(\lambda) \cdot \mu(\lambda)$, where $t(\lambda)$ is an upper bound on the run-time of R.*

By a standard argument, Lemma 2 implies the following asymptotic version theorem.

Theorem 3. *If there exists a nice reduction $\mathcal{R}$ from a generalized cryptographic game $(\mathcal{CH}, \mathcal{G}, \tau)$ to an intractability assumption $(\mathcal{CH}', \tau)$, then $(\mathcal{CH}, \mathcal{G}, \tau)$ is $\mathcal{G}$-selectively secure.*

Step 2: Fully Adaptive Security We now show how to move from $\mathcal{G}$-selective security to fully adaptive security for the class of guessing games with security loss $2^{L_G(\lambda)}$, where $L_G(\lambda)$ is the output length of $\mathcal{G}$, provided that the challenger's messages hide the information of $G(\alpha)$ computationally. We start with formalizing this hiding property.
Roughly speaking, the challenger CH of a generalized experiment (CH, G) is G-hiding, if for any α and α', interactions with CH receiving $G(\alpha)$ or $G(\alpha')$ at the beginning are indistinguishable. Denote by $CH(x)$ the challenger with x hardcoded as the first message.

Definition 10 (G-hiding). *We say that a generalized guessing game (CH, G, τ) is $\mu(\lambda)$-G-hiding for security parameter λ, if its challenger CH satisfies that for every α and α', and every non-uniform* PPT *adversary A,*

$$|\Pr[\mathsf{Out}_A(\lambda, CH(G(\alpha)), A) = 1] - \Pr[\mathsf{Out}_A(\lambda, CH(G(\alpha')), A) = 1]| \leq \mu(\lambda)$$

Moreover, we say that a generalized cryptographic guessing game $(\mathcal{CH}, \mathcal{G}, \tau)$ is $\mathcal{G}$-hiding, if there is a negligible function μ, such that, $(\mathcal{CH}_\lambda, G_\lambda, \tau(\lambda))$ is $\mu(\lambda)$-G_λ-hiding for every λ.

The following lemma says that if a generalized guessing game $(CH, G, 1/2)$ is G-selectively secure and G-hiding, then it is fully adaptively secure with 2^{L_G} security loss. Its formal proof is deferred to the full version.

Lemma 3. *Let $(CH, G, 1/2)$ be a generalized cryptographic guessing game for security parameter λ. If there exists a fully adaptive adversary A for $(CH, G, 1/2)$ with advantage $\gamma(\lambda)$ and $(CH, G, 1/2)$ is $\mu(\lambda)$-G-hiding with $\mu(\lambda) \leq \gamma/2^{L_G(\lambda)+1}$, then there exists a G-selective adversary A' for $(CH, G, 1/2)$ with advantage $\gamma(\lambda)/2^{L_G(\lambda)+1}$, where L_G is the output length of G.*

Therefore, for a generalized cryptographic guessing game $(\mathcal{CH}, \mathcal{G}, \tau)$, if $\mathcal{G}$ has logarithmic output length $L_G(\lambda) = O(\log \lambda)$ and the game is $\mathcal{G}$-hiding, then its $\mathcal{G}$-selective security implies fully adaptive security.

Theorem 4. *Let $(\mathcal{CH}, \mathcal{G}, \tau)$ be a $\mathcal{G}$-selectively secure generalized cryptographic guessing game. If $(\mathcal{CH}, \mathcal{G}, \tau)$ is $\mathcal{G}$-hiding and $L_G(\lambda) = O(\log \lambda)$, then $(\mathcal{CH}, \mathcal{G}, \tau)$ is fully adaptively secure.*

Remark 1. The above proof of small-loss complexity leveraging can be extended to a more general class of security games, beyond the guessing games. The challenger with an arbitrary threshold τ has the form that if the adversary aborts, the challenger toss a biased coin and outputs 1 with probability τ. The same argument above goes through for games with this class of challengers.

