Secure Database Commitments and Universal Arguments of Quasi Knowledge

Melissa Chase
Microsoft Research
Redmond, USA
melissac@microsoft.com

Ivan Visconti*
University of Salerno
ITALY
visconti@dia.unisa.it

Abstract

In this work we focus on a simple database commitment functionality where besides the standard security properties, one would like to hide the size of the input of the sender. Hiding the size of the input of a player is a critical requirement in some applications, and relatively few works have considered it. Notable exceptions are the work on zero-knowledge sets introduced in [MRK03], and recent work on size-hiding private set intersection [ADCT11]. However, neither of these achieves a secure computation (i.e., a reduction of a real-world attack of a malicious adversary into an ideal-world attack) of the proposed functionality.

The first result of this submission consists in defining "secure" database commitment and in observing that previous constructions do not satisfy this definition. This leaves open the question of whether there is any way this functionality can be achieved.

We then provide an affirmative answer to this question by using new techniques that combined together achieve "secure" database commitment. Our construction is in particular optimized to require only a constant number of rounds, to provide non-interactive proofs on the content of the database, and to rely only on the existence of a family of CRHFs. This is the first result where input-size hiding secure computation is achieved for an interesting functionality and moreover we obtain this result with standard security (i.e., simulation in expected polynomial time against fully malicious adversaries, without random oracles, non-black-box extraction assumptions, hardness assumptions against super-polynomial time adversaries, or other controversial/strong assumptions).

A key building block in our construction is a universal argument enjoying an improved proof of knowledge property, that we call quasi-knowledge. This property is significantly closer to the standard proof of knowledge property than the weak proof of knowledge property satisfied by previous constructions.
Keywords: ZK sets, universal arguments, input-size hiding computation.

1 Introduction

Secure computation. The standard notion of "security" for any multi-party computation [GMW87] involves describing an ideal model where parties have access to a trusted

*Work partially done while visiting University of California at Los Angeles, USA.

functionality which will carry out the desired computation without revealing any information about the parties' secret inputs and outputs. Then, very informally, given a communication network in the real world, without any trusted functionality, a protocol π securely realizes a multi-party computation if any successful attack by an adversary of π can be translated into a successful attack in the ideal world. Since the latter is trivially secure then π is secure in the real world as well.

Therefore, the real-world/ideal-world paradigm [GMW87] allows one to prove the security of protocols in a simulation-based fashion, with strong security guarantees. This turns out to be very useful when protocols are used as subprotocols and provides robust security when the ideal functionality is correctly designed. All the fundamental primitives have been defined in terms of ideal functionalities and thus they can be securely realized in the real-world/ideal-world paradigm using the general results of [GMW87].

However the current state-of-the art still does not properly address the issue of considering *the size of the input* of a player as information to be kept private. Here we consider the problem of hiding the input size in one of the most basic primitives: a commitment scheme.

Secure Database Commitments. Here we address the case where a player wants to commit to a large set of data, and then to partially open that commitment, both without revealing the size of the committed set. More specifically, we consider the setting where a party (the sender) may want to commit to a large elementary database composed by key-value pairs, without revealing its size. Then in the opening phase, the sender might want to reveal part of his database one item at a time depending on queries of the receiver. In particular, the receiver will ask for some keys, and the sender wants to convince the receiver of the value associated to each requested key. These partial opening queries should reveal no information about the rest of the database, including the size.

The main question: feasibility of Secure Database Commitments. Following the above discussion, and the fact that we have general results for achieving secure multi-party computation, one could ask the following natural question: "Why should we focus on the design of protocols for secure database commitments if we can use the known constructions for secure two-party computation?". There is a crucial difference between this problem, and the one considered in [GMW87] and in all subsequent work (to the best of our knowledge). Our notion of secure database commitment critically requires that the size of the prover's input must be hidden. In contrast, the [GMW87] definition, according to [Gol04] and to the known constructions, allows a party either at the beginning of or during the computation, to learn the size of the other party's input. This information is actually revealed in all known protocols, mainly because many of the tools used to allow extraction of the adversary's implicit input (e.g., zero-knowledge proofs of knowledge) have communication that depends (polynomial) on the size of the input.

