Monotone Batch NP-Delegation with Applications to Access Control

Zvika Brakerski* Yael Tauman Kalai†

Abstract

Consider an access policy for some resource which only allows access to users of the system who own a certain set of attributes. Specifically, we consider the case where such an access structure is defined by a *monotone formula* (or logarithmic depth circuit) $\mathcal{F} : \{0,1\}^N \to \{0,1\}$, where N is the number of possible attributes.

In this work we present two results, which we believe to be of individual interest even regardless of the above application, and show how to combine them to achieve a *succinct* single-round *private* access control protocol. That is, a verifier can be convinced that an approved user (i.e. one which holds an approved set of attributes) is accessing the system, without learning any additional information about the user or the set of attributes.

First, assuming a computational PIR scheme (which can be based, for example, on the polynomial hardness of the LWE assumption), we construct for any **NP** language L, a *succinct* single-round (2-message) protocol for delegating *monotone batch L computations*. Explicitly, for every $N \in \mathbb{N}$, every $x_1, \ldots, x_N \in \{0,1\}^n$, and every monotone formula $\mathcal{F} : \{0,1\}^N \to \{0,1\}$, a prover can *succinctly* prove that $\mathcal{F}(\mathbf{1}_{x_1 \in L}, \ldots, \mathbf{1}_{x_N \in L}) = 1$, where $\mathbf{1}_{x_i \in L} = 1$ if and only if $x_i \in L$, and where the communication complexity is $m \cdot \text{polylog}(N)$ where m is the length of a *single* witness.

Second, assuming a quasi-polynomially secure two-message oblivious transfer scheme with statistical sender privacy (which can be based on quasi-polynomial hardness of the DDH, QR or DCR assumptions), we show how to convert *any* single-round protocol into a *witness indistinguishable* one, with similar communication complexity.

1 Introduction

Verifying the correctness of computations is one of the most fundamental tasks in computer science. The renowned complexity class **NP** is defined as the class of problems whose computation can be verified in polynomial time. Formally, an **NP** language L is defined using a polynomial time computable relation R over pairs (x, w) (where $|w| = \text{poly}(|x|)$), and $x \in L$ if and only if $\exists w.\ (x, w) \in R$. This means that a statement of the form $x \in L$ can be verified in polynomial time given a suitable witness (namely, w).

In the cryptographic setting we wonder whether it is possible to verify **NP** statements more efficiently than by reading the entire witness w, while relaxing the soundness requirement to only

*Weizmann Institute of Science. Supported by the Israel Science Foundation (Grant No. 468/14), Binational Science Foundation (Grants No. 2016726, 2014276) and European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482) and via Project PROMETHEUS (Grant 780701).

†Microsoft Research and Massachusetts Institute of Technology.

apply against computationally bounded cheating provers (this is also known as an *argument system*). Moreover, we require that the honest prover can generate such a succinct proof efficiently given w. In this work, we focus on *single-round* (2-message) arguments, where the verifier sends a query message to the prover, who responds with an answer string, and based on the answer the verifier decides whether to accept or reject. Such a single-round argument system is said to be *succinct* if its communication complexity, and the verifier computational complexity, is $o(|w|)$ (ideally, for security parameter λ, the communication complexity should be only $\mathrm{poly}(\log|x|, \lambda)$ and the verifier's computational complexity should be only $|x| \cdot \mathrm{poly}(\lambda)$).

It is known how to construct such succinct single-round arguments for **NP** in the random oracle model [Mic94], and under knowledge assumptions [DFH12, BCCT13, BCC+14]. However, we do not have constructions under any standard (falsifiable) assumptions, and not even under the assumption that indistinguishability obfuscation (iO) exists. That said, for special classes of **NP** statements, we do know how to construct succinct single-round arguments based on standard (falsifiable) assumptions.

Specifically, Brakerski, Holmgren and Kalai [BHK17] construct an argument system for *batch* **NP** *delegation*, assuming the existence of a (polynomially secure) computational PIR scheme. A batch statement is of the form $(x_1 \in L) \wedge \cdots \wedge (x_N \in L)$. Most generally, a witness for such a statement is a tuple $(w_1, \ldots, w_N)$ s.t. w_i is a witness for $x_i \in L$. Thus, if the length of an instance is $|x_i| = n$ and the length of a single witness is $|w_i| = m$ $(= \mathrm{poly}(n))$, then the witness length for the batch statement is $N \cdot m$. Brakerski et al. [BHK17] showed an argument system with communication complexity and verifier computation $(m + \mathrm{polylog}(n, N)) \cdot \mathrm{poly}(\lambda)$, where λ is the security parameter. In addition, in a followup work to this current work, Badrinarayanan et al. [BKK+17] construct a single-round argument system for the class NTISP. We elaborate on this followup work after presenting our result in Section 1.1.

We also mention the works of Reingold, Rothblum and Rothblum [RRR16, RRR18], which studies multi-round batch *proofs* (i.e. with statistical soundness) for sub-classes of **NP**, and posed the open problem of constructing an interactive proof (with statistical soundness) for batch **NP** statements with communication complexity $m \cdot \mathrm{poly}(\lambda, \log(n, N))$ and where the prover is efficient given the witnesses.

1.1 Our Results

In this work we extend the class of **NP** statements that admit succinct single-round argument systems. Specifically, we present a succinct arguments for *monotone batch* **NP** statements. Such a statement is characterized by a collection of instances $x_1, \ldots, x_N$ respective to a language L, and a monotone formula (i.e. without negation gates) $\mathcal{F} : \{0,1\}^N \to \{0,1\}$. The statement $((x_1, \ldots, x_N), \mathcal{F})$ holds if $\mathcal{F}(\mathbf{1}_{x_1 \in L}, \ldots, \mathbf{1}_{x_N \in L}) = 1$, where $\mathbf{1}_{x_i \in L} = 1$ if and only if $x_i \in L$. For example, we can manage statements of the form $(((x_1 \in L) \wedge (x_2 \in L)) \vee (x_3 \in L)) \wedge (x_4 \in L)$, and much more. In order to produce an accepting proof, an honest prover needs a set of witnesses for a subset $S \subseteq [N]$ of the x_i's that makes the formula accept. Namely a set of witnesses $\{w_i\}_{i \in S}$ so that w_i is a witness for x_i and the set S is sufficient for $\mathcal{F}$ to accept; i.e., $\mathcal{F}(\mathbf{1}_{1 \in S}, \ldots, \mathbf{1}_{N \in S}) = 1$. Since $\mathcal{F}$ is monotone, this indeed implies that $\mathcal{F}(\mathbf{1}_{x_1 \in L}, \ldots, \mathbf{1}_{x_N \in L}) = 1$ (since S is a subset of the x_i's that are in L).

The aforementioned batch **NP** statements are a special case where $\mathcal{F}$ only contains an AND operation. In the AND special case, the prover must prove w.r.t. the entire set $S = [N]$. In contrast, in the monotone batch case there could exist many possible sets S that yield an accepting proof.

This makes the verifier's task much harder, since the prover can answer every time with respect to a different accepting subset S, depending on the query being asked. In particular, the prover can "claim" that $x_i \notin L$ for any specific x_i (of course not for all at the same time) and not be "caught in a lie".

In terms of efficiency and assumptions, our protocol is asymptotically comparable to the more restricted [BHK17]. The communication complexity and verifier computational complexity are $(m + \text{polylog}(n, N)) \cdot \text{poly}(\lambda)$, and our cryptographic building block is a succinct single server (computationally secure) private information retrieval scheme. These can be instantiated based on the phi-hiding assumption [CMS99] or on the Learning with Errors assumption [BV11]. Furthermore, similarly to [BHK17], our scheme has a *proof-of-knowledge* property, meaning that one can efficiently extract a valid witness $\{w_i\}_{i \in S}$ from any (possibly cheating) prover that convinces the verifier to accept with non-negligible probability.

Our scheme is *privately verifiable*, i.e. the verifier needs to keep a secret state that is used to verify the prover's answer. We note that one can can construct a single-round *publicly verifiable* scheme at the price of relying on the Random Oracle Model [Mic94] or on knowledge assumptions [DFH12, BCCT13, BCC$^+$14]. However, all known single-round schemes that rely on standard falsifiable assumptions, even those for deterministic computations, are all privately verifiable.

Independent and followup work. A similar result was obtained independently by Holmgren [Hol18]. In a followup work, Badrinarayanan et al. [BKK$^+$17] construct a single-round delegation scheme for all *bounded space* non-deterministic computations, assuming the existence of a subexponentially secure computational PIR scheme. Namely, for any language L computable by an S-space and T-time non-deterministic Turing machine, they construct a single-round delegation scheme for L, where the communication complexity is $\text{poly}(S, \lambda)$. We mention that this protocol has adaptive soundness (whereas ours does not), but relies on sub-exponential hardness assumptions (whereas our protocol relies on polynomial hardness assumptions).

An open problem that remains open from these works, is constructing a succinct single-round monotone batch **NP** delegation scheme for monotone *circuits*, as opposed to monotone formulas. To date, we only know how to do this based on knowledge assumptions or in the random oracle model.

We refer the reader to Section 1.3 for a high-level overview of our techniques, and to Section 3 for the formal treatment.

Witness Indistinguishability. We show a *generic transformation* that converts any single-round (2-message) delegation scheme into one that is also *witness indistinguishable* (WI), *without blowing up the communication complexity*. This transformation relies on the existence of a quasi-poly secure OT scheme, which can be based on the quasi-polynomial hardness of the DDH, QR or Paillier's decisional composite residuosity assumption (DCR). The communication complexity and verifier complexity remain unchanged up to $\text{poly}(\lambda)$ factors. This transformation relies on a recent 2-message strong WI protocol in the delayed input setting, proposed by [JKKR17]. See details in Section 1.3 and Section 4. We note while we achieve computational WI, it may be possible to achieve statistical witness indistinguishability using the results and techniques of [KKS18].

1.2 Applications

We show two applications of the above two results.

Delegation Beyond Monotonicity. Our basic protocol shows that batch **NP** statements are not

the most general subclass of **NP** for which succinct arguments exist under standard assumptions. It is a very interesting open problem to characterize this class of statements.

Simply removing the monotonicity requirement while still only relying on falsifiable assumptions would imply falsifiable succinct **NP**-delegation,[1] which is perhaps the most important open problem in the delegation literature. We therefore believe that additional techniques may be require to overcome the monotonicity condition altogether.

That said, our protocol can go beyond monotone forumlas if the language L is in $\mathbf{NP} \cap \mathbf{coNP}$. In such case, we can prove *general* batch statements, i.e. of the form $\mathcal{F}(\mathbf{1}_{x_1 \in L}, \ldots, \mathbf{1}_{x_N \in L}) = 1$ even for *non-monotone* formulas. This can be done by "monotonizing" the formula by pushing all negation gates to the input layer. We get a monotone formula $\mathcal{F}'$ where $\mathcal{F}(\mathbf{1}_{x_1 \in L}, \ldots, \mathbf{1}_{x_N \in L}) = \mathcal{F}'(\mathbf{1}_{x_1 \in L}, \mathbf{1}_{x_1 \in \overline{L}}, \ldots, \mathbf{1}_{x_N \in L}, \mathbf{1}_{x_N \in \overline{L}})$. Since $\mathcal{F}'$ is monotone and all of its inputs are **NP** statements (since $\overline{L} \in \mathbf{NP}$), our monotone delegation protocol can be used in this case as well.

Succinct Single-Round Access Control. By applying our WI transformation to our succinct single-round argument, we get a succinct single-round witness indistinguishable argument system for monotone batch **NP**, which we call "a succinct access control scheme".

Consider a setting where there are N public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_N$ (for a very large N), and each user receives for some subset $S \subset [N]$ (corresponding to his credentials) a set of secret keys $\{\mathsf{sk}_i\}_{i \in S}$, where each sk_i corresponds to pk_i. Now suppose a user wishes to prove *anonymously* and *succinctly* that his credentials satisfy some monotone formula $\mathcal{F} : \{0,1\}^N \to \{0,1\}$. Namely, he wishes to prove that his set S satisfies $\mathcal{F}(\mathbf{1}_{1 \in S}, \ldots, \mathbf{1}_{n \in S}) = 1$. Combining our two main results (monotone **NP** delegation and our WI transformation) we obtain a single-round *succinct* and *anonymous* scheme, where a user can succinctly prove that his set of secret keys satisfies some monotone access structure (formulated as a monotone formula), where the anonymity property is WI and the length of a proof is $|\mathsf{sk}_i| \cdot \mathrm{poly}(\log N, \lambda)$, where $|\mathsf{sk}_i|$ is the length of a *single* secret key, and λ is the security parameter. We call such a scheme a *succinct single-round access control scheme*.

Moreover, we can make our scheme *collusion resilient*. Namely, we can ensure that if two users have credentials corresponding to two sets $S_1, S_2 \subseteq [N]$, then together they cannot get credentials corresponding to $S_1 \cup S_2$, and moreover together they cannot prove more than what each user could have proven individually. This is done by introducing a signature scheme and setting each secret key to be a signature on the attribute concatenated with a random tag that is unique for the user. The random tags will prevent mixing an matching between different users' attributes. We refer to Section 5 for the formal definition and the construction.

We note that our notion of access control systems is similar to the notion of *anonymous credentials* [Cha85]. We identify two main differences between the two notions. One is that anonymous credentials require anonymity even against the issuer of the credentials, whereas in our model the issuer is a trusted party. The second is that anonymous credentials are not required to be succinct, in the sense that the proof could depend on the number of attributes, whereas succinctness is a cornerstone in the definition of access control systems. We believe that our techniques may be useful towards the construction of *succinct* anonymous credential schemes under standard assumptions by replacing the signature scheme from our construction in Section 5 with *blind signatures* [Cha82].

[1] This can be done by taking L to be the trivial language (i.e. all x is in L, this language is in **NP** with an empty witness). To provide a succinct proof that a 3-CNF formula ϕ on N variables is satisfiable, we will ask the prover to provide a non-monotone formula delegation for $(x_1, \ldots, x_N, \psi)$ with $x_1 = \cdots = x_N = 0$.

1.3 Technical Overview

We start by explaining the high-level ideas behind our monotone batch **NP** delegation scheme, and then proceed with the high-level ideas behind our WI transformation.

Monotone Batch NP Delegation. Our starting point is the delegation scheme of [KRR14]. This delegation scheme was proven sound for **P**, and moreover, was shown to imply some form of *local soundness* even for **NP**. This was formulated by Paneth and Rothblum [PR14], via the notion of *local assignment generator*. More specifically, they show that the [KRR13,KRR14] analysis implies the following: If the [KRR13,KRR14] delegation scheme has communication complexity $k \cdot \text{poly}(\lambda)$ then one can convert any (possibly cheating) prover who proves that a circuit Φ is satisfiable, into a k-local assignment generator for the wires of Φ.