3.4 Nice Indistinguishability Proof

In this section, we characterize an abstract framework of proofs—called "nice" proofs—for showing the indistinguishability of two ensembles of (standard) cryptographic experiments. We focus on a common type of indistinguishability proof, which consists of a sequence of hybrid experiments and shows that neighboring hybrids are indistinguishable via a reduction to a intractability assumption. We formalize required nice properties of the hybrids and reductions such that a fully selective security proof can be lifted to prove fully adaptive security by local application of small-loss complexity leveraging technique to neighboring hybrids. We start by describing common indistinguishability proofs using the language of generalized experiments and games.

Consider two ensembles of standard cryptographic experiments $\mathcal{RL}_0$ and $\mathcal{RL}_1$. They are special cases of generalized cryptographic experiments with a function $G = \mathsf{null} : \{0,1\}^* \to \{\varepsilon\}$ that always outputs the empty string, that is, $(\mathcal{RL}_0, \mathsf{null})$ and $(\mathcal{RL}_1, \mathsf{null})$; we refer to them as the "real" experiments.

Consider a proof of indistinguishability of $(\mathcal{RL}_0, \mathsf{null})$ and $(\mathcal{RL}_1, \mathsf{null})$ against fully selective adversaries via a sequence of hybrid experiments. As discussed in the overview, the challenger of the hybrids often depends non-trivially on partial information of the adversary's initial choice. Namely, the hybrids are generalized cryptographic experiments with non-trivial $\mathcal{G}$ function. Since small-loss complexity leveraging has exponential security loss in the output length of $\mathcal{G}$, we require all hybrid experiments have logarithmic-length $\mathcal{G}$ function. Below, for convenience, we use the notation $\mathcal{X}_i$ to denote an ensemble of the form $\{X_{i,\lambda}\}$, and the notation $\mathcal{X}_I$ with a function I, as the ensemble $\{X_{I(\lambda),\lambda}\}$.

1. **Security via hybrids with logarithmic-length $\mathcal{G}$ function:** The proof involves a sequence of *polynomial* number $\ell(\star)$ of hybrid experiments. More precisely, for every $\lambda \in \mathbb{N}$, there is a sequence of $\ell(\lambda)+1$ hybrid (generalized) experiments $(H_{0,\lambda}, G_{0,\lambda}), \cdots (H_{\ell(\lambda),\lambda}, G_{\ell(\lambda),\lambda})$, such that, the "end" experiments matches the real experiments,

$$(\mathcal{H}_0, \mathcal{G}_0) = (\{H_{0,\lambda}\}, \{G_{0,\lambda}\}) = (\mathcal{RL}_0, \mathsf{null})$$
$$(\mathcal{H}_\ell, \mathcal{G}_\ell) = (\{H_{\ell(\lambda),\lambda}\}, \{G_{\ell(\lambda),\lambda}\}) = (\mathcal{RL}_1, \mathsf{null}),$$

Furthermore, there exists a function $L_G(\lambda) = O(\log \lambda)$ such that for every λ and i, the output length of $G_{i,\lambda}$ is at most $L_G(\lambda)$.

We next formalize required properties to lift security proof of neighboring hybrids. Towards this, we formulate indistinguishability of two generalized cryptographic experiments as a generalized cryptographic guessing game. The following is a known fact.