Consequently, traditional results for secure two-party computation cannot be used to obtain secure database commitments, and we need to develop new tools and a significantly new approach. This presents the following interesting open problem: *Is the notion of secure database commitment and more generally a meaningful notion of input-size hiding secure computation achievable?*

Difficulties in achieving input-size hiding secure computation with standard assumptions. We stress that the main challenge in size-hiding secure computation is as follows: when proving security against a real world adversarial prover, we need to design a simulator which can extract the input from the adversary so that it can provide it to the ideal functionality. At the same time, we can not assume any fixed polynomial upper bound for the size of the input (the protocol should work for *any* polynomial sized input), and we can not allow the amount of communication to vary with the size of the input.

The real difficulties lie in obtaining a standard security notion (showing an efficient ideal adversary from an efficient real-world adversary), therefore avoiding controversial (sometimes not even falsifiable) conjectures such as random oracles, non-black-box extraction assumptions (e.g., the Diffie-Hellman knowledge of exponent assumption and its variations [HT98]), complexity leveraging (i.e., hardness assumptions against super-polynomial time adversaries) and other non-standard assumptions.

Relationship to Zero-Knowledge Sets. We note that the requirements described for secure database commitments are essentially those given in [MRK03] where Micali, Rabin and Kilian introduced the concept of a zero-knowledge set (ZKS). This primitive allows a party to first commit to a database, and later to answer queries on that database and to prove that the responses are consistent with the original commitment, as in the secure database commitments described above.

However, the definition given by [MRK03] and by all subsequent papers is a *property-based definition* that requires: 1) soundness: the commitment should be binding, i.e., for each query there should be only one response for which the prover can give a convincing proof; and 2) zero-knowledge: both the commitment and the proofs should not reveal any information about the rest of the database (beyond the information that is asked in the query), not even the number of elements it contains.

Furthermore, we argue that the constructions they provide (and later constructions for ZKS, see [CHL+05, CDV06, GM06, CFM08, PX09, LY10]) do not satisfy a typical real-world/ideal-world definition in the spirit of "secure computation". To see this, note that all previous schemes for ZKS included a non-interactive commitment of the set. As mentioned above, we do not consider non-black-box extraction assumptions; moreover, standard non-black-box techniques introduced in [Bar01] so far have been successfully used only in conjunction with interaction.

However, we argue that black-box extraction is impossible for a scheme which is size hiding and has a non-interactive commitment phase. This follows because such a scheme must allow for sets of *any* polynomial size, which means there will be at least $2^{\mathsf{superpoly}(k)}$ possible sets; at the same time, the length of the non-interactive commitment must be bounded by a fixed polynomial in k. Thus, a simple counting argument shows that there is no way (based on standard assumptions and security notions) that a simulator can correctly identify the committed set given only the single commitment message. Note that this argument also holds in the CRS model, as long as the CRS is of length polynomial in k, as even in this case the simulator still gets only $\mathsf{poly}(k)$ bits of information to identify one database out of $2^{\mathsf{superpoly}(k)}$. Therefore, no previous construction of zero-knowledge sets can be proved to be a secure realization of the database commitment functionality (with a black-box simulator).

Looking ahead, in our construction we will avoid these limitations by allowing an interactive commitment phase. The possibility of using interaction was already mentioned in some previous work, but only for the query/proof phase, thus without making any contribution towards addressing this extraction problem. Instead, our goal is to use interaction in the commitment phase, still keeping the query/answer phase non-interactive.

Other related work. Recently similar issues have been considered by Ateniese et al. in [ADCT11] for the problem of Private Set Intersection. However their solution uses random oracles, and obtains security with respect to semi-honest adversaries only. The goal of our work is to obtain security against malicious players as required in secure computation, and we will obtain this result in the standard model using only standard complexity-theoretic assumptions.