A k-local assignment generator for the satisfiability of Φ is a randomized algorithm that takes as input a tuple of k wires of Φ and returns an assignment to the values of these wires in a *locally consistent and no signaling* manner:

- Local consistency means that the k returned values need to be consistent within themselves, i.e. when querying both the input wires and the output wire of a gate, the output value will be equal to the application of the gate on the given input values (with overwhelming probability). Furthermore, the marginal distribution given by the local assignment generator to the output wire of Φ is 1 (with overwhelming probability).
- No signaling means that for every set of $\ell \leq k$ wires, when querying a superset of these wires, the *marginal distribution* of the ℓ-tuple of answers for these wires will be the same, regardless of what other wires are included in the k-tuple superset. More precisely, as in [BHK17] we only require *computational* no-signaling, which is the requirement that any change in the marginal distribution of the ℓ-tuple is indistinguishable to any polynomial time distinguisher.

The local assignment generator abstraction is used in [KRR13,KRR14] to achieve **P** delegation, which is just CIRCUIT-SAT for Φ_x, where Φ_x has no input wires at all (but instead has the input x hard-coded). Specifically, they show that the existence of a local assignment generator for Φ_x implies that indeed $\Phi(x) = 1$. Loosely speaking, this is done by iteratively proving that the assignment generator must always be consistent with the "correct" wire values. Unfortunately, in this iterative process there is a "decay" of the probability of consistency with each level of the circuit. Roughly speaking, if one applies the union bound on the probability of inconsistency of the input wires of a gate, then the probability of inconsistency of the output wire can double. Thus the decay in consistency can be exponential in the depth of the circuit.

Starting from [KRR14] it is shown how to overcome this obstacle and limit the decay to being polynomial in the size of the circuit for **P** delegation. Looking ahead to our contribution, unfortunately we were unable to apply these techniques in the monotone **NP** setting. Thus the aforementioned decay limits us to working with monotone *formula* (which can always be balanced to have logarithmic depth) rather than monotone *circuit*.

Be it circuits or formulas, applying the above outline in the **NP** setting is much more difficult since there is no single "correct" wire assignment, as any witness induces different wire values to the circuit. In [BHK17] batch **NP** delegation is achieved as follows. First, they notice that if $k = m + \text{polylog}(n)$ then **NP** delegation can be proven.[2] This is done by querying the assignment

[2]This parameter regime is not interesting as an end result since we can always achieve m-bit delegation by just sending the witness. However, this will be instrumental for our exposition.

generator only on k-tuples that always contain the m input wires (i.e. the wires that correspond to the witness that Φ is satisfiable). As explained above, the m input wires do not necessarily take the same value every time so one cannot apply the [KRR13, KRR14] technique as is. Instead, in [BHK17] they propagate the assertion that "the values of the queried wires are *globally* consistent with the values of the queried input wires". It is important to notice that this proof strategy allows to achieve more than soundness, in fact it allows to *extract* a valid **NP** witness from a (possibly cheating) prover that convinces the verifier to accept with non-negligible probability.

To prove batch statements they notice that now the circuit Φ can be thought of as containing N sub-circuits, each computing an **NP** statement. However, there is no requirement for consistency *between* sub-circuits. It is sufficient that each sub-circuit is globally consistent amongst itself. Many details that are immaterial for this high level overview are omitted from this abbreviated description of these prior works.

Our Monotone Batch Delegation Scheme. For monotone batch **NP** statements, the situation is even more complicated than in [BHK17]. Indeed, the relevant circuit Φ can be described as a collection of N sub-circuits verifying **NP** relations, with the monotone formula $\mathcal{F}$ applied to the output of these **NP** verification circuits. We assume that $\mathcal{F}$ is balanced, i.e. $\mathsf{depth}(\mathcal{F}) = O(\log |\mathcal{F}|)$ as this can be guaranteed w.l.o.g (while preserving the monotonicity of $\mathcal{F}$). Unlike the setting considered in [BHK17], here cross-consistency between different sub-circuits is very important. Indeed, the assignment generator may answer according to a mixed distribution corresponding to various accepting sets $S \subseteq [N]$. We can therefore no longer rely on the global consistency of the sub-circuits alone, since each sub-circuit individually can have output value that takes 0 almost all the time. We show that the monotonicity of $\mathcal{F}$ can come to our rescue in tackling this problem.

Assume that indeed there is global consistency within each sub-circuit. Now consider those sub-circuits whose output wire takes value 1 with probability at least ϵ, for some inverse polynomial ϵ. Then it is possible to use an extraction process analogous to the one for [BHK17] described above to extract **NP** witnesses for those sub-circuits in $\mathrm{poly}(1/\epsilon)$ time. Choosing ϵ to be a small enough polynomial (specifically $1/\mathrm{poly}(|\mathcal{F}|) = 1/\exp(\mathsf{depth}(\mathcal{F}))$), we can work in polynomial time and "harvest" witnesses for all of those sub-circuits. Intuitively, other sub-circuits (ones whose output wire is assigned 1 with probability less than ϵ) should not have much effect on the output of $\mathcal{F}$ since even applying the union bound, the probability that *any* of them takes a 1 value is very small. We show that indeed letting S^* denote the set of sub-circuits whose probability of accepting is $\geq \epsilon$, it holds that $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*}) = 1$. This statement will conclude the proof since we are able to extract all witnesses for $\{x_i\}_{i\in S^*}$ and therefore it must be the case that indeed $\mathcal{F}(\mathbf{1}_{x_1\in L}, \ldots, \mathbf{1}_{x_N\in L}) = 1$ (since $\mathcal{F}$ is monotone), and furthermore we extracted a witness for this statement (the set of witnesses for $\{x_i\}_{i\in S^*}$).

In order to prove that $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*}) = 1$, we rely on "promoting the consistency" similarly to the outline in [KRR13, KRR14, BHK17] described above. Specifically we show that the assignment generator cannot assign value 1 to a wire which takes value 0 in the computation of $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$, except with very low probability. Once this is established, we rely on the assignment generator's assigning value 1 with high probability to the output wire to conclude that $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*}) = 1$. By definition of S^*, input wires to $\mathcal{F}$ with value 0 in the evaluation of $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$ are indeed assigned 1 with probability at most ϵ. The argument proceeds iteratively going gate by gate in topological order (or equivalently layer by layer). Consider a gate whose output wire takes a value 0 in $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$, then if it is an OR gate then both of its inputs take value 0 in $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$, and by local consistency its probability to be assigned

1 is at most twice the probability of its inputs. For AND gates the situation is slightly better since we only need to argue that one of the inputs is 0 with high probability. As can be seen from this process, the probability of 1 for a wire that takes a 0 in $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$ grows from ϵ in the input layer to at most $\epsilon \cdot 2^{\mathsf{depth}(\mathcal{F})}$ in the output wire, and since we chose $\epsilon \ll 2^{-\mathsf{depth}(\mathcal{F})}$ it follows that since the assignment generator assigns 1 to the output wire with high probability, it must be the case that this wire takes value 1 in $\mathcal{F}(\mathbf{1}_{1\in S^*}, \ldots, \mathbf{1}_{N\in S^*})$. We refer the reader to Section 3 for the formal proof.

To conclude, let us briefly explain why we could not apply the methods from [KRR13, KRR14, BHK17] to remove the exponential decay in the depth. At a high level, these works use a commitment scheme (be it information theoretic in the form of a low degree extension or computational in the form of collision resistant hashing) to enforce a statement about the values in an entire layer of the evaluated circuit. Once the assignment generator assigns a value to the commitment, it is in fact bound to the values of the entire layer. In the monotone setting the assigner should be allowed not to commit to the value of the layer, nor to its consistency with some input layer, but rather only to not "turn off" some of the wires in the layer (whose identity is only determined in the proof and is not known a-priori). We could not devise an appropriate commitment mechanism to allow this enforcement, hence our inability to support monotone circuits, however we do not see an obvious barrier that prevents doing this in principle.

Witness Indistinguishability. We show how to convert any single-round (2-message) argument system (and in particular, our single-round delegation protocol) with super-polynomial security into a witness indistinguishable one, with minimal (asymptotic) blowup to the communication complexity, albeit witness indistinguishability holds only against polynomial time distinguishers. We note that we can get super-polynomial security by properly strengthening the assumption, namely for any function $T(\lambda) \geq \lambda$ (where λ is the security parameter), if the original scheme was secure against any poly(T)-size adversary then we get witness indistinguishability against all $T^{o(1)}$-size adversaries. Furthermore, if the original protocol is extractable then the transformation would allow to apply the extractor as well.

The basic idea is for the verifier to simply send the first message of the protocol, and for the prover to compute its response according to the protocol, but rather than sending it to the verifier "in the clear", it will send a *statistically binding commitment* to the response. The idea is then for the prover to provide a WI proof (in parallel) that he indeed sent a commitment to an accepting response to the verifier's first message.

This idea runs into several obstacles, let us present the most severe ones. First, the original protocol may not be publicly verifiable (and indeed we would like to apply it to our aforementioned privately verifiable protocol), in which case the prover cannot prove that he is committing to a message that corresponds to an accepting response, since he does not know the verifier's verdict function. Second, we require that the prover commits to the accepting response using a statistically binding commitment, but this means that there is only one accepting witness and WI becomes meaningless. We next explain how to address these obstacles.

To address the first obstacle, we consider the secret state that the verifier keeps and is used to render the verdict of acceptance on the prover's response. In our new protocol, the verifier will send, along with its delegation query, its random tape in an encoded form. This encoded form should allow to apply the functionality of the prover under the encoding and send the encoded result back to the verifier. To this end, we present an abstraction that we call *private remote evaluation scheme*, which can be thought of as a one-time non compact homomorphic encryption

scheme with malicious circuit privacy. We show that this primitive can be constructed using garbled circuits and using an oblivious transfer protocol with security against malicious receivers (the same assumption is required for the WI proof system that we need to use). Given the verifier's random tape encoded in this way, the prover can "homomorphicly" check that indeed applying the verifier's query generation on the encoded random tape results in the query string sent by the verifier, and that the prover's response to this query string will result in the verifier accepting. The prover will perform this operation on the encoded random tape (note that the expected output should always be an encoding of 1) and prove in WI that the resulting encoding was indeed generated using the aforementioned operation. Since our encoding scheme is circuit-private, the verifier will not learn anything from the encoding itself (since it is just an encoding of 1), but the WI proof will guarantee that indeed the prover committed to a message that would have made the verifier of the original protocol to accept.

Let us now specify the properties of the two message WI protocol that is required for this approach to go through. First of all, we notice that we need a protocol with *adaptive soundness*, i.e. soundness holds even against a prover that chooses the statement to be proven after seeing the verifier's first message.

Second, we need to address the aforementioned vacuousness of the standard notion of WI when proving with respect to a committed value. This is resolved by resorting to the notion of *strong WI*, which considers two distributions over instance-witness pairs, and requires that if the instance components of the two distributions are computationally indistinguishable, then the verifier cannot distinguish which instance-witness pair was used to generate the proof. Indeed, the recently proposed protocol of Jain et al. [JKKR17] has the required properties (in the delayed input setting), under the assumption that a quasi-poly secure OT scheme exists (we refer to Section 4 for details, and in particular to Theorem 4.1).

Lastly, we require extractability, namely being able to extract the committed response to the delegation protocol in case the WI protocol accepted. However, since the prover only sends a single message, we cannot get extractability under standard assumptions. We therefore rely on complexity leveraging, and extract the prover answer by brute-force breaking the hiding of the commitment scheme. This means that in order for soundness to hold, we need all components other than the commitment scheme to be secure even in the presence of this brute-force extractor, i.e. to have super-polynomial security. This way, we can scale down the hardness of the commitment scheme and allow it to be broken while leaving the other building blocks secure.

2 Preliminaries

Throughout this manuscript, we let λ denote the security parameter.

2.1 Local Satisfiability

In what follows, we define the notion of local satisfiability for circuits, using the formalism from [PR14].

For our needs, we consider any polynomial-time computable function $n = n(\lambda)$, and any polynomial-time computable function $m = m(n)$. We think of n as the instance length, and we think of m as the witness length. For any $\lambda \in \mathbb{N}$, we consider a circuit

$$\mathcal{C}_n : \{0,1\}^n \times \{0,1\}^m \to \{0,1\},$$

where $n = n(\lambda)$. In what follows, we denote by V_n the set of variables corresponding to the wires of $\mathcal{C}_n$.

Definition 2.1 (Local Assignment Generator [PR14])**.** *Fix a polynomial-time computable function $n = n(\lambda)$. A (k, ϵ)-local assignment generator* Assign *for the circuit family $\{\mathcal{C}_n\}_{\lambda\in\mathbb{N}}$ (described above) with corresponding inputs $\{x_n\}_{\lambda\in\mathbb{N}}$ (where $x_n \in \{0,1\}^n$) is a probabilistic algorithm that takes as input a security parameter in unary 1^λ and a set of at most $k(\lambda)$ queries $Q_n \subseteq V_n$ (i.e., $|Q_n| \leq k(\lambda)$), and outputs an assignment $\mathsf{a} : Q_n \to \{0,1\}$, such that the following two properties hold.*

- **Everywhere Local Consistency.** *For every $\lambda \in \mathbb{N}$, every set $Q_n \subseteq V_n$ such that $|Q_n| \leq k(\lambda)$, with probability $1 - \epsilon(\lambda)$ over a draw*

$$\mathsf{a} \leftarrow \mathsf{Assign}(1^\lambda, Q_n) \ ,$$

  the assignment a is locally consistent with the circuit $\mathcal{C}_n$ on the input x_n. That is, for every gate g, with input wires $q_1, q_2 \in Q_n$ and with an output wire $q_3 \in Q_n$, it holds that the assignment $\mathsf{a}(q_1), \mathsf{a}(q_2), \mathsf{a}(q_3)$ satisfies the gate g (with probability $1 - \epsilon(\lambda)$). Moreover, if q is the i'th input wire (for $i \in [n]$) then $\mathsf{a}(q) = x_i$, and if q is the output wire then $\mathsf{a}(q) = 1$ (with probability $1 - \epsilon(\lambda)$).

- **No-signaling.** *For every polynomial size distinguisher $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$ and every sets $Q'_\lambda \subseteq Q_\lambda \subseteq V_\lambda$ such that $|Q_\lambda| \leq k(\lambda)$ there exists a negligible function* $\mathrm{negl}(\lambda)$ *such that*

$$\left| \Pr_{\mathsf{a}\leftarrow\mathsf{Assign}(1^\lambda, Q_\lambda)} \left[\mathcal{D}(\mathsf{a}[Q'_\lambda]) = 1\right] - \Pr_{\mathsf{a}'\leftarrow\mathsf{Assign}(1^\lambda, Q'_\lambda)} \left[\mathcal{D}(\mathsf{a}') = 1\right] \right| = \mathrm{negl}(\lambda) \ .$$

We say that Assign *is a k-local assignment generator if it is a (k, negl)-local assignment generator for some negligible function* negl.