Fact. Let $(\mathcal{CH}_0, \mathcal{G}_0)$ and $(\mathcal{CH}_1, \mathcal{G}_1)$ be two ensembles of generalized cryptographic experiments, $\mathcal{F}$ be an ensemble of efficiently computable functions, and $\mathcal{C}_{\mathcal{F}}$ denote the class of non-uniform PPT adversaries $\mathcal{A}$ that are $\mathcal{F}$-selective in $(\mathcal{CH}_b, \mathcal{G}_b)$ for both $b = 0, 1$. Indistinguishability of $(\mathcal{CH}_0, \mathcal{G}_0)$ and $(\mathcal{CH}_1, \mathcal{G}_1)$ against (efficient) $\mathcal{F}$-selective adversaries is equivalent to $\mathcal{F}$-selective security of a generalized cryptographic guessing game $(\mathcal{D}, \mathcal{G}_0||\mathcal{G}_1, 1/2)$, where $\mathcal{G}_0||\mathcal{G}_1 = \{G_{0,\lambda}||G_{1,\lambda}\}$ are the concatenations of functions $G_{0,\lambda}$ and $G_{1,\lambda}$, and the challenger $\mathcal{D} = \{D_\lambda[CH_{0,\lambda}, CH_{1,\lambda}]\}$ proceeds as follows: For every security parameter $\lambda \in \mathbb{N}$, $D = D_\lambda[CH_{0,\lambda}, CH_{1,\lambda}]$, $G_b = G_{b,\lambda}$, $CH_b = CH_{b,\lambda}$, in experiment $\mathsf{Exp}(\lambda, D, G_0||G_1, \star)$,

- D tosses a random bit $b \overset{\$}{\leftarrow} \{0,1\}$.
- Upon receiving $g_0||g_1$ (corresponding to $g_d = G_d(\alpha)$ for $d = 0, 1$ where α is the initial choice of the adversary), D internally runs challenger CH_b by feeding it g_b and forwarding messages to and from CH_b.
- If the adversary aborts, D output 0. Otherwise, upon receiving the adversary's output bit b', it output 1 if and only if $b = b'$.

By the above fact, indistinguishability of neighboring hybrids $(\mathcal{H}_i, \mathcal{G}_i)$ and $(\mathcal{H}_{i+1}, \mathcal{G}_{i+1})$ against $\mathcal{F}$-selective adversary is equivalent to $\mathcal{F}$-selective security of the generalized cryptographic guessing game $(\mathcal{D}_i, \mathcal{G}_i||\mathcal{G}_{i+1}, 1/2)$, where $\mathcal{D}_i = \{D_{i,\lambda[H_{i,\lambda}, H_{i+1,\lambda}]}\}$. We can now state the required properties for every pair of neighboring hybrids:

2. Indistinguishability of neighboring hybrids via nice reduction For every neighboring hybrids $(\mathcal{H}_i, \mathcal{G}_i)$ and $(\mathcal{H}_{i+1}, \mathcal{G}_{i+1})$, their indistinguishability proof against fully selective adversary is established by a nice reduction $\mathcal{R}_i$ from the corresponding guessing game $(\mathcal{D}_i, \mathcal{G}_i||\mathcal{G}_{i+1}, 1/2)$ to some intractability assumption.

3. $\mathcal{G}_i||\mathcal{G}_{i+1}$-hiding For every neighboring hybrids $(\mathcal{H}_i, \mathcal{G}_i)$ and $(\mathcal{H}_{i+1}, \mathcal{G}_{i+1})$, their corresponding guessing game $(\mathcal{D}_i, \mathcal{G}_i||\mathcal{G}_{i+1}, 1/2)$ is $\mathcal{G}_i||\mathcal{G}_{i+1}$-hiding.

In summary,

Definition 11 (Nice Indistinguishability Proof). *A "nice" proof for the indistinguishability of two real experiments* $(\mathcal{RL}_0, \mathsf{null})$ *and* $(\mathcal{RL}_1, \mathsf{null})$ *is one that satisfy properties 1, 2, and 3 described above.*

It is now straightforward to lift security of nice indistinguishability proof by local application of small-loss complexity leveraging for neighboring hybrids. Please refer to the full version for its proof.