Timing attacks. We note that there may be some cases in which any attempt to hide the size of inputs is vulnerable to timing attacks as discussed in [IP07] (in which an adversary can guess the size of the input depending on the amount of time required to perform computations). However, there are many settings where these attacks will have only limited impact. Indeed, notice that since the adversary does not necessarily know the amount of resources of the other player (the committer could perform his computation by distributing the workload among computing devices in number proportional to the size of the input) the timing may in fact give very little information.

1.1 Our Results

In this work we first put forth the notion of secure database commitment. Following the traditional notion of "secure computation" we define the natural ideal functionality for database commitment $\mathcal{F}_{DbCom}$ and observe that it implies all required security guarantees needed by zero-knowledge sets.

We then present a constant-round protocol that securely realizes the $\mathcal{F}_{DbCom}$ functionality. The protocol is interactive, and is secure in the standard model (i.e., we do not require any set-up assumptions) based on the existence of families of collision-resistant hash functions (CRHFs, for short) (notice that CRHFs are implied by the existence of ZK sets [CDV06]).

We stress that our protocol has optimal amortized round complexity, as queries and answers are non-interactive. In addition, the use of the real-world/ideal-world paradigm, and the simulation in polynomial time should make it much easier to use this primitive as part of a larger protocol.

Our construction is based on a special universal argument that enjoys a new proof of knowledge property which is closer to the traditional proof of knowledge property. We define this new property, give a construction which satisfies it based on CRHFs, and show how it can be used to implement secure database commitments. (However, we stress that there were many other subtle issues to address, and our stronger notion of universal argument alone is not sufficient for directly obtaining input-size hiding secure computation.)

Techniques. As described above, the biggest challenge is in defining a simulator that can extract a witness whose length can be any polynomial, from only a fixed polynomial

amount of communication. Note that the standard approach does not work in this setting: it might seem that a simple solution would be to ask the sender to give a commitment to the database, and an interactive zero-knowledge proof of knowledge of the database contained in the provided commitment. However, in general such proofs may reveal the size of the witness, which in this case means the size of the database.

Instead, we use a different tool, namely a witness indistinguishable universal argument of quasi knowledge (UAQK). A universal argument is a (computationally-sound) proof system which guarantees that the communication is only proportional to the size of the statement. At the same time, it guarantees that the honest prover can run in time polynomial in the size of the witness, which is the best that we can hope for.

Universal arguments were introduced by Barak [Bar01] as part of the design of a non-black-box simulator. Barak showed a construction based on CRHFs which satisfied a weak proof of knowledge property. However, there are some inherent challenges in defining a standard proof of knowledge for universal arguments, and the extraction guarantees of his definition do not seem to be sufficient for our application. To deal with this we define a new proof of knowledge property which we call "quasi knowledge" which provides a functionality somewhat closer to that of a standard proof of knowledge. We show that it can be used in our application to implement the traditional commitment and proof of knowledge strategy described above. (Of course we are glossing over many issues here, and the application is not straightforward at all - see Section 4 for details.)

Finally, we note that this is of additional interest, because we use universal arguments for a completely different purpose than the one that motivated Barak's original construction [Bar01], which was the design of a non-black-box simulator.

Universal arguments: from weak proof of knowledge to proof of quasi knowledge. Very informally, the standard proof of knowledge property guarantees that if a prover can convince a verifier of the truthfulness of a theorem with non-negligible probability, then one can efficiently obtain an NP witness for that theorem in expected polynomial time with overwhelming probability.

When one focuses instead on universal languages (i.e., when one would like to prove that a Turing machine accepts the provided input within a given — not necessarily polynomial — number of steps), then one can not always efficiently obtain a corresponding witness since its length can be superpolynomial. In this case a restricted proof of knowledge property might instead focus on extracting an implicit representation of the witness, in the form of a polynomial-sized circuit that when evaluated on input i provides the i-th bit of the witness.

Unfortunately there is no known construction of a universal argument with such a property. The current state of the art, [BG02], shows how to get a *weak* proof of knowledge property that includes one more restriction: when a prover proves a