2.2 Single-Round Argument with Local Satisfiability

The following theorem was proven in [BHK17], based on techniques developed in [KRR13,KRR14].

Theorem 2.1 ([BHK17])**.** *Fix any polynomial-time computable function $m = m(n)$, and any circuit family $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ such that $\mathcal{C}_n : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$. Let M be a (possibly non-uniform) Turing machine that on input 1^n runs in time $\mathsf{T} = \mathsf{T}(n) \leq \mathrm{poly}(|\mathcal{C}_n|)$ and outputs a description of the circuit $\mathcal{C}_n$. Assume the existence of a computational* PIR *scheme (with polynomial security). Then there exists a (succinct) single-round (2-message) argument (P, V), such that for any security parameter $\lambda \in [\log \mathsf{T}, \mathsf{T}]$, and any locality parameter $k \in [\lambda, \mathsf{T}]$, the following holds.*

1. **Efficiency.** *The protocol (P, V) has the following efficiency guarantees:*

   (a) *The communication complexity of (P, V), on input $(1^\lambda, M, x, \mathsf{T})$, is $k \cdot \mathrm{poly}(\lambda)$.*[3]

[3] We think of M and $\mathsf{T} = \mathsf{T}(|x|)$ as part of the input, even though they are determined in advance. The fixing of M is essentially without loss of generality since we can always fix M to be the universal Turing machine, and append the actual Turing machine to the input x. Moreover, the fixing of T is also without loss of generality, since one can consider the (fixed) time bounds $\mathsf{T}_i = 2^i$ for every $i \in [\lambda]$, and use the scheme corresponding to the relevant T_i (a similar trick was used in [BHK17], and we refer the reader to [BHK17] for details).

(b) *The runtime of V on input $(1^\lambda, M, x, \mathsf{T})$ is $\tilde{O}(|M| + n) + k \cdot \mathrm{poly}(\lambda)$, where $|M|$ denotes the size of the non-uniform advice of M.*

(c) *The runtime of P, given a witness w such that $\mathcal{C}_n(x, w) = 1$, is $\mathrm{poly}(\mathsf{T})$.*

2. **Perfect Completeness.** *For every $(1^\lambda, M, x, \mathsf{T})$ as above, and for every satisfying assignment w such that $\mathcal{C}_n(x, w) = 1$,*

$$\Pr\left[(P(w), V)(1^\lambda, M, x, \mathsf{T}) = 1\right] = 1,$$

where the probability is over the random coin tosses of V.

3. **Local Soundness.** *Consider an "augmented" version of $\mathcal{C}_n(x, \cdot)$, denoted by $\widetilde{C}_n(x, \cdot)$, which contains, in addition to the original circuit, a Merkle hash-tree for each layer of $C_n(x, \cdot)$. The Merkle hash requires a collision-resistant hash function and we note that since we assume a secure PIR scheme, a collision-resistant hash is implied.*[4]

For every constant $c \in \mathbb{N}$ there exists a probabilistic oracle machine Assign_c, running in time $\mathrm{poly}(k, \lambda)$, such that if there exists a PPT *cheating prover P^* that on input 1^λ generates $x \in \{0,1\}^n$, where $\mathsf{T}(n) \in [\lambda, 2^\lambda]$, and for infinitely many $\lambda \in \mathbb{N}$,*

$$\Pr\left[(P^*, V)(1^\lambda, M, x, \mathsf{T}) = 1\right] \geq \frac{1}{\lambda^c},$$

then for all such $\lambda \in \mathbb{N}$, $\mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, \cdot)$ is a k-local assignment generator for $\widetilde{C}_n(x, \cdot)$.

Moreover, there exists a polynomial $\ell = \ell(\lambda)$ (that does not depend on $\{\mathcal{C}_n\}$), such that any *k-local assignment generator for $\widetilde{C}_n(x, \cdot)$, with $k \geq m \cdot \ell(\lambda)$, satisfies that on input the set of variables corresponding to the witness wires, outputs a valid witness for x with overwhelming probability.*

3 Succinct Single-Round Delegation for Monotone NP

We prove that the very protocol from Theorem 2.1, with $k = m \cdot \ell(\lambda) + 1$, has the standard soundness guarantee (and even a proof-of-knowledge guarantee) when applied to any circuit $\mathcal{C}$ that takes as input $(x_1, \ldots, x_N) \in (\{0,1\}^n)^N$, and a corresponding "witness string" $(w_1, \ldots, w_N) \in (\{0,1\}^m)^N$, and first computes $(b_1, \ldots, b_N)$, where each b_i is the output of an **NP** verification circuit on input (x_i, w_i), and then applies a *monotone* formula on $(b_1, \ldots, b_N)$. Our proof does not rely on any specific property of the protocol. Instead, we show that for such circuits, the local soundness guarantee (from Item 3) implies the standard soundness guarantee, and more specifically, we show how to convert a local assignment generator into an extractor that extracts a valid witness.

Fix an **NP** language L, and let $\mathcal{R}_L$ denote the corresponding **NP** relation. For any x, we let $\mathcal{R}_x$ be a Boolean circuit such that $\mathcal{R}_x(w) = 1$ if and only if $(x, w) \in \mathcal{R}_L$. We denote the length of x by $n = |x|$ and the length of a corresponding witness w by $m = m(n) = |w|$.

Fix any $N : \mathbb{N} \to \mathbb{N}$. For every n, let $N = N(n)$, and let

$$\mathcal{F}_N : \{0,1\}^N \to \{0,1\}$$

[4] We specifically use the augmentation as in [BHK17] as opposed to the different information theoretic augmentation used in [KRR14].

be any monotone formula. Namely, $\mathcal{F}_N$ is a Boolean formula with only AND and OR gates. Without loss of generality, we think of $\mathcal{F}_N$ as a (monotone) circuit of depth $D = O(\log |\mathcal{F}_N|)$, this is without loss of generality since formulas (and in particular monotone formulas) can be balanced (and in particular maintain monotonicity).[5] Consider the Boolean circuit

$$\mathcal{C}_n : (\{0,1\}^n)^N \times (\{0,1\}^m)^N \to \{0,1\},$$

that for $N = N(n)$, takes as input a pair $\mathbf{x} = (x_1, \ldots, x_N) \in (\{0,1\}^n)^N$ and $\mathbf{w} = (w_1, \ldots, w_N) \in (\{0,1\}^m)^N$, and outputs

$$\mathcal{C}_n(\mathbf{x}, \mathbf{w}) \triangleq \mathcal{F}_N(\mathcal{R}_{x_1}(w_1), \ldots, \mathcal{R}_{x_N}(w_N)).$$

Let M be a (possibly non-uniform) Turing machine that on input 1^n outputs the description of the circuit $\mathcal{C}_n$, and denote by $\mathsf{T} = \mathsf{T}(n) \leq \text{poly}(|\mathcal{C}_n|)$ the runtime of M on input 1^n.

Theorem 3.1. *Any protocol satisfying Theorem 2.1, with locality parameter $k = m \cdot \ell(\lambda) + 1$ (where ℓ is the polynomial from Item 3 in Theorem 2.1), satisfies the following proof-of-knowledge guarantee:*

For every constant $c \in \mathbb{N}$ there exists a PPT *oracle machine $\mathcal{E}_c$ such that if there exists a non-uniform* PPT *cheating prover P^*, that on input 1^λ generates $\mathbf{x} = \mathbf{x}(\lambda) \in (\{0,1\}^n)^N$, where $n = n(\lambda)$, such that for infinitely many $\lambda \in \mathbb{N}$,*

$$\Pr\left[(P^*, V)(1^\lambda, M, \mathbf{x}, \mathsf{T}) = 1\right] \geq \frac{1}{\lambda^c},$$

then for these λ's,

$$\Pr[\mathcal{E}_c^{P^*}(1^\lambda, M, \mathbf{x}, \mathsf{T}) = \mathbf{w} \text{ s.t. } \mathcal{C}(\mathbf{x}, \mathbf{w}) = 1] = 1 - \text{negl}(\lambda).$$

Proof. Consider the (succinct) single-round protocol (P, V) given by Theorem 2.1 with $k = m \cdot \ell(\lambda) + 1$, and consider any non-uniform PPT cheating prover P^* that on input 1^λ generates $\mathbf{x} = \mathbf{x}(\lambda) \in (\{0,1\}^n)^N$, where $n = n(\lambda)$, such that for infinitely many $\lambda \in \mathbb{N}$,

$$\Pr\left[(P^*, V)(1^\lambda, M, \mathbf{x}, \mathsf{T}) = 1\right] \geq \frac{1}{\lambda^c}, \tag{1}$$

for some constant $c > 0$.

By the local soundness property of (P, V) (see Item 3), there exists a probabilistic oracle machine Assign_c, running in time $\text{poly}(\lambda, k)$, such that $\mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, \cdot)$ is a k-local assignment generator for the circuit $\mathcal{C}_n(\mathbf{x}, \cdot)$, and for its "augmented" version, denoted by $\widetilde{\mathcal{C}}_n(\mathbf{x}, \cdot)$.

The input of $\widetilde{\mathcal{C}}_n$, similarly to the input of $\mathcal{C}_n$, consists of a pair $(\mathbf{x}, \mathbf{w})$, where $\mathbf{x} = (x_1, \ldots, x_N) \in (\{0,1\}^n)^N$ and $\mathbf{w} = (w_1, \ldots, w_N) \in (\{0,1\}^m)^N$. We think of $\mathbf{x}$ as the input and think of $\mathbf{w}$ as the witness.

For every $i \in [N]$, we denote by Q_i the set of $m = m(n)$ wires corresponding to the i'th witness w_i, and we denote by q_{out_i} the output wire of $\mathcal{R}_{x_i}$.

Recall that $\mathcal{C}$ is of the form

$$\mathcal{C}_n(\mathbf{x}, \mathbf{w}) \triangleq \mathcal{F}_N(\mathcal{R}_{x_1}(w_1), \ldots, \mathcal{R}_{x_N}(w_N)).$$

[5] For the sake of completeness, we outline the proof in the monotone case. We use the combinatorial property that any tree has an edge that splits it in an approximately balanced manner. Applying to a formula $\mathcal{F}$ we get two formulas A, B s.t. $|A|, |B| \geq |\mathcal{F}|/3$ and $\mathcal{F} = A(B)$. Balance A, B recursively to obtain A', B' and finally output $\mathcal{F}' = A'(0) \vee (A'(1) \wedge B)$. Monotonicity guarantees that $\mathcal{F}' \equiv \mathcal{F}$ and balance follows by induction.

The augmentation $\widetilde{\mathcal{C}}_n(\mathbf{x},\cdot)$ (from Theorem 2.1) has the property that it contains the circuit

$$\mathcal{F}_N\left(\widetilde{\mathcal{R}}_{x_1}(\cdot),\ldots,\widetilde{\mathcal{R}}_{x_N}(\cdot)\right)$$

as a sub-circuit, and the output wire of $\mathcal{F}_N\left(\widetilde{\mathcal{R}}_{x_1}(\cdot),\ldots,\widetilde{\mathcal{R}}_{x_N}(\cdot)\right)$ is the output wire of $\widetilde{\mathcal{C}}_n(\mathbf{x},\cdot)$. Therefore the k-local assignment generator Assign_c for $\widetilde{\mathcal{C}}_n$ is, in particular, a k-local assignment generator for

$$\mathcal{F}_N\left(\widetilde{\mathcal{R}}_{x_1}(\cdot),\ldots,\widetilde{\mathcal{R}}_{x_N}(\cdot)\right).$$

Let

$$p \triangleq 4\cdot 2^D\cdot\lambda, \tag{2}$$

where recall that $D = O(\log|\mathcal{F}_N|)$ denotes the depth of $\mathcal{F}_N$. The PPT oracle machine $\mathcal{E}^{P^*} = \mathcal{E}_c^{P^*}$ does the following for each $i\in[N]$:

1. For every $j\in[p]$, compute $(w_{i,j},b_{i,j})\leftarrow \mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, Q_i\cup\{q_{\mathrm{out}_i}\})$.
2. If there exists $j\in[p]$ such that $b_{i,j}=1$ then set $w_i = w_{i,j}$ (for the first j such that $b_{i,j}=1$) and set $b_i = 1$. Otherwise, set $w_i=\perp$ and set $b_i = 0$.

Let

$$\mathbf{b} = (b_1,\ldots,b_N)\in\{0,1\}^N.$$

We prove that for every λ for which Equation (1) holds,

$$\Pr[\mathcal{F}_N(b_1,\ldots,b_N)=1]\geq 1/2, \tag{3}$$

where the probability is over the randomness used by $\mathcal{E}^{P^*}$ to generate $(b_1,\ldots,b_N)$. This is sufficient, since we can run the extractor $\mathcal{E}^{P^*}$ λ times, and thus with overwhelming probability $(1-2^{-\lambda})$ extract $w_1,\ldots,w_N$, for which the corresponding $(b_1,\ldots,b_N)$ satisfies $\mathcal{F}_N(b_1,\ldots,b_N)=1$.

To this end, fix λ for which Equation (1) holds. Denote by V' the set of all the wires in $\widetilde{\mathcal{C}}_n$ corresponding to $\mathcal{F}_n$, and partition V' into D disjoint sets $V' = V'_1\cup\ldots\cup V'_D$, where for every $i\in[D]$, the set of wires V'_i corresponds to the i'th layer of $\mathcal{F}_N$. Denote by W the width of each layer of $\mathcal{F}_n$.[6]

Let $\{v'_{i,j}\}_{i\in[D],j\in[W]}$ denote the values of all the wires in V' corresponding to the input $(b_1,\ldots,b_N)$. To prove Equation (3) we need to prove that

$$\Pr[v'_{D,1}=1]\geq 1/2,$$

where the probability is over the randomness used by $\mathcal{E}^{P^*}$ to generate $(b_1,\ldots,b_N)$.

We start by proving the following two claims.

Claim 3.1.1. *For every $i\in[N]$, if*

$$\Pr\left[b=1 \text{ where } (w,b)\leftarrow\mathsf{Assign}_c(1^\lambda,Q_i\cup\{q_{\mathrm{out}_i}\})\right]\geq\frac{1}{4\cdot 2^D} \tag{4}$$

then

$$\Pr[b_i=1] = 1-\mathrm{negl}(\lambda).$$

[6] We assume without loss of generality that all the layers of $\mathcal{F}_n$ have the same width.