Theorem 5. *A "nice" proof for the indistinguishability of two real experiments* $(\mathcal{RL}_0, \mathsf{null})$ *and* $(\mathcal{RL}_1, \mathsf{null})$ *implies that these experiments are indistinguishable against fully adaptive adversaries.*

4 Adaptive Delegation for RAM computation

In this section, we introduce the notion of adaptive delegation for RAM computation ($\mathcal{DEL}$) and state our formal theorem. In a $\mathcal{DEL}$ scheme, a client outsources the database encoding and then generates a sequence of program encodings. The server will evaluate those program encodings with intended order on the database encoding left over by the previous one. For security, we focus on *full privacy* where the server learns nothing about the database, delegated programs, and its outputs. Simultaneously, $\mathcal{DEL}$ is required to provide *soundness* where the client has to receive the correct output encoding from each program and current database.
We first give a brief overview of the structure of the delegation scheme. First, the setup algorithm DBDel, which takes as input the database, is executed. The result is the database encoding and the secret key. PDel is the program encoding procedure. It takes as input the secret key, session ID and the program to be encoded. Eval takes as input the program encoding of session ID sid along with a memory encoding associated with sid. The result is an encoding which is output along with a proof. Along with this the updated memory state is also output. We employ a verification algorithm Ver to verify the correctness of computation using the proof output by Eval. Finally, Dec is used to decode the output encoding.
We present the formal definition below.

4.1 Definition

Definition 12 ($\mathcal{DEL}$ with Persistent Database).
A $\mathcal{DEL}$ scheme with persistent database, consists of PPT algorithms $\mathcal{DEL} = \mathcal{DEL}.\{\mathsf{DBDel}, \mathsf{PDel}, \mathsf{Eval}, \mathsf{Ver}, \mathsf{Dec}\}$, is described below. Let sid be the program session identity where $1 \leq \mathsf{sid} \leq l$. We associate $\mathcal{DEL}$ with a class of programs $\mathcal{P}$.

- $\mathcal{DEL}.\mathsf{DBDel}(1^\lambda, \mathsf{mem}^0, S) \to (\widetilde{\mathsf{mem}}^1, \mathsf{sk})$*: The database delegation algorithm* DBDel *is a randomized algorithm which takes as input the security parameter* 1^λ*, database* mem^0*, and a space bound* S*. It outputs a garbled database* $\widetilde{\mathsf{mem}}^1$ *and a secret key* sk*.*

- $\mathcal{DEL}.\mathsf{PDel}(1^\lambda, \mathsf{sk}, \mathsf{sid}, P_{\mathsf{sid}}) \to \widetilde{P}_{\mathsf{sid}}$*: The algorithm* PDel *is a randomized algorithm which takes as input the security parameter* 1^λ*, the secret key* sk*, the session ID* sid *and a description of a* RAM *program* $P_{\mathsf{sid}} \in \mathcal{P}$*. It outputs a program encoding* $\widetilde{P}_{\mathsf{sid}}$*.*

- $\mathcal{DEL}.\mathsf{Eval}\left(1^\lambda, T, S, \widetilde{P}_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}}\right) \to \left(c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}+1}\right)$*: The evaluating algorithm* Eval *is a deterministic algorithm which takes as input the security parameter* 1^λ*, time bound* T*, space bound* S*, a garbled program* $\widetilde{P}_{\mathsf{sid}}$*, and the database* $\widetilde{\mathsf{mem}}^{\mathsf{sid}}$*. It outputs* $(c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}+1})$ *or* $\perp$*, where* c_{sid} *is the encoding of the output* y_{sid}*,* σ_{sid} *is a proof of* c_{sid}*, and* $(y_{\mathsf{sid}}, \mathsf{mem}^{\mathsf{sid}+1}) = P_{\mathsf{sid}}(\mathsf{mem}^{\mathsf{sid}})$*.*

- $\mathcal{DEL}.\mathsf{Ver}(1^\lambda, \mathsf{sk}, c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}) \rightarrow b_{\mathsf{sid}} \in \{0,1\}$*: The verification algorithm takes as input the security parameter* 1^λ*, secret key* sk*, encoding* c_{sid}*, proof* σ_{sid} *and returns* $b_{\mathsf{sid}} = 1$ *if* σ_{sid} *is a valid proof for* c_{sid}*, or returns* $b_{\mathsf{sid}} = 0$ *if not.*

- $\mathcal{DEL}.\mathsf{Dec}(1^\lambda, \mathsf{sk}, c_{\mathsf{sid}}) \rightarrow y_{\mathsf{sid}}$*: The decoding algorithm* Dec *is a deterministic algorithm which takes as input the security parameter* 1^λ*, secret key* sk*, output encoding* c_{sid}*. It outputs* y_{sid} *by decoding* c_{sid} *with* sk*.*

Associated to the above scheme are correctness, (adaptive) security, (adaptive) soundness and efficiency properties.

Correctness. A delegation scheme $\mathcal{DEL}$ is said to be *correct* if both verification and decryption are correct: for all $\mathsf{mem}^0 \in \{0,1\}^{\mathrm{poly}(\lambda)}, 1 \leq \mathsf{sid} \leq \ell, P_{\mathsf{sid}} \in \mathcal{P}$, consider the following process:
- $(\widetilde{\mathsf{mem}}^1, \mathsf{sk}) \leftarrow \mathcal{DEL}.\mathsf{DBDel}(1^\lambda, \mathsf{mem}^0, S)$;
- $\widetilde{P}_{\mathsf{sid}} \leftarrow \mathcal{DEL}.\mathsf{PDel}(1^\lambda, \mathsf{sk}, \mathsf{sid}, P_{\mathsf{sid}})$;
- $(c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}+1}) \leftarrow \mathcal{DEL}.\mathsf{Eval}(1^\lambda, T, S, \widetilde{P}_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}})$;
- $b_{\mathsf{sid}} = \mathcal{DEL}.\mathsf{Ver}(1^\lambda, \mathsf{sk}, c_{\mathsf{sid}}, \sigma_{\mathsf{sid}})$;
- $y_{\mathsf{sid}} = \mathcal{DEL}.\mathsf{Dec}(1^\lambda, \mathsf{sk}, c_{\mathsf{sid}})$;
- $(y'_{\mathsf{sid}}, \mathsf{mem}^{\mathsf{sid}+1}) \leftarrow P_{\mathsf{sid}}(\mathsf{mem}^{\mathsf{sid}})$;

The following holds:

$$\Pr\left[(y_{\mathsf{sid}} = y'_{\mathsf{sid}} \wedge b_{\mathsf{sid}} = 1)\ \forall \mathsf{sid}, 1 \leq \mathsf{sid} \leq l\right] = 1.$$

Adaptive Security (full privacy). This property is designed to protect the privacy of the database and the programs from the adversarial server. We formalize this using a simulation based definition. In the real world, the adversary is supposed to declare the database at the beginning of the game. The challenger computes the database encoding and sends it across to the adversary. After this, the adversary can submit programs to the challenger and in return it receives the corresponding program encodings. We emphasize the program queries can be made adaptively. On the other hand, in the simulated world, the simulator does not get to see either the database or the programs submitted by the adversary. But instead it receives as input the length of the database, the lengths of the individual programs and runtimes of all the corresponding computations.[6] It then generates the simulated database and program encodings. The job of the adversary in the end is to guess whether he is interacting with the challenger (real world) or whether he is interacting with the simulator (ideal world).

Definition 13. *A delegation scheme* $\mathcal{DEL} = \mathcal{DEL}.\{\mathsf{DBDel}, \mathsf{PDel}, \mathsf{Eval}, \mathsf{Ver}, \mathsf{Dec}\}$ *with persistent database is said to be* adaptively secure *if for all sufficiently large* $\lambda \in \mathbb{N}$*, for all total round* $l \in \mathrm{poly}(\lambda)$*, time bound*

[6] Note that unlike the standard simulation based setting, the simulator does not receive the output of the programs. This is because the output of the computation is never revealed to the adversary.