Proof of Claim 3.1.1. Fix $i \in [N]$ such that Equation (4) holds. Recall that (by definition) $b_i = 0$ if and only if for every $j \in [p]$ it holds that $b_{i,j} = 0$ where

$$(w_{i,j}, b_{i,j}) \leftarrow \mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, Q_i \cup \{q_{\mathrm{out}_i}\}).$$

By Equation (4), $\Pr[b_{i,j} = 0] \leq 1 - \frac{1}{4 \cdot 2^D}$. Thus,

$$\Pr[b_j = 0] \leq \left(1 - \frac{1}{4 \cdot 2^D}\right)^p = \left(1 - \frac{1}{4 \cdot 2^D}\right)^{4 \cdot 2^D \cdot \lambda} = \mathrm{negl}(\lambda),$$

as desired. □

Claim 3.1.2. *For every $\lambda \in \mathbb{N}$ and for every $i \in [N]$,*

$$\Pr\left[(w_i, 1) \leftarrow \mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, Q_i \cup \{q_{\mathrm{out}_i}\}) \bigwedge (\mathcal{R}_{x_i}(w_i) = 0)\right] = \mathrm{negl}(\lambda).$$

Proof of Claim 3.1.2. Suppose for the sake of contradiction that there exists a polynomial poly such that for infinitely many $\lambda \in \mathbb{N}$ there exists $i \in [N]$ such that

$$\Pr\left[(w_i, 1) \leftarrow \mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, Q_i \cup \{q_{\mathrm{out}_i}\}) \bigwedge (\mathcal{R}_{x_i}(w_i) = 0)\right] \geq \frac{1}{\mathrm{poly}(\lambda)}. \tag{5}$$

We construct the following $(k-1)$-local assignment generator Assign' for the circuit $\widetilde{R}_{x_i}$.

As above, we denote by q_{out_i} the output wire of $\widetilde{\mathcal{R}}_{x_i}$. We denote by V_i the set of wires in the circuit $\widetilde{\mathcal{R}}_{x_i}$. The assigner Assign' takes as input $(1^\lambda, Q)$ where $Q \subseteq V_i$ and $|Q| \leq k-1$, and does the following:

1. Set $\ell = 1$.
2. Sample $(\mathsf{a}_\ell, b_\ell) \leftarrow \mathsf{Assign}_c^{P^*_\lambda}(1^\lambda, Q \cup \{q_{\mathrm{out}_i}\})$.
3. If $b_\ell = 1$ then output a_ℓ. Otherwise, if $b = 0$ then set $\ell = \ell + 1$.
4. If $\ell < \lambda \cdot \mathrm{poly}(\lambda)$ then goto Item 2. Else, output $\perp$.

We next argue that Assign' is indeed a $(k-1)$-local assignment generator for the circuit $\widetilde{\mathcal{R}}_{x_i}$. To this end, we need to prove that it is computational no-signaling and that it is locally consistent. The computational no-signaling condition follows immediately from the fact that Assign_c is computational no-signaling. As for local consistency, first note that Equation (5) implies (in particular) that

$$\Pr\left[(w_i, 1) \leftarrow \mathsf{Assign}_c^{P^*_\lambda}\left(1^\lambda, Q_i \cup \{q_{\mathrm{out}_i}\}\right)\right] \geq \frac{1}{\mathrm{poly}(\lambda)}.$$

This, together with the no-signaling requirement, implies that Assign outputs $\perp$ only with negligible probability. Thus, the local consistency of Assign' follows from that of Assign_c. Moreover, by definition, Assign' always assigns the output wire the value 1, as desired.

Thus, Assign' is indeed a $(k-1)$-local assignment generator for $\widetilde{\mathcal{R}}_{x_i}$, which together with the fact that $k - 1 \geq m \cdot \ell(\lambda)$, and with Item 3 of Theorem 2.1, implies that indeed $\mathsf{Assign}'(1^\lambda, Q_i)$

outputs a valid witness with overwhelming probability. This is in contradiction to Equation (5), which together with the definition of Assign', implies that there exists a polynomial poly such that

$$\Pr\left[w_i \leftarrow \mathsf{Assign}'(1^\lambda, Q_i) \bigwedge (\mathcal{R}_{x_i}(w_i) = 0)\right] \geq \frac{1}{\mathrm{poly}(\lambda)}.$$

□

In what follows, fix

$$\delta \triangleq \frac{1}{4 \cdot 3^D}.$$

We next argue that for every $i \in \{0, 1, \ldots, D\}$ and every $j \in [W]$,

$$\Pr[v_{i,j} < \mathsf{a}(V'_{i,j})] < \frac{2^i}{4 \cdot 2^D} + 3^i\delta \tag{6}$$

where $V'_{i,j}$ is the variable corresponding to the j'th wire in i'th layer of $\mathcal{F}_n$, and where $\mathsf{a}(V'_{i,j}) \leftarrow \mathsf{Assign}_c(1^\lambda, V_{i,j})$. This suffices since it implies that indeed

$$\Pr[v_{D,1} = 0] \leq \Pr[v_{D,1} < \mathsf{a}(V'_{D,1})] < \frac{2^D}{4 \cdot 2^D} + 3^D\delta = \frac{1}{4} + \frac{1}{4} = \frac{1}{2},$$

as desired.

We prove Equation (6) by induction on i, starting with $i = 0$.

The Induction Base. For $i = 0$, note that by definition, $v_{0,j} = b_j$ for every $j \in [N]$. Denote by E the event that

$$\Pr\left[b = 1 \text{ where } (w, b) \leftarrow \mathsf{Assign}_c(1^\lambda, Q_j \cup \{q_{\mathrm{out}_j}\})\right] \geq \frac{1}{4 \cdot 2^D},$$

where the probability is over the random coin tosses of Assign_c. Then,

$$\begin{aligned}
&\Pr[v_{0,j} < \mathsf{a}(V'_{0,j})] = \\
&\Pr[b_j < \mathsf{a}(V'_{0,j})] = \\
&\Pr[b_j < \mathsf{a}(V'_{0,j}) \mid E] \cdot \Pr[E] + \Pr[b_j < \mathsf{a}(V'_{0,j}) \mid \neg E] \cdot \Pr[\neg E] \leq \\
&\Pr[b_j = 0 \mid E] + \Pr[\mathsf{a}(V'_{0,j}) = 1 \mid \neg E] \leq \\
&\mathrm{negl}(\lambda) + \Pr[\mathsf{a}(V'_{0,j}) = 1 \mid \neg E] \leq \\
&\mathrm{negl}(\lambda) + \frac{1}{4 \cdot 2^D} \leq \\
&\frac{1}{4 \cdot 2^D} + \delta,
\end{aligned}$$

where the first equation follows from the definition of $v_{0,j}$; the second and third equations follow basic probability theory; the forth equation follows from Claim 3.1.1 (together with the definition of event E); the fifth equation follows from the definition of the event E, and the last equation follows from the fact that $\delta \geq \frac{1}{\mathrm{poly}(\lambda)}$ (for some large enough polynomial poly).

The Induction Step. Suppose Equation (6) holds for $i \in [D-1]$ and we will prove that it holds for $i+1$. To this end, fix any $j \in [w]$ and consider the two children V'_{i,j_1} and V'_{i,j_2} of $V'_{i+1,j}$, where $j_1, j_2 \in [w]$. By the induction hypothesis

$$\Pr[v_{i,j_1} < \mathsf{a}(V'_{i,j_1})] < \frac{2^i}{4 \cdot 2^D} + 3^i\delta$$

and

$$\Pr[v_{i,j_2} < \mathsf{a}(V'_{i,j_2})] < \frac{2^i}{4 \cdot 2^D} + 3^i\delta$$

By a straightforward union bound, this implies that

$$\Pr\left[\left(v_{i,j_1} < \mathsf{a}(V'_{i,j_1})\right) \cup \left(v_{i,j_2} < \mathsf{a}(V'_{i,j_2})\right)\right] < \frac{2^{i+1}}{4 \cdot 2^D} + 2 \cdot 3^i\delta.$$

This, together with the no-signaling and local consistency properties, implies that

$$\Pr[v_{i+1,j} < \mathsf{a}(V'_{i+1,j})] < \frac{2^{i+1}}{4 \cdot 2^D} + 2 \cdot 3^i\delta + \mathrm{negl}(\lambda) \leq \frac{2^{i+1}}{4 \cdot 2^D} + 3^{i+1}\delta,$$

as desired, where the latter inequality follows from the fact that δ' is negligible whereas $3^{i+1}\delta \geq \frac{1}{\mathrm{poly}(\lambda)}$ (for a large enough polynomial poly).

□

4 Delegation with Secrecy

In this section we present our general transformation for converting any 2-message argument system into a 2-message witness indistinguishable one with only modest increase in communication complexity.

4.1 Preliminaries

Our transformation makes use of several cryptographic building blocks, which we present below.

Garbled Circuits. We rely on a decomposable randomized encoding scheme. For the sake of concreteness we consider garbled circuits.

Definition 4.1 (Garbled Circuits)**.** *A garbling scheme consists of a tuple of three algorithms* $(\mathsf{Garble}, \mathsf{GCEval}, \mathsf{GCSim})$ *where:*

1. $\mathsf{Garble}(1^\lambda, 1^n, 1^m, C)$ *is a PPT algorithm that takes as input the security parameter* λ *and a circuit* $C : \{0,1\}^n \to \{0,1\}^m$*, and outputs a garbled circuit* $\widehat{C}$ *along with input labels* $(\mathsf{lab}_{i,b})_{i\in[n],b\in\{0,1\}}$ *where each label* $\mathsf{lab}_{i,b} \in \{0,1\}^\lambda$*.*

2. $\mathsf{GCEval}(1^\lambda, \widehat{C}, \widehat{\mathsf{lab}})$ *is a deterministic algorithm that takes as input a garbled circuit* $\widehat{C}$ *along with a set of* n *labels* $\widehat{\mathsf{lab}} = (\mathsf{lab}_i)_{i\in[n]}$*, and outputs a string* $y \in \{0,1\}^m$*.*

3. $\mathsf{GCSim}(1^\lambda, 1^{|C|}, 1^n, y)$ *is a* PPT *algorithm that takes as input the security parameter, the description length of* C*, an input length* n *and a string* $y \in \{0,1\}^m$*, and outputs a simulated garbled circuit* $\widetilde{C}$ *and labels* $\widetilde{\mathsf{lab}}$*.*

We often omit the first input to these algorithms (namely, 1^λ) when it is clear from the context. We require that the garbling scheme satisfies two properties:

1. *Correctness: For all circuits C, inputs x, and all $(\widehat{C}, (\mathsf{lab}_{i,b})_{i,b}) \leftarrow \mathsf{Garble}(C, x)$ and $\widehat{\mathsf{lab}} = (\mathsf{lab}_{i,x_i})_{i\in[n]}$, we have that $\mathsf{GCEval}(\widehat{C}, \widehat{\mathsf{lab}}) = C(x)$.*
2. *Simulation Security: For all circuits $C : \{0,1\}^n \to \{0,1\}^m$ and all inputs $x \in \{0,1\}^n$, the following two distributions are computationally indistinguishable:*

$$\left\{(\widehat{C}, \widehat{\mathsf{lab}}) : (\widehat{C}, (\mathsf{lab}_{i,b})_{i,b}) \leftarrow \mathsf{Garble}(C, x), \widehat{\mathsf{lab}} = (\mathsf{lab}_{i,x_i})_{i\in[n]}\right\}$$
$$\stackrel{c}{\approx} \left\{(\widetilde{C}, \widetilde{\mathsf{lab}}) : (\widetilde{C}, \widetilde{\mathsf{lab}}) \leftarrow \mathsf{GCSim}(1^\lambda, 1^{|C|}, 1^n, C(x))\right\} .$$

Oblivious Transfer Secure Against Malicious Receivers. We use a notion of oblivious transfer that has computational security against senders (i.e. receiver privacy) but also (statistical) security against malicious receivers (sender privacy). That is, regardless of the receiver's first message, the sender's response never reveals more than one of its inputs, even to an unbounded adversary.

Definition 4.2 (Two-Message Oblivious Transfer with Statistical Sender Security)**.** *A two-message oblivious transfer is a protocol between two parties, a sender S with messages (m_0, m_1) and receiver $R = (R_1, R_2)$ with a choice bit b, such that R obtains output m_b at the end of the protocol. Specifically, $R_1(b) = R_1(1^\lambda, b)$ outputs (σ, e), where e is the message sent to the receiver and σ is a local state that is kept private. The sender responds with an answer $v = S(1^\lambda, (m_0, m_1), e)$. Finally $R_2(1^\lambda, \sigma, v)$ outputs a message m. We omit the security parameter input to these procedures when it is clear from the context.*

We consider OT that satisfies the following properties:

- **Computational Receiver Security.** *The distributions $R(0)$ and $R(1)$ are computationally indistinguishable. We sometimes require super-polynomial security, specifically, we say that the OT scheme is T-receiver secure if $T \cdot \mathrm{poly}(\lambda)$-size distinguishers have advantage less than $\frac{\mathrm{negl}(\lambda)}{T}$.*
- **Statistical Sender Security.** *For all λ and for all $e^* \in \{0,1\}^*$ there exists a bit b^* such that the distributions $S(1^\lambda, (m_0, m_1), e^*)$ and $S(1^\lambda, (m_{b^*}, m_{b^*}), e^*)$ are statistically indistinguishable. It would sometimes be convenient to think about b^* as produced by a computationally unbounded procedure Ext so that $b^* = \mathsf{Ext}(1^\lambda, e^*)$ (we sometimes omit 1^λ when it is clear from the context).*

Oblivious transfer protocols satisfying these definitions have been introduced based on assumptions such as DDH, QR and DCR [NP01, Kal05, HK07].

Delayed-Input Interactive Protocols and Strong Witness Indistinguishability. A ℓ-message delayed-input interactive protocol (P, V) for deciding an **NP** language L with associated relation R_L proceeds in the following manner:

- At the beginning of the protocol, P and V receive the size of the instance and the security parameter, denoted by n and λ, respectively, and execute the first $\ell - 1$ messages.

- Before sending the last message, P receives as input a pair $(x, w) \in R_L$, where $|x| = n$, and V receives x. Upon receiving the last message from P, V outputs 1 or 0.

An execution of (P, V) with instance x and witness w is denoted as $\langle P, V\rangle(x, w)$. Whenever clear from context, we also use the same notation to denote the output of V.

A ℓ-message delayed-input interactive argument for a language L must satisfy the standard notion of completeness (in the delayed-input setting) as well as *adaptive soundness*, where the soundness requirement holds even against malicious PPT provers who choose the statement adaptively, depending upon the first $\ell - 1$ messages of the protocol.

Definition 4.3 (Delayed-Input Interactive Arguments)**.** *A ℓ-message delayed-input interactive protocol (P, V) for deciding a language L is an interactive argument for L if it satisfies the following properties:*

- **Adaptive Completeness:** *For every $(x, w) \in R_L$ chosen adaptively after $\ell - 1$ rounds of interaction,*
$$\Pr\left[\langle P, V\rangle(x, w) = 1\right] = 1,$$
where the probability is over the random coins of P and V.

- **Adaptive Soundness:** *For every (non-uniform) PPT prover P^* that chooses $n = \text{poly}(\lambda)$ and chooses $x \in \{0,1\}^n \setminus L$ adaptively, depending upon the first $\ell - 1$ messages,*
$$\Pr\left[\langle P^*, V\rangle(x) = 1\right] = \text{negl}(\lambda),$$
where the probability is over the random coins of V.

Definition 4.4 ((Strong) Witness Indistinguishability)**.** *Let $n = n(\lambda) \leq \text{poly}(\lambda)$. An interactive argument (P, V) for a language L is* strong witness indistinguishable *(which we denote* sWI*) if for every pair of distributions over pairs $\{(\mathcal{X}_{1,n(\lambda)}, \mathcal{W}_{1,n(\lambda)})\}_{\lambda\in\mathbb{N}}$ and $\{(\mathcal{X}_{2,n(\lambda)}, \mathcal{W}_{2,n(\lambda)})\}_{\lambda\in\mathbb{N}}$ supported over R_L, for which the distributions $\{\mathcal{X}_{1,n(\lambda)}\}_{\lambda\in\mathbb{N}}$ and $\{\mathcal{X}_{2,n(\lambda)}\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable, for every PPT verifier V^*, and for every (non-uniform) PPT distinguisher $\mathcal{D}$,*

$$\left| \Pr_{(x,w)\leftarrow(\mathcal{X}_{1,n(\lambda)},\mathcal{W}_{1,n(\lambda)})} \left[\mathcal{D}(x, \text{View}_{V^*}[\langle P, V^*\rangle(x, w)] = 1\right] \right.$$

$$\left. - \Pr_{(x,w)\leftarrow(\mathcal{X}_{2,n(\lambda)},\mathcal{W}_{2,n(\lambda)})} \left[\mathcal{D}(x, \text{View}_{V^*}[\langle P, V^*\rangle(x, w)] = 1\right] \right| \leq \text{negl}(\lambda) \ .$$

Standard (as opposed to strong) witness indistinguishability (which we denote simply by WI*) only requires that the above holds for singleton distributions, which is equivalent (due to the indistinguishability condition) to defining a deterministic sequence of input and witness pairs*

$$\{(x_{n(\lambda)}, w_{1,n(\lambda)}, w_{2,n(\lambda)})\}_{\lambda\in\mathbb{N}}.$$

In delayed input *strong witness indistinguishability, the above is only required to hold with respect to PPT verifiers V^* who obtain the instance together with the last prover message in the protocol (i.e., who generate their messages obliviously of x). Note that this notion is vacuous for standard (non-strong)* WI*.*

Theorem 4.1 ([JKKR17]). *For any $T = \lambda^{\omega(1)}$, assume the existence of a non-interactive statistically binding commitment scheme, that is hiding against poly-size adversaries, but where the hiding property can be broken by* $\mathrm{poly}(T)$ *adversaries, and assume the existence of a* $\mathrm{poly}(T)$*-secure OT scheme as in Definition 4.2. Then there exists a 2-message delayed-input strong WI protocol for every language in* **NP** *such that soundness holds against* $\mathrm{poly}(T)$*-size adversaries, but (strong) WI property holds only against poly-size cheating verifiers.*

Remark 4.1. *The strong WI property can be strengthened to hold against* $\mathrm{poly}(T^*)$*-size cheating verifiers, for any $T^* = T^{o(1)}$. However, this requires assuming that the underlying commitment scheme that can be broken in time* $\mathrm{poly}(T)$*, is secure against* $\mathrm{poly}(T^*)$ *size adversaries.*

4.2 Private Remote Evaluation

Our transformation makes use of a primitive that we call a *private remote evaluation scheme*. Loosely speaking, this can be thought of as a *one-time* non-succinct fully homomoprhic encryption scheme with strong malicious circuit privacy [GHV10, OPP14], which in turn follows the outline of Yao's 2-party 2-round secure function evaluation protocol [Yao82].

Rather than formally defining this primitive, we construct it (using a garbling scheme satisfying Definition 4.1 and using an oblivious transfer protocol satisfying Definition 4.2), and state its properties.

Let $(R = (R_1, R_2), S)$ be an OT scheme that satisfies Definition 4.2 and let $(\mathsf{Garble}, \mathsf{GCEval}, \mathsf{GCSim})$ be a garbling scheme. Our private remote evaluation scheme consists of a tuple of four algorithms $(\mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Sim})$, defined as follows.

- The encoding algorithm Enc takes an input a security parameter 1^λ and a string $x \in \{0,1\}^n$, and outputs an encoded output ψ and a secret state σ. Specifically, for every bit of x, Enc runs $R_1(1^\lambda, x_i)$ to compute the first OT receiver message $\psi^{(i)}$ and the state $\sigma^{(i)}$. It outputs $\psi = \{\psi^{(i)}\}_i$, $\sigma = \{\sigma^{(i)}\}_i$. We sometimes denote by Enc_1 the algorithm that computes Enc and only outputs the ψ component, and we often omit the security parameter from the notation.

- The evaluation algorithm Eval takes as input a circuit $C : \{0,1\}^n \to \{0,1\}^m$ and an encoded input $\psi = \{\psi^{(i)}\}_i$. It creates a garbled circuit $\widehat{C}$ for C with labels $\mathsf{lab}_{i,b}$, and computes the sender response for each OT execution $\psi'^{(i)} = S((\mathsf{lab}_{i,0}, \mathsf{lab}_{i,1}), \psi^{(i)})$. It finally outputs $\psi' = (\{\psi'^{(i)}\}_i, \widehat{C})$.

- The decoding procedure Dec takes as input $\psi' = (\{\psi'^{(i)}\}_i, \widehat{C})$ and $\sigma = \{\sigma^{(i)}\}_i$, and applies the OT receiver protocol to obtain $\mathsf{lab}_i = R_2(\sigma^{(i)}, \psi'^{(i)})$. It finally evaluates the garbled circuit $\widehat{C}$ on $\{\mathsf{lab}_i\}_i$ and outputs the resulting $y \in \{0,1\}^m$.

- For all $1^n, 1^m, 1^c$ representing input, output and circuit size (these inputs are often omitted when they are clear from the context), there exists a simulator

$$\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2) \ ,$$

  such that the following holds. Let Ext be the OT extractor from Definition 4.2. The simulator Sim_1 takes as input a (possibly adversarially chosen) sequence $\psi = \{\psi^{(i)}\}_{i=1}^n$, and runs Ext on each $\psi^{(i)}$ to obtain a bit x_i. Let $x \in \{0,1\}^n$ denote the collection of the extracted bits.

  The simulator Sim_2, takes as input (ψ, x) together with a string $y \in \{0,1\}^m$, it runs in probabilistic polynomial time, and does the following:

1. It runs the PPT garbled circuit simulator GCSim, on input y, to generate simulated circuit and labels of the appropriate size and input-output lengths $\widetilde{C}, \widetilde{\mathsf{lab}}$.
2. It generates simulated sender messages $\{\widetilde{\psi}^{(i)}\} = S((\widetilde{\mathsf{lab}}_i, \widetilde{\mathsf{lab}}_i), x_i)$.
3. It outputs $\widetilde{\psi} = (\{\widetilde{\psi}^{(i)}\}, \widetilde{C})$.

Claim 4.2. *For any $\psi = \{\psi^{(i)}\}_{i=1}^n$ and any circuit $C : \{0,1\}^n \to \{0,1\}^m$, it holds that*

$$\mathsf{Eval}(C, \psi) \stackrel{c}{\approx} \mathsf{Sim}_2(\psi, x, C(x)) \ ,$$

where $x \leftarrow \mathsf{Sim}_1(\psi)$.

Proof. By definition,

$$\mathsf{Eval}(C, \psi) = \left(\left\{ S((\mathsf{lab}_{i,0}, \mathsf{lab}_{i,1}), \psi^{(i)}) \right\}_i, \widehat{C} \right) \ .$$

Since ψ is fixed, then the value $x \leftarrow \mathsf{Sim}_1(\psi)$ is also fixed. It follows from Definition 4.2 that

$$\mathsf{Eval}(C, \psi) \stackrel{s}{\approx} \left(\left\{ S((\mathsf{lab}_{i,x_i}, \mathsf{lab}_{i,x_i}), \psi^{(i)}) \right\}_i, \widehat{C} \right) \ .$$

Now we use the garbled circuit security to argue that

$$\mathsf{Eval}(C, \psi) \stackrel{c}{\approx} \left(\left\{ S((\widetilde{\mathsf{lab}}_i, \widetilde{\mathsf{lab}}_i), \psi^{(i)}) \right\}_i, \widetilde{C} \right) = \mathsf{Sim}_2(\psi, x, C(x)) \ ,$$

where $\widetilde{\mathsf{lab}}, \widetilde{C}$ are produced by the garbled circuit simulator given $y = C(x)$. □

The following claims are immediate from the OT correctness and receiver security.

Claim 4.3 (Correctness)**.** *For every $n = n(\lambda)$ (not necessarily polynomially bounded), every $x \in \{0,1\}^n$, every $C : \{0,1\}^n \to \{0,1\}$, letting $(\psi, \sigma) \leftarrow \mathsf{Enc}(1^\lambda, x)$, $\psi' \leftarrow \mathsf{Eval}(C, \psi)$, $y = \mathsf{Dec}(\sigma, \psi')$, it holds that $y = C(x)$ with probability 1.*

Claim 4.4 (Receiver Privacy.)**.** *For every $n = n(\lambda) \leq \mathrm{poly}(\lambda)$ and every sequences of inputs $x, x' \in \{0,1\}^n$ it holds that $\mathsf{Enc}_1(1^\lambda, x) \stackrel{c}{\approx} \mathsf{Enc}_1(1^\lambda, x')$.*

4.3 Making Single-Round Protocols Witness Indistinguishable

We show how to convert any single-round (2-message) protocol (P, V) with super-polynomial security and perfect completeness into a single-round (2-message) *witness indistinguishable* (WI) protocol, such that if the communication complexity of the original protocol (P, V) is $\mathrm{cc}(n, \lambda)$ then the communication complexity of the resulting WI protocol $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ is $\mathrm{cc}(n, \lambda) + \mathrm{poly}(v(n, \lambda))$, where $v(n, \lambda)$ is the total runtime of the original verifier V, both in generating the query string to be sent to the prover and in verifying the response received by the prover. We use the term *verdict function* to refer to the second step on V, namely the function that takes as input the communication transcript and an internal secret state of the verifier, and outputs whether the verifier accepts or rejects. Our transformation requires that the original protocol (P, V) is sound against super-polynomial time adversaries (as we intend to use complexity leveraging). Our theorem statement follows.

Theorem 4.5. *For any super-polynomial function $T : \mathbb{N} \to \mathbb{N}$, there is a generic transformation that transforms any (privately or publicly verifiable) single-round argument (P, V) for an* **NP** *language L with perfect completeness and soundness against $\mathrm{poly}(T)$-size cheating provers, into a* privately verifiable *witness indistinguishable single-round argument $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ for L with the following properties:*

- **Succinctness.** *If the communication complexity of (P, V) is $\mathrm{cc}(n, \lambda)$, and V has total time complexity $v(n, \lambda)$, then the communication complexity of $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ is*

$$\mathrm{cc}_{\mathrm{WI}}(n, \lambda) \triangleq \mathrm{cc}(n, \lambda) + \mathrm{poly}(\lambda, v(n, \lambda)).$$

- **Completeness.** *For every $x \in L$ and any witness w for x, it holds that $(P_{\mathrm{WI}}(x.w), V_{\mathrm{WI}}(x))$ accepts with probability* 1*.*
- **Soundness.** *$(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ is sound against (non-uniform) cheating provers of size $\mathrm{poly}(T)$.*
- **Witness Indistinguishability.** *$(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ is witness indistinguishable against (non-uniform)* PPT *cheating verifiers (but not against $\mathrm{poly}(T)$-size cheating verifiers, see also Remark 4.2 below).*

This transformation requires the following building blocks:

- *A statistically binding non-interactive commitment scheme* Com *that can be broken in time $\mathrm{poly}(T)$ for all sufficiently large value of λ.*
- *The private remote evaluation scheme $(\mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}, \mathsf{Sim})$, as described in Section 4.2, where the underlying OT scheme has receiver privacy against $\mathrm{poly}(T)$-size adversaries (i.e, Claim 4.4 is satisfied against $\mathrm{poly}(T)$-size adversaries).*
- *A delayed-input single-round (2-message) strong WI (*sWI*) argument system $(P_{\mathrm{sWI}}, V_{\mathrm{sWI}})$ for* **NP***, that is sound against $\mathrm{poly}(T)$ size cheating provers.*

In fact, we will show that our transformation enjoys an even stronger soundness guarantee as described next. There exist black-box non-rewinding, instance preserving (where applicable) $\mathrm{poly}(T)$-time reductions M_1, M_2, M_3, such that for every (possibly inefficient) cheating prover P^*_{WI} it holds that $M_1^{P^*_{\mathrm{WI}}}$ is a cheating prover against the sWI proof system, $M_2^{P^*_{\mathrm{WI}}}$ is a distinguisher for the remote evaluation scheme, and $M_3^{P^*_{\mathrm{WI}}}$ is a cheating prover against the original argument system, and it holds that the sum of advantages of these adversaries in their related game is at least the advantage of P^*_{WI} in the compiled protocol (up to negligible terms).

We note that the resulting WI protocol is only privately verifiable, even if the underlying protocol was publicly verifiable.

Remark 4.2. *One can strengthen the above theorem so that WI holds against any $\mathrm{poly}(T^*)$-size adversaries, for any $T^* = T^{o(1)}$, by relying on a quantified version of Theorem 4.1 (see Remark 4.1), with WI against $\mathrm{poly}(T^*)$-size adversaries. This requires assuming that the underlying commitment scheme* Com*, which can be broken in time $\mathrm{poly}(T)$, is secure against $\mathrm{poly}(T^*)$-adversaries.*

Proof. Consider a language L, time complexity bound T, a protocol (P, V), a private remote evaluation scheme $(\mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}, \mathsf{Sim})$ and a delayed input strong WI argument system $(P_{\text{sWI}}, V_{\text{sWI}})$, all as described in the theorem statement. We denote by (Q, A) the first and second message respectively exchanged in the protocol (P, V).

Let Com be a statistically binding non-interactive commitment scheme that can be broken in time $\text{poly}(T)$, as described in the theorem statement. Such commitment schemes can be constructed from injective one-way functions. We note that for our purposes it is possible to use Naor's two-message commitment scheme from any one-way function [Nao89] since we can allow a message from the receiver to the sender prior to the commitment message, but for the sake of simplicity we will assume that Com is non-interactive. We further assume w.l.o.g that the length of the commitment string is equal to the length of the committed message plus an additive $\text{poly}(\lambda)$ term. This can be achieved generically using "key encapsulation" (committing to a PRG seed and using the PRG output to mask the message).