T, space bound S, for every interactive PPT adversary $\mathcal{A}$, there exists an interactive PPT simulator $\mathcal{S}$ such that $\mathcal{A}$'s advantage in the following security game $\textsf{\textit{Exp-Del-Privacy}}(1^\lambda, \mathcal{DEL}, \mathcal{A}, \mathcal{S})$ *is at most negligible in λ.*

$\textsf{\textit{Exp-Del-Privacy}}(1^\lambda, \mathcal{DEL}, \mathcal{A}, \mathcal{S})$

1. *The challenger $\mathcal{C}$ chooses a bit $b \in \{0, 1\}$.*
2. *$\mathcal{A}$ chooses and sends database mem^0 to challenger $\mathcal{C}$.*
3. *If $b = 0$, challenger $\mathcal{C}$ computes $(\widetilde{\mathsf{mem}}^1, \mathsf{sk}) \leftarrow \mathcal{DEL}.\mathsf{DBDel}(1^\lambda, \mathsf{mem}^0, S)$. Otherwise, $\mathcal{C}$ simulates $(\widetilde{\mathsf{mem}}^1, \mathsf{sk}) \leftarrow \mathcal{S}(1^\lambda, |\mathsf{mem}^0|)$, where $|\mathsf{mem}^0|$ is the length of mem^0. $\mathcal{C}$ sends $\widetilde{\mathsf{mem}}^1$ back to $\mathcal{A}$.*
4. *For each round sid from 1 to l,*
   (a) $\mathcal{A}$ chooses and sends program P_{sid} to $\mathcal{C}$.
   (b) If $b = 0$, challenger $\mathcal{C}$ sends $\widetilde{P}_{\mathsf{sid}} \leftarrow \mathcal{DEL}.\mathsf{PDel}(1^\lambda, \mathsf{sk}, \mathsf{sid}, P_{\mathsf{sid}})$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ simulates and sends $\widetilde{P}_{\mathsf{sid}} \leftarrow \mathcal{S}(1^\lambda, \mathsf{sk}, \mathsf{sid}, 1^{|P_{\mathsf{sid}}|}, 1^{|c_{\mathsf{sid}}|}, T, S)$ to $\mathcal{A}$.
5. *$\mathcal{A}$ outputs a bit b'. $\mathcal{A}$ wins the security game if $b = b'$.*

We notice that an unrestricted *adaptive adversary* can adaptively choose RAM programs P_i depending on the program encodings it receives, whereas a restricted *selective adversary* can only make the choice of programs statically at the beginning of the execution.

Adaptive Soundness. This property is designed to protect the clients against adversarial servers producing invalid output encodings. This is formalized in the form of a security experiment: the adversary submits the database to the challenger. The challenger responds with the database encoding. The adversary then chooses programs to be encoded adaptively. In response, the challenger sends the corresponding program encodings. In the end, the adversary is required to submit the output encoding and the corresponding proof. The soundness property requires that the adversary can only submit a convincing "false" proof only with negligible probability.

Definition 14. *A delegation scheme $\mathcal{DEL}$ is said to be* adaptively sound *if for all sufficiently large $\lambda \in \mathbb{N}$, for all total round $l \in \mathrm{poly}(\lambda)$, time bound T, space bound S, there exists an interactive PPT adversary $\mathcal{A}$, such that the probability of $\mathcal{A}$ win in the following security game* $\textsf{\textit{Exp-Del-Soundness}}(1^\lambda, \mathcal{DEL}, \mathcal{A})$ *is at most negligible in λ.*