We show how to convert (P, V) into a 2-message witness indistinguishable argument, denoted by $(P_{\text{WI}}, V_{\text{WI}})$, which preserves the succinctness property of (P, V), as stated in the theorem statement. Since (P, V) is not necessarily publicly verifiable, in order to verify a transcript (Q, A) the verifier may need a private state, which we denote by st. We will assume w.l.o.g that st is simply the random tape of the verifier V. This will allow to check, given some possible query string Q whether Q is the string generated when V starts with random tape st. If this condition holds, we say that st is consistent with Q, we denote this by $\mathsf{st} \models Q$. The resulting protocol $(P_{\text{WI}}, V_{\text{WI}})$ makes use of an underlying (not necessarily succinct) delayed-input strong WI 2-message argument $(P_{\text{sWI}}, V_{\text{sWI}})$ for the **NP** language L', defined as follows:

$$\begin{aligned} L' = \{(1^\lambda, x, Q, c, \mathsf{st}) : \ & \exists (A, r) \text{ s.t.} \\ & \big(\mathsf{st} \not\models Q\big) \ \vee \ \big(c = \text{Com}(A, r) \wedge V(1^\lambda, x, Q, A, \mathsf{st}) = 1\big)\} \ . \end{aligned} \tag{7}$$

Note that every instance where Q is *inconsistent* with st is trivially *in the language*. Intuitively, this is to force witness indistinguishability also against verifiers who produce inconsistent transcripts. This condition will never be relevant for honest verifiers.

In the protocol $(P_{\text{WI}}, V_{\text{WI}})$, the prover will send a commitment to his answer A (as opposed to sending it in the clear, which may reveal information), followed by a proof that the committed value is an accepting answer. However, to generate such a proof he needs to know the verdict function, and thus, needs the verifier's secret state. However, he cannot receive this secret state "in the clear", since that may breech soundness. Instead, the verifier will send the prover an encoding of his secret state st using the private remote evaluation scheme.

We are now ready to define the protocol $(P_{\text{WI}}, V_{\text{WI}})$:

1. On input 1^λ and $x \in \{0,1\}^n$ the verifier does the following:
   (a) Compute $(Q, \mathsf{st}) \leftarrow V(1^\lambda, x)$, where Q is the message to be sent to the prover P, and st is the corresponding secret state of V.
   (b) Compute $(\psi, \sigma) \leftarrow \mathsf{Enc}(\mathsf{st})$.
   (c) Compute $(\text{sWI}_1, \mathsf{st}_{\text{sWI}}) \leftarrow V_{\text{sWI}}(1^\lambda)$.
   Note that the first message sWI_1 is independent of the instance since $(P_{\text{sWI}}, V_{\text{sWI}})$ is a delayed-input 2-message argument (see Definition 4.3).

   Send (Q, sWI_1, ψ) to the prover, and store $(\sigma, \mathsf{st}, \mathsf{st}_{\text{sWI}})$ as the secret state for verification.

2. The prover, on input $(1^\lambda, x, w)$, and given the message $(Q, \mathrm{sWI}_1, \psi)$, does the following:

   (a) Compute $A \leftarrow P(1^\lambda, x, w, Q)$

   (b) Choose a random string $r \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)}$ and compute $c = \mathrm{Com}(A, r)$.

   (c) Define (implicitly since st is not known) $x' = (1^\lambda, x, Q, c, \mathsf{st})$, and $w' = (A, r)$ as its corresponding witness with respect to $R_{L'}$, i.e. $(x', w') \in \mathcal{R}_{L'}$.

   (d) Given ψ, compute $\psi' = \mathsf{Eval}(f, \psi)$ where $f = f_{1^\lambda, x, Q, c, w', \mathrm{sWI}_1}$, is the function that on input st outputs $\mathrm{sWI}_2 \leftarrow P_{\mathrm{sWI}}(1^\lambda, x', w', \mathrm{sWI}_1)$.

   Send (c, ψ') to the verifier.

3. Upon receiving a message (c, ψ') from the prover, and given a secret state $(\sigma, \mathsf{st}, \mathsf{st}_{\mathrm{sWI}})$ the verifier does the following:

   (a) Decrypt the ciphertext ψ', by computing $\mathrm{sWI}_2 \leftarrow \mathsf{Dec}(\sigma, \psi')$.

   (b) Accept if and only if $V_{\mathrm{sWI}}(1^\lambda, x', \mathrm{sWI}_1, \mathrm{sWI}_2, \mathsf{st}_{\mathrm{sWI}}) = 1$, where $x' = (1^\lambda, x, Q, c, \mathsf{st})$.

Succinctness. We first argue that $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ satisfies the succinctness property as in the theorem statement. To do this, we argue that

$$\mathrm{cc}(P_{\mathrm{WI}}, V_{\mathrm{WI}}) = \mathrm{cc}(P, V) + \mathrm{poly}(\lambda) + \mathrm{poly}(\lambda, v(n, \lambda)) \ ,$$

which would immediately imply the required succinctness.

The first additive $\mathrm{poly}(\lambda)$ term is due to the overhead of sending a commitment to the answer A rather than sending A itself (as explained above, we can assume additive overhead w.l.o.g). The second $\mathrm{poly}(\lambda, v(n, \lambda))$ term is an upper bound on the length of ψ'. The value ψ' is the output of applying Eval on a function f of size $v(n, \lambda) + \mathrm{poly}(\lambda) \leq \mathrm{poly}(\lambda, v(n, \lambda))$ (an upper bound on the prover complexity of P_{sWI} when proving $(x', w') \in \mathcal{R}_{L'}$). Verifying that $(x', w') \in \mathcal{R}_{L'}$ can be done in time proportional to the total complexity of V since checking whether $\mathsf{st} \models Q$ is proportional to running the first phase of V, and checking the value of the verdict function is proportional to the second phase. Add to that checking the commitment which is polynomial in $(\lambda, |A|)$. Since Eval introduces a fixed polynomial overhead, its output length is at most $\mathrm{poly}(\lambda, v(n, \lambda))$.

It remains to prove that $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ satisfies the standard completeness and soundness guarantees, and in addition that it is witness indistinguishable.

Completeness. The completeness of $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ follows immediately from the completeness of (P, V), the delayed-input completeness of $(P_{\mathrm{sWI}}, V_{\mathrm{sWI}})$, and the correctness of the underlying private remote evaluation scheme.

Soundness. Consider a cheating prover P^*_{WI} that for any security parameter 1^λ generates $x \in \{0,1\}^n \setminus L$, where $n \leq \mathrm{poly}(\lambda)$, such that for some non-negligible function $\alpha = \alpha(\lambda)$

$$\Pr[\mathsf{Output}_{V_{\mathrm{WI}}}(P^*_{\mathrm{WI}}, V_{\mathrm{WI}})(1^\lambda, x) = 1] \geq \alpha \ . \tag{8}$$

Recall that P^*_{WI}, upon receiving a message $(Q, \mathrm{sWI}_1, \psi)$, where $\psi \leftarrow \mathsf{Enc}_1(\mathsf{st})$, from the verifier, generates a response (c, ψ'). Since Com is a statistically binding commitment scheme that can be broken in $\mathrm{poly}(T)$ time, there exists a $\mathrm{poly}(T)$-time algorithm that given c outputs (A', r') such that $c = \mathrm{Com}(A', r')$.

Define

$$\alpha_1 = \alpha_1(\lambda) \triangleq \Pr[\big(\mathsf{Output}_{V_{\mathrm{WI}}}(P^*_{\mathrm{WI}}, V_{\mathrm{WI}})(1^\lambda, x) = 1\big) \wedge \big(V(1^\lambda, x, Q, A', \mathsf{st}) = 0\big)]. \tag{9}$$

We consider a cheating prover $P^*_{\mathrm{sWI}} = M_1^{P^*_{\mathrm{WI}}}$ (where M_1 is a non-rewinding reduction) that succeeds in breaking the delayed input soundness of $(P_{\mathrm{sWI}}, V_{\mathrm{sWI}})$ with probability α_1. The reduction M_1, takes as input a message sWI_1 from the verifier V_{sWI}, and does the following:

1. Generate $x \leftarrow P^*_{\mathrm{WI}}(1^\lambda)$ using the P^*_{WI} oracle.
2. Compute $(Q, \mathsf{st}) \leftarrow V(1^\lambda, x)$.
3. Compute $(\psi, \sigma) \leftarrow \mathsf{Enc}(\mathsf{st})$.
4. Send $(Q, \mathrm{sWI}_1, \psi)$ to the P^*_{WI} oracle to obtain $(c, \psi') = P^*_{\mathrm{WI}}(Q, \mathrm{sWI}_1, \psi)$. Recall that for an honest P_{WI}, it holds that ψ' decrypts to sWI_2.
5. Let $x' = (1^\lambda, x, Q, c, \mathsf{st})$.
6. Compute $\mathrm{sWI}_2 \leftarrow \mathsf{Dec}(\sigma, \psi')$.
7. Send (x', sWI_2) to the verifier.

Note that it suffices to argue that

$$\Pr[V_{\mathrm{sWI}}(1^\lambda, \mathrm{sWI}_1, (x', \mathrm{sWI}_2), \mathsf{st}_{\mathrm{sWI}}) = 1 \wedge (x' \notin L')] \geq \alpha_1.$$

This follows immediately from Eq. (9), together with the fact that $x' \notin L'$ if and only if $V(1^\lambda, x, Q, A', \mathsf{st}) = 0$, where A' is the value that c commits to, and the fact that

$$V_{\mathrm{WI}}(1^\lambda, x, (Q, \mathrm{sWI}_1, \psi), (c, \psi'), \mathsf{st}) = 1$$

only if

$$V_{\mathrm{sWI}}(1^\lambda, \mathrm{sWI}_1, (x', \mathrm{sWI}_2), \mathsf{st}_{\mathrm{sWI}}) = 1 \ .$$

Note that by Eq. (8) and (9),

$$\Pr[\big(\mathsf{Output}_{V_{\mathrm{WI}}}(P^*_{\mathrm{WI}}, V_{\mathrm{WI}})(1^\lambda, x) = 1\big) \wedge \big(V(1^\lambda, x, Q, A', \mathsf{st}) = 1\big)] \geq \alpha - \alpha_1. \tag{10}$$

We now present poly(T)-time straight line reductions M_2, M_3 converting P^*_{WI} into an adversary $\mathcal{A}$ that breaks the indistinguishability property of the encoding scheme (i.e., breaks Claim 4.4 with respect to a poly(T)-size adversary), and into cheating prover P^* for the underlying 2-message argument (P, V), respectively, so that the sum of the advantages of the resulting adversaries, denoted α_2, α_3 respectively is $\alpha_2 + \alpha_3 \geq \alpha - \alpha_1$. Furthermore, M_3 is also input preserving.

The distinguisher $\mathcal{A} = M_2^{P^*_{\mathrm{WI}}}$ runs as follows.

1. Generate $x \leftarrow P^*_{\mathrm{WI}}(1^\lambda)$.
2. Run the verifier $V(1^\lambda, x)$ to generate (Q, st).

3. Send $(\mathsf{st}, 0^{|\mathsf{st}|})$ as the two messages for the distinguishing advantage, and receive a challenge encoding ψ from the encoding scheme challenger.
4. Generate $(\mathrm{sWI}_1, \mathsf{st}_{\mathrm{sWI}}) \leftarrow V_{\mathrm{sWI}}(1^\lambda)$.
5. Send $(Q, \mathrm{sWI}_1, \psi)$ to the P^*_{WI} oracle to obtain $(c, \psi') = P^*_{\mathrm{WI}}(Q, \mathrm{sWI}_1, \psi)$.
6. Run in time $\mathrm{poly}(T)$ to find (A', r') such that $c = \mathrm{Com}(A', r')$.
7. Return $V(1^\lambda, x, Q, A', \mathsf{st})$.

The cheating prover $P^* = M_3^{P^*_{\mathrm{WI}}}$ is as follows.

1. Upon receiving a security parameter 1^λ, generate $x \leftarrow P^*_{\mathrm{WI}}(1^\lambda)$.
2. Upon receiving a message Q from the verifier $V(1^\lambda, x)$, do the following:
   (a) compute $(\mathrm{sWI}_1, \mathsf{st}_{\mathrm{sWI}}) \leftarrow V_{\mathrm{sWI}}(1^\lambda)$.
   (b) Generate $\psi \leftarrow \mathsf{Enc}_1(0^{|\mathsf{st}|})$ (while st itself is unknown, its length is specified by the protocol, we recall that Enc_1 is the algorithm that executes Enc but only outputs the ψ component, see Section 4.2).
   (c) Send $(Q, \mathrm{sWI}_1, \psi)$ to the P^*_{WI} oracle to obtain $(c, \psi') = P^*_{\mathrm{WI}}(Q, \mathrm{sWI}_1, \psi)$.
3. Run in time $\mathrm{poly}(T)$ to find (A', r') such that $c = \mathrm{Com}(A', r')$.
4. Send A' to the verifier.

Note that σ is not used at all by our P^* (and of course also not by V which is the distinguisher for the original protocol). Consider an experiment with a prover $\tilde{P}^*$ which is identical to P^* except it uses $\psi = \mathsf{Enc}_1(\mathsf{st})$, where st is the actual secret state corresponding to Q. Then by Eq. (10),

$$\Pr[(\tilde{P}^*, V)(x) = 1] \geq \alpha - \alpha_1 \ .$$

However, by definition of $\tilde{P}^*$, it is identical to P^* except for the use of ψ that encodes st instead of $0^{|\mathsf{st}|}$. If the two behave differently this translates to advantage for the distinguisher $\mathcal{A}$. In other words, the success probability of $\mathcal{A}$ is exactly

$$\alpha_2 = \Pr[(P^*, V)(x) = 1] - \Pr[(\tilde{P}^*, V)(x) = 1] \ .$$

We conclude that $\alpha_1 + \alpha_2 + \alpha_3 \geq \alpha$ as required.

Witness Indistinguishability. It remains to argue that $(P_{\mathrm{WI}}, V_{\mathrm{WI}})$ satisfies the WI criterion. Fix a function $n = n(\lambda) \leq \mathrm{poly}(\lambda)$, and fix any ensemble $\{(x_n, w_{1,n}, w_{2,n})\}_{\lambda \in \mathbb{N}}$, such that $(x_n, w_{1,n}) \in \mathcal{R}_L$ and $(x_n, w_{2,n}) \in \mathcal{R}_L$. Suppose for the sake of contradiction that there exists a (non-uniform) poly-size cheating verifier V^*_{WI}, such that

$$\mathsf{View}_{V^*_{\mathrm{WI}}}(P_{\mathrm{WI}}(1^\lambda, x_n, w_{1,n}), V^*_{\mathrm{WI}}(1^\lambda, x_n)) \not\approx \mathsf{View}_{V^*_{\mathrm{WI}}}(P(1^\lambda, x_n, w_{2,n}), V^*_{\mathrm{WI}}(1^\lambda, x_n)).$$

Assume w.l.o.g that V^*_{WI} is deterministic and denote $V^*_{\mathrm{WI}} = (V^*_{\mathrm{WI},1}, V^*_{\mathrm{WI},2})$ s.t. $(Q, \mathrm{sWI}_1, \psi) = V^*_{\mathrm{WI},1}(1^\lambda, x_n)$ generates the first message of V^*_{WI}, and $V^*_{\mathrm{WI},2}(c, \psi')$ is the distinguisher that takes the

message from P_{WI} and outputs a bit. Note that λ determines n and thus also x and (Q, sWI_1, ψ). Let $\mathsf{st} = \mathsf{Sim}_1(\psi)$, where $\mathsf{Sim}_1(\cdot)$ is the possibly inefficient first part of the simulator for the remote evaluation scheme (see Section 4.2). Note that st is uniquely well defined per λ.

We design a cheating non-uniform adversary V^*_{sWI} for the strongly witness indistinguishable scheme. Note that since the adversary is allowed to be non-uniform, we can hard-code the values $(x_n, w_{1,n}, w_{2,n}, Q, \text{sWI}_1, \psi, \mathsf{st})$ into V^*_{sWI}.

We start by defining the two distributions

$$\{\mathcal{X}'_{1,n(\lambda)}, \mathcal{W}'_{1,n(\lambda)}\}_{\lambda\in\mathbb{N}} \text{ and } \{\mathcal{X}'_{2,n(\lambda)}, \mathcal{W}'_{2,n(\lambda)}\}_{\lambda\in\mathbb{N}} ,$$

as required by the definition of sWI. The samplers for these distributions can also depend on $(x_n, w_{1,n}, w_{2,n}, Q, \text{sWI}_1, \psi, \mathsf{st})$. Formally, for $b \in \{1, 2\}$, the distribution $(\mathcal{X}'_{b,n(\lambda)}, \mathcal{W}'_{b,n(\lambda)})$ generates pairs $(x'_{b,n}, w'_{b,n}) \in \mathcal{R}_{L'}$ as follows:

1. Emulate the prover $P_{\text{WI}}(1^\lambda, x_n, w_{b,n}, Q, \text{sWI}_1, \psi)$, as follows.
   (a) Compute $A \leftarrow P(1^\lambda, x_n, w_{b,n}, Q)$.
   (b) Compute $c = \text{Com}(A, r)$ with uniformly chosen $r \leftarrow \{0, 1\}^{\text{poly}(\lambda)}$.
2. Set $x'_{b,n} = (1^\lambda, x_n, Q, c, \mathsf{st})$ and $w'_{b,n} = (A, r)$.

The computational hiding property of the commitment scheme implies that indeed

$$\{\mathcal{X}'_{1,n(\lambda)}\}_{\lambda\in\mathbb{N}} \overset{c}{\approx} \{\mathcal{X}'_{2,n(\lambda)}\}_{\lambda\in\mathbb{N}} .$$

We still need to prove that $(x'_{b,n}, w'_{b,n}) \in R_{L'}$ for $b \in \{1, 2\}$. If $\mathsf{st} \models Q$ then this follows from the perfect completeness of (P, V). If $\mathsf{st} \not\models Q$ this follows by definition (see Eq. (7)).

For this pair of distributions, the cheating verifier V^*_{sWI} runs as follows.

1. Send the fixed value sWI_1 as the first message.
2. Receive $x' = (1^\lambda, x_n, Q, c, \mathsf{st})$ and message $\text{sWI}_2 = P_{\text{sWI}}(x', w', \text{sWI}_1)$.
3. Generate simulated $\psi' = \mathsf{Sim}_2(\psi, \mathsf{st}, \text{sWI}_2)$, where Sim_2 is the simulator for the remote evaluation scheme (see Section 4.2), and output $V^*_{\text{WI},2}(c, \psi', \text{sWI}_2)$.

To prove that V^*_{sWI} indeed distinguishes between the distributions $\{\mathcal{X}'_{b,n(\lambda)}, \mathcal{W}'_{b,n(\lambda)}\}_{\lambda\in\mathbb{N}}$, we consider a hybrid where ψ' is generated as $\mathsf{Eval}(f_{1^\lambda,x,Q,c,w'_{b,n},\text{sWI}_1}, \psi)$. This hybrid is computationally indistinguishable from the original experiment by Claim 4.2. However, in this hybrid the distribution given to $V^*_{\text{WI},2}$ is identical to the one produced by P_{WI}, and since we assume that V^*_{WI} is a successful adversary against WI, it follows that our V^*_{sWI} successfully distinguishes between the distributions $\{\mathcal{X}'_{b,n(\lambda)}, \mathcal{W}'_{b,n(\lambda)}\}_{\lambda\in\mathbb{N}}$ in contradiction to the strong witness indistinguishability property. □

5 Succinct Single-Round Access Control Scheme

In this section we formalize the notion of succinct single-round access control presented in Section 1.2. The motivation is to allow authorities to provide users with certificates of owning certain

attributes (coming from a very large attribute universe). An authority is specified by a pair of master secret and public keys. After being issued a certificate, the user can succinctly prove in a witness indistinguishable manner that its attributes (issued by a specific authority) satisfy an arbitrary monotone formula. A formal definition follows.

Definition 5.1. *A succinct access control scheme consists of a tuple of* PPT *algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Query}, \mathsf{Proof}, \mathsf{Verdict})$*, with the following syntax:*

- Setup *takes as input the security parameter* 1^λ *and outputs a pair* $(\mathsf{mpk}, \mathsf{msk})$ *of master public and secret keys.*
- KeyGen *takes as input a tuple* $(1^\lambda, \mathsf{msk}, N, S, \mathsf{id})$*, where* λ *is the security parameter,* msk *is a master secret key (supposedly generated by* $\mathsf{Setup}(1^\lambda)$*),* $N \in \mathbb{N}$ *is a parameter such that* $N < 2^\lambda$*,* $S \subseteq [N]$*, and* $\mathsf{id} \in \{0,1\}^\lambda$*. It outputs a secret key* sk*.*
- Query *takes as input the security parameter* 1^λ *and outputs a pair* $(\mathsf{query}, \mathsf{state})$*.*
- Proof *takes as input a tuple* $(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{sk})$*, where* $\mathcal{F} : \{0,1\}^N \to \{0,1\}$ *is a monotone formula of size* $\mathrm{poly}(N)$ *and* $N < 2^\lambda$*,* query *is supposedly generated by running* $\mathsf{Query}(1^\lambda)$*, and* sk *is supposedly generated by running* KeyGen*. It outputs a* succinct *proof, denoted by* pf*, of length* $\leq \mathrm{poly}(\lambda)$*.*
- $\mathsf{Verdict}$ *takes as input a tuple* $(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{state}, \mathsf{mpk}, \mathsf{pf})$ *where* $\mathcal{F} : \{0,1\}^N \to \{0,1\}$ *is a monotone formula of size* $\mathrm{poly}(N)$ *for* $N < 2^\lambda$*,* $(\mathsf{query}, \mathsf{state})$ *is supposedly generated by* $\mathsf{Query}(1^\lambda)$*,* mpk *is supposedly generated by* $\mathsf{Setup}(1^\lambda)$*, and outputs* 1 *if and only if* pf *is accepting with respect to* $(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{state}, \mathsf{mpk})$*.*

  Moreover, the running time of $\mathsf{Verdict}$ *is not polynomial in* $\mathcal{F}$*, but rather is polynomial in the description length of a Turing machine that outputs* $\mathcal{F}$ *in* $\mathrm{poly}(|\mathcal{F}|)$ *time.*

In addition, an access control scheme must satisfy the following conditions:

- **Completeness.** *For any* $\lambda \in \mathbb{N}$ *any* $N < 2^\lambda$*, any poly-size monotone formula* $\mathcal{F} : \{0,1\}^N \to \{0,1\}$*, any identity* $\mathsf{id} \in \{0,1\}^\lambda$*, and any set* $S \subseteq [N]$ *such that* $\mathcal{F}(1_{1\in S}, \ldots, 1_{N\in S}) = 1$*,*

$$\Pr[\mathsf{Verdict}(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{state}, \mathsf{mpk}, \mathsf{pf}) = 1] = 1$$

  where the probability is over the random coin tosses of $\mathsf{Verdict}$*, over* $(\mathsf{query}, \mathsf{state}) \leftarrow \mathsf{Query}(1^\lambda)$*, over* $\mathsf{pf} \leftarrow \mathsf{Proof}(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{sk})$*, where* $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, N, S, \mathsf{id})$ *and* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$*.*

- **Soundness.** *For any* $\lambda \in \mathbb{N}$*, any polynomially-bounded* $N = N(\lambda)$*, any poly-size monotone formula* $\mathcal{F} : \{0,1\}^N \to \{0,1\}$*, the following holds: Fix any* PPT *adversary* $\mathcal{A}$ *that takes as input* $(1^\lambda, \mathsf{query}, \mathsf{mpk})$*, and has oracle access to an oracle* $\mathcal{O}$*, that on input* (id, S)*, outputs* $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, N, S, \mathsf{id})$ *if and only if* $\mathsf{id} \in \{0,1\}^\lambda$*,* $S \subseteq [N]$*, and* $\mathcal{F}(\mathbf{1}_{1\in S}, \ldots, \mathbf{1}_{N\in S}) = 0$*; and otherwise output* $\perp$*. Then*

$$\Pr[\mathsf{Verdict}(1^\lambda, \mathcal{F}, \mathsf{query}, \mathsf{state}, \mathsf{mpk}, \mathsf{pf}^*) = 1] = \mathrm{negl}(\lambda),$$

  where $\mathsf{pf}^* \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \mathsf{query}, \mathsf{mpk})$*, and where* $(\mathsf{query}, \mathsf{state}) \leftarrow \mathsf{Query}(1^\lambda)$ *and* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$*.*

- **Witness Indistinguishability (WI).** *For any $\lambda \in \mathbb{N}$, any polynomially-bounded $N = N(\lambda)$, any poly-size monotone formula $\mathcal{F} : \{0,1\}^N \to \{0,1\}$, any $\mathsf{id}_0, \mathsf{id}_1 \in \{0,1\}^\lambda$, and any sets $S_0, S_1 \subseteq [N]$ such that $\mathcal{F}(1_{1\in S_b}, \ldots, 1_{N\in S_b}) = 1$ for both $b = 0$ and $b = 1$, the following holds: For any* PPT *adversary $\mathcal{A}$ that generates $(\mathsf{query}^*, \mathsf{state}^*) = \mathcal{A}(1^\lambda, \mathsf{msk}, \mathsf{mpk})$,*

$$(\mathsf{query}^*, \mathsf{state}^*, \mathsf{msk}, \mathsf{mpk}, \mathsf{pf}_0) \approx (\mathsf{query}^*, \mathsf{state}^*, \mathsf{msk}, \mathsf{mpk}, \mathsf{pf}_1),$$

where

$$\mathsf{pf}_b(\lambda) \leftarrow \mathsf{Proof}(1^\lambda, \mathcal{F}, \mathsf{query}^*, \mathsf{sk}_b),$$

where $\mathsf{sk}_b \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, N, S_b, \mathsf{id}_b)$.

Remark 5.1. *We note that Definition 5.1 above guarantees that the identity of the prover remains hidden, even if the prover issues many proofs. Specifically, the WI guarantee implies that given msk (and given λ, N) one can efficiently simulate proofs for any prover, without knowing the attributes or the identity of the prover.*

We now formally state the result that is hinted in Section 1.2.

Theorem 5.1. *For any given super-polynomial function $T : \mathbb{N} \to \mathbb{N}$, assume the existence of a $\mathrm{poly}(T)$-secure computational PIR scheme (with $\mathrm{poly}(\lambda)$ communication complexity), the existence of a $\mathrm{poly}(T)$-secure 2-message oblivious transfer with statistical sender privacy (as in Definition 4.2 where Claim 4.4 is satisfied w.r.t. $\mathrm{poly}(T)$-size adversaries), the existence of a $\mathrm{poly}(T)$-secure signature scheme, and assume the existence of a statistically-binding commitment scheme that can be broken in time $\mathrm{poly}(T)$. Then there exists a succinct single-round access control scheme.*

Proof. The access control scheme uses the two components of this work:

- A monotone batch delegation scheme that is sound against $\mathrm{poly}(T)$-size cheating provers. This can be constructed from any $\mathrm{poly}(T)$-secure computational PIR scheme (see Theorems 2.1 and 3.1). We denote this delegation scheme by (P, V).
- the WI compiler w.r.t. the super polynomial function $T = T(n)$ (as in Theorem 4.5). This can be constructed assuming the existence of a $\mathrm{poly}(T)$-secure 2-message oblivious transfer with statistical sender privacy (as in Definition 4.2 where Claim 4.4 is satisfied w.r.t. $\mathrm{poly}(T)$-size adversaries), and assuming the existence of a statistically binding commitment scheme Com that can be broken in time $\mathrm{poly}(T)$.

In addition, the access control uses any $\mathrm{poly}(T)$-secure signature scheme SIG (i.e., one that is existentially unforgeable against chosen message attacks by a $\mathrm{poly}(T)$-size adversary), which can be based on any $\mathrm{poly}(T)$-hard to invert one-way function, and does not require additional assumptions.