$\textsf{\textit{Exp-Del-Soundness}}(1^\lambda, \mathcal{DEL}, \mathcal{A})$

1. *$\mathcal{A}$ chooses and sends database mem^0 to challenger $\mathcal{C}$.*
2. *The challenger $\mathcal{C}$ computes $(\widetilde{\mathsf{mem}}^1, \mathsf{sk}) \leftarrow \mathcal{DEL}.\mathsf{DBDel}(1^\lambda, \mathsf{mem}^0, S)$. $\mathcal{C}$ sends $\widetilde{\mathsf{mem}}^1$ back to $\mathcal{A}$.*
3. *For each round sid from 1 to l,*
   (a) $\mathcal{A}$ chooses and sends program P_{sid} to $\mathcal{C}$.
   (b) $\mathcal{C}$ sends $\widetilde{P}_{\mathsf{sid}} \leftarrow \mathcal{DEL}.\mathsf{PDel}(1^\lambda, \mathsf{sk}, \mathsf{sid}, P_{\mathsf{sid}})$ to $\mathcal{A}$.
4. *$\mathcal{A}$ outputs a triplet (k, c_k^*, σ_k^*). $\mathcal{A}$ wins the security game if $1 \leftarrow \mathcal{DEL}.\mathsf{Ver}(1^\lambda, \mathsf{sk}, c_k^*, \sigma_k^*)$ and $c_k^* \neq c_k$ for the k-th round, where c_k is generated as follows: for $\mathsf{sid} = 1, \ldots, k$, $(c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}+1}) \leftarrow \mathcal{DEL}.\mathsf{Eval}(1^\lambda, T, S, \widetilde{P}_{\mathsf{sid}}, \widetilde{\mathsf{mem}}^{\mathsf{sid}})$.*

Efficiency. For every session with session ID sid, we require that DBDel and PDel execute in time $\text{poly}(\lambda, |\mathsf{mem}^0|)$ and $\text{poly}(\lambda, |P_{\mathsf{sid}}|)$ respectively. Furthermore we require that Eval run in time $\text{poly}(\lambda, t^*_{\mathsf{sid}})$, where t^*_{sid} denotes the running time of P_{sid} on $\mathsf{mem}^{\mathsf{sid}}$. We require that both Ver and Dec run in time $\text{poly}(\lambda, |y_{\mathsf{sid}}|)$. Finally, the length of $c_{\mathsf{sid}}, \sigma_{\mathsf{sid}}$ should depend only on $|y_{\mathsf{sid}}|$.
A construction of adaptive delegation is provided in the full version [2] with its security proof.

Theorem 6. *Assuming the existence of $\mathsf{i}\mathcal{O}$ for circuits and DDH, there exists an efficient RAM delegation scheme $\mathcal{DEL}$ with persistent database with adaptive security and soundness.*

Acknowledgements

We thank Yael Kalai for insightful discussions in the early stages of this project.
This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.
Prabhanjan Ananth is supported in part by grant #360584 from the Simons Foundation and supported in part from a DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C-0205. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.
Kai-Min Chung was partially supported by Ministry of Science and Technology, Taiwan, under Grant no. MOST 103-2221-E-001-022-MY3.
Huijia Lin was partially supported by NSF grants CNS-1528178 and CNS-1514526.

References

1. Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. From selective to adaptive security in functional encryption. In *CRYPTO*, 2015.
2. Prabhanjan Ananth, Yu-Chi Chen, Kai-Min Chung, Huijia Lin, and Wei-Kai Lin. Delegating ram computations with adaptive soundness and privacy. Cryptology ePrint Archive, Report 2015/1082, 2015. http://eprint.iacr.org/2015/1082.
3. Prabhanjan Ananth, Aayush Jain, Moni Naor, Amit Sahai, and Eylon Yogev. Universal obfuscation and witness encryption: Boosting correctness and combining security. In *CRYPTO*, 2016.
4. Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. pages 308–326, 2015.

5. Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Achieving compactness generically: Indistinguishability obfuscation from non-compact functional encryption. *IACR Cryptology ePrint Archive*, 2015:730, 2015.
6. Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Patchable obfuscation. *IACR Cryptology ePrint Archive*, 2015:1084, 2015.
7. Prabhanjan Ananth and Amit Sahai. Functional encryption for turing machines. In *TCC 2016-A*.
8. Applebaum, Benny. Bootstrapping obfuscators via fast pseudorandom functions. In *ASIACRYPT*, 2014.
9. Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In *EUROCRYPT*, 2014.
10. Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Siddartha Telang. Succinct randomized encodings and their applications. In *STOC*, 2015.
11. Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 171–190, 2015.
12. Elette Boyle, Kai-Min Chung, and Rafael Pass. Oblivious parallel RAM and applications. In *TCC 2016-A*.
13. Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In *TCC*, 2014.
14. Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *TCC*, 2014.
15. Ran Canetti, Yilei Chen, Justin Holmgren, and Mariana Raykova. Succinct adaptive garbled RAM. In *TCC 2016-B*.
16. Ran Canetti and Justin Holmgren. Fully succinct garbled RAM. ITCS, 2016.
17. Ran Canetti, Justin Holmgren, Abhishek Jain, and Vinod Vaikuntanathan. Indistinguishability obfuscation of iterated circuits and RAM programs. In *STOC*, 2015.
18. Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 468–497, 2015.
19. Binyi Chen, Huijia Lin, and Stefano Tessaro. Oblivious parallel ram: Improved efficiency and generic constructions. In *TCC 2016-A*.
20. Yu-Chi Chen, Sherman S. M. Chow, Kai-Min Chung, Russell W. F. Lai, Wei-Kai Lin, and Hong-Sheng Zhou. Cryptography for parallel RAM from indistinguishability obfuscation. ITCS, 2016.
21. Kai-Min Chung and Rafael Pass. A simple ORAM. *IACR Cryptology ePrint Archive*, 2013:243, 2013.
22. Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrede Lepoint, Hemanta K Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In *CRYPTO*. 2015.
23. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO*, 2013.

24. Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In *CRYPTO*. 2015.
25. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, 2013.
26. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.
27. Sanjam Garg, Steve Lu, and Rafail Ostrovsky. Black-box garbled RAM. In *FOCS*, 2015.
28. Sanjam Garg, Steve Lu, Rafail Ostrovsky, and Alessandra Scafuro. Garbled RAM from one-way functions. In *STOC*, 2015.
29. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *TCC*. 2015.
30. Craig Gentry, Shai Halevi, Steve Lu, Rafail Ostrovsky, Mariana Raykova, and Daniel Wichs. Garbled RAM revisited. In *EUROCRYPT*, 2014.
31. Craig Gentry, Shai Halevi, Mariana Raykova, and Daniel Wichs. Outsourcing private RAM computation. In *FOCS*, 2014.
32. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO*, 2012.
33. Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In *TCC*. 2010.
34. Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. In *CRYPTO*, 2016.
35. Yael Tauman Kalai and Omer Paneth. Delegating RAM computations. *IACR Cryptology ePrint Archive*, 2015:957, 2015.
36. Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: The power of no-signaling proofs. In *STOC*, 2014.
37. Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In *STOC*, 2015.
38. Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 28–57. Springer, 2016.
39. Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. Cryptology ePrint Archive, Report 2016/795, 2016. http://eprint.iacr.org/2016/795.
40. Steve Lu and Rafail Ostrovsky. How to garble RAM programs. In *EUROCRYPT*, 2013.
41. Steve Lu and Rafail Ostrovsky. Black-box parallel garbled RAM. Cryptology ePrint Archive, Report 2015/1068, 2015. http://eprint.iacr.org/2015/1068.

42. Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In *CRYPTO*, 2016.
43. Moni Naor. On cryptographic assumptions and challenges. In *CRYPTO*, 2003.
44. Brent Waters. A punctured programming approach to adaptively secure functional encryption. In *CRYPTO*, 2015.