In what follows, we first present an access control scheme without the WI guarantee. We denote the algorithms in this (non WI) scheme by

$$(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Query}', \mathsf{Proof}', \mathsf{Verdict}')$$

We then use our WI compiler from Section 4 to compile $(\mathsf{Query}', \mathsf{Proof}', \mathsf{Verdict}')$ into a witness indistinguishable protocol $(\mathsf{Query}, \mathsf{Proof}, \mathsf{Verdict})$, thus obtaining our final access control scheme

$$(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Query}, \mathsf{Proof}, \mathsf{Verdict}).$$

- $\mathsf{Setup}(1^\lambda)$ generates a pair of keys $(\mathsf{mpk}, \mathsf{msk})$ by running the key generation algorithm of the signature scheme SIG (with security parameter λ).
- $\mathsf{KeyGen}(1^\lambda, \mathsf{msk}, N, S, \mathsf{id})$ samples a random tag $\mathsf{tag} \in \{0,1\}^\lambda$, it computes a signature $\sigma_{\mathsf{tag},i} = \mathsf{Sign}_{\mathsf{msk}}(\mathsf{tag}\|i)$ for every attribute $i \in S$. It outputs $(\mathsf{tag}, \{\sigma_{\mathsf{tag},i}\}_{i\in S})$.
- $\mathsf{Query}'(1^\lambda)$ generates a pair $(Q, \mathsf{st}) \leftarrow V(1^\lambda)$,[7] where Q is the query string and st is the internal state.
- $\mathsf{Proof}'(1^\lambda, \mathcal{F}, Q, (\mathsf{tag}, \{\sigma_{\mathsf{tag},i}\}_{i\in S}))$ runs the prover P (from the monotone **NP** delegation scheme), respective to $(R, \mathcal{F}, \{x_i\}_{i\in[N]})$, where $x_i = \mathsf{mpk}\|\mathsf{tag}\|i$, R is the **NP** relation defined by

  $$(\mathsf{mpk}\|\mathsf{tag}\|i, \sigma) \in R \Leftrightarrow \mathsf{Verify}_{\mathsf{mpk}}(\mathsf{tag}\|i, \sigma) = 1,$$

  and $\mathcal{F}$ is the formula corresponding to the access structure. Denote the answer generated by P by A. Then Proof' outputs

  $$\mathsf{pf} = (A, \mathsf{tag}).$$

- $\mathsf{Verdict}'(1^\lambda, \mathcal{F}, Q, \mathsf{pf}, \mathsf{st}, \mathsf{mpk})$ checks that A verifies correctly respective to the **NP** statement $\{x_i\}_{i\in[N]}$, where $x_i = \mathsf{mpk}\|\mathsf{tag}\|i$.

As mentioned above, the algorithms $(\mathsf{Query}, \mathsf{Proof}, \mathsf{Verdict})$ are constructed by applying the WI transformation from Section 4. Namely, the algorithm $\mathsf{Query}(1^\lambda)$ does the following:

1. Compute $(Q, \mathsf{st}) \leftarrow \mathsf{Query}'(1^\lambda)$.
2. Compute $(\psi, \sigma) \leftarrow \mathsf{Enc}(\mathsf{st})$, where Enc is the encoding of the private remote evaluation scheme, as defined in Section 4.[8] has nothing is the secret information
3. Compute $(\mathrm{sWI}_1, \mathsf{st}_{\mathrm{sWI}}) \leftarrow V_{\mathrm{sWI}}(1^\lambda)$, where V_{sWI} is the verifier of the delayed input strong WI scheme, as defined in Section 4.
4. Output $(Q, \psi, \mathrm{sWI}_1)$, and keep $(\sigma, \mathsf{st}, \mathsf{st}_{\mathrm{sWI}})$ as the secret state for verification.

The algorithm $\mathsf{Proof}(1^\lambda, \mathcal{F}, (Q, \psi, \mathrm{sWI}_1), (\mathsf{tag}, \{\sigma_{\mathsf{tag},i}\}_{i\in S}))$ does the following:

1. Compute $(A, \mathsf{tag}) \leftarrow \mathsf{Proof}'(1^\lambda, \mathcal{F}, Q, (\mathsf{tag}, \{\sigma_{\mathsf{tag},i}\}_{i\in S}))$.
2. Sample randomness $r \leftarrow \{0,1\}^\lambda$, and compute $c = \mathrm{Com}((A, \mathsf{tag}), r)$.
3. Similarly to the construction of the WI scheme in Section 4, define the **NP** language L' as follows:

   $$L' = \{(1^\lambda, Q, c, \mathsf{st}, \mathsf{mpk}) : \exists (A, \mathsf{tag}, r, x) \text{ s.t.}$$
   $$\left(\mathsf{st} \not\models Q\right) \vee \left(c = \mathrm{Com}((A, \mathsf{tag}), r) \wedge V(1^\lambda, x, Q, A, \mathsf{st}) = 1 \wedge (x = (x_i)_{i\in[N]} = (\mathsf{mpk}\|\mathsf{tag}\|i)_{i\in[N]})\right)\} .$$

   Define (implicitly since st is not known) $x' = (1^\lambda, Q, c, \mathsf{st}, \mathsf{mpk})$, and $w' = (A, \mathsf{tag}, r, x)$ as its corresponding witness with respect to $R_{L'}$, i.e. $(x', w') \in \mathcal{R}_{L'}$.

[7] Note that it is important that in the monotone delegation scheme the query string generation is independent of the instance to be proven.

[8] We note that σ is not related to the signature scheme, and denoting by $\sigma_{\mathsf{tag},i}$ a signature of $\mathsf{tag}\|i$ is an abuse of notion.

4. Given ψ, compute $\psi' = \mathsf{Eval}(f, \psi)$ where $f = f_{1^\lambda, Q, c, \mathsf{mpk}, w', \mathrm{sWI}_1}$, is the function that on input st outputs $\mathrm{sWI}_2 \leftarrow P_{\mathrm{sWI}}(1^\lambda, x', w', \mathrm{sWI}_1)$.

5. Output (c, ψ').

The algorithm $\mathsf{Verdict}(1^\lambda, \mathcal{F}, (Q, \psi, \mathrm{sWI}_1), (c, \psi'), (\sigma, \mathsf{st}, \mathsf{st}_{\mathrm{sWI}}), \mathsf{mpk})$ does the following:

1. Decrypt the ciphertext ψ', by computing $\mathrm{sWI}_2 \leftarrow \mathsf{Dec}(\sigma, \psi')$.

2. Accept if and only if
$$V_{\mathrm{sWI}}(1^\lambda, x', \mathrm{sWI}_1, \mathrm{sWI}_2, \mathsf{st}_{\mathrm{sWI}}) = 1,$$
where $x' = (1^\lambda, Q, c, \mathsf{st}, \mathsf{mpk})$.

We next argue that the final access control scheme,

$$(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Query}, \mathsf{Proof}, \mathsf{Verdict}),$$

satisfies the desired completeness, soundness and WI guarantees.

Completeness. The completeness follows immediately from the completeness of (P, V), the completeness of the WI transformation (see Theorem 4.5), and the correctness of the signature scheme.

Soundness. Fix any PPT adversary $\mathcal{A}$, and suppose for the sake of contradiction that for infinitely many $\lambda \in \mathbb{N}$,

$$\begin{aligned}&\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, (Q, \psi, \mathrm{sWI}_1), \mathsf{mpk}) = (c, \psi') : \\ &(\mathsf{Verdict}(1^\lambda, \mathcal{F}, (Q, \psi, \mathrm{sWI}_1), (c, \psi'), (\sigma, \mathsf{st}, \mathsf{st}_{\mathrm{sWI}}), \mathsf{mpk}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)},\end{aligned}$$

where the probability is over $(Q, \mathsf{st}) \leftarrow \mathsf{Query}(1^\lambda)$, over $(\psi, \sigma) \leftarrow \mathsf{Enc}(1^\lambda, \mathsf{st})$, over mpk generated using the key generation procedure of SIG, over $(\mathrm{sWI}_1, \mathsf{st}_{\mathrm{sWI}}) \leftarrow V_{\mathrm{sWI}}(1^\lambda)$, and over the randomness of the oracle $\mathcal{O}$. Recall that $\mathcal{O}$, on input (id, S), outputs $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, N, S, \mathsf{id})$ if and only if $\mathsf{id} \in \{0,1\}^\lambda$, $S \subseteq [N]$, and $\mathcal{F}(\mathbf{1}_{1 \in S}, \ldots, \mathbf{1}_{N \in S}) = 0$; and otherwise output $\perp$.

Denote by
$$\mathrm{sWI}_2 \triangleq \mathsf{Dec}(\psi', \sigma).$$

By the definition of $\mathsf{Verdict}$,

$$\begin{aligned}&\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, (Q, \psi, \mathrm{sWI}_1), \mathsf{mpk}) = (c, \psi') : \\ &V_{\mathrm{sWI}}((1^\lambda, Q, c, \mathsf{st}, \mathsf{mpk}), \mathrm{sWI}_1, \mathrm{sWI}_2, \mathsf{st}_{\mathrm{sWI}}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)}.\end{aligned}$$

Let (A, tag) be the value committed to by c. Recall that c is a statistically binding commitment, and according to our assumption, the value committed to can be found in time $\mathrm{poly}(T)$. The soundness of the strong WI scheme implies that

$$\begin{aligned}&\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, (Q, \psi, \mathrm{sWI}_1), \mathsf{mpk}) = (c, \psi') : \\ &V(1^\lambda, x, Q, A, \mathsf{st}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)},\end{aligned}$$

where $x = (x_1, \ldots, x_N)$, and $x_i = \mathsf{mpk} \| \mathsf{tag} \| i$ for each $i \in [N]$, and where $c = \mathrm{Com}((A, \mathsf{tag}), r)$ for some $r \in \{0,1\}^\lambda$.

This implies that there exists a $\mathrm{poly}(T)$-size adversary $\mathcal{A}$ such that

$$\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, Q, \psi, \mathsf{mpk}) = (A, \mathsf{tag})) : V(1^\lambda, x, Q, A, \mathsf{st}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)},$$

where $x = (x_1, \ldots, x_N)$, and $x_i = \mathsf{mpk} \| \mathsf{tag} \| i$ for each $i \in [N]$. This, together with the fact that the underlying encoding scheme (i.e., the underlying OT-scheme) is $\mathrm{poly}(T)$-secure, implies that

$$\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, Q, \mathsf{mpk}) = (A, \mathsf{tag})) : V(1^\lambda, x, Q, A, \mathsf{st}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)},$$

where $x = (x_1, \ldots, x_N)$, and $x_i = (\mathsf{mpk}, \mathsf{tag}, i)$ for each $i \in [N]$.

Suppose $\mathcal{A}$ makes at most $q = q(\lambda)$ oracle queries. Let $\mathsf{tag}_1, \ldots, \mathsf{tag}_q \leftarrow \{0,1\}^\lambda$ be random and independently chosen tags, and suppose the oracle answers the i'th query with tag_i as the tag. In what follows we denote the i'th oracle answer by sk_i. Then, for infinitely many $\lambda \in \mathbb{N}$,

$$\Pr[\mathcal{A}(1^\lambda, Q, \mathsf{mpk}, \{\mathsf{sk}_i\}_{i \in [q]}) = (A, \mathsf{tag}) : V(1^\lambda, x, Q, A, \mathsf{st}) = 1] \geq \frac{1}{\mathrm{poly}(\lambda)},$$

where $x = (x_1, \ldots, x_N)$ and $x_i = (\mathsf{mpk}, \mathsf{tag}, i)$ for every $i \in [N]$.

By the proof-of-knowledge of the underlying monotone batch delegation scheme (see Theorem 3.1), it follows that there exists a PPT extractor algorithm $\mathcal{E}$ such that for infinitely many λ's

$$\Pr[\mathcal{E}(1^\lambda, Q, \mathsf{mpk}, \{\mathsf{sk}_i\}_{i \in [q]}) = (A, \mathsf{tag}, S, (\sigma_{\mathsf{tag},i})_{i \in S}) : (\forall i \in S,\ \mathsf{Verify}_{\mathsf{mpk}}(\mathsf{tag} || i, \sigma_{\mathsf{tag},i}) = 1) \wedge (\mathcal{F}(1_{1 \in S}, \ldots, 1_{N \in S}) = 1)] \geq \frac{1}{\mathrm{poly}(\lambda)}.$$

Recall that by the definition of the oracle, $\{\mathsf{sk}_i\}_{i \in [\ell]}$ does not contain signatures of $(\mathsf{tag} || i)_{i \in S}$ for any set S for which $\mathcal{F}(1_{1 \in S}, \ldots, 1_{N \in S}) = 1$. This implies that

$$\Pr[\mathcal{E}(1^\lambda, Q, \mathsf{mpk}, \{\mathsf{sk}_i\}_{i \in [q]}) = (S, \{\sigma_{\mathsf{tag},i}\}_{i \in S}) : (\forall i \in S,\ \mathsf{Verify}_{\mathsf{mpk}}(\mathsf{tag} || i, \sigma_{\mathsf{tag},i}) = 1) \wedge \mathsf{tag} \notin \{\mathsf{tag}_1, \ldots, \mathsf{tag}_q\}] \geq \frac{1}{\mathrm{poly}(\lambda)},$$

contradicting the fact that the underlying signature scheme is secure against $\mathrm{poly}(T)$-size adversaries.

Witness Indistinguishability. The WI condition follows immediately from the fact that the commitment scheme is (computationally) hiding, and from the strong WI property of $(P_{\mathrm{sWI}}, V_{\mathrm{sWI}})$. □

Acknowledgements

We thank Shafi Goldwasser and Justin Holmgren for suggesting the question of generalizing the work on delegating batch NP computations to other predicates.

References

[BCC+14] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. The hunting of the SNARK. *IACR Cryptology ePrint Archive*, 2014:580, 2014.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *STOC*, pages 111–120. ACM, 2013.

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 474–482. ACM, 2017.

[BKK+17] Saikrishna Badrinarayanan, Yael Tauman Kalai, Dakshita Khurana, Amit Sahai, and Daniel Wichs. Non-interactive delegation for low-space non-deterministic computation. Cryptology ePrint Archive, Report 2017/1250, 2017. To appear in STOC 2018.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *FOCS*, pages 97–106. IEEE, 2011. Invited to SIAM Journal on Computing.

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982.*, pages 199–203. Plenum Press, New York, 1982.

[Cha85] David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, 1985.

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414. Springer, 1999.

[DFH12] Ivan Damgård, Sebastian Faust, and Carmit Hazay. Secure two-party computation with low communication. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 54–74, 2012.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-hop homomorphic encryption and rerandomizable yao circuits. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172. Springer, 2010.

[HK07] Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *IACR Cryptology ePrint Archive*, 2007:118, 2007.

[Hol18] Justin Holmgren. Private communication, 2018.

[JKKR17] Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Ron Rothblum. Distinguisher-dependent simulation in two rounds and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 158–189. Springer, 2017.

[Kal05] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 78–95. Springer, 2005.

[KKS18] Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Statistical witness indistinguishability (and more) in two messages. *IACR Cryptology ePrint Archive*, 2018:168, 2018.

[KRR13] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. Delegation for bounded space. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 565–574. ACM, 2013.

[KRR14] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *STOC*, pages 485–494. ACM, 2014.

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 436–453. IEEE Computer Society, 1994. Full version in SIAM J. Comput., 30(4):1253 1298, 2000.

[Nao89] Moni Naor. Bit commitment using pseudo-randomness. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 128–136. Springer, 1989.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA.*, pages 448–457, 2001.

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 536–553. Springer, 2014.

[PR14] Omer Paneth and Guy N. Rothblum. Publicly verifiable non-interactive arguments for delegating computation. *IACR Cryptology ePrint Archive*, 2014:981, 2014.

[RRR16] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 49–62, 2016.

[RRR18] Omer Reingold, Guy Rothblum, and Ron Rothblum. Efficient batch verification for UP. ECCC Report TR18-022, 2018.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164, 1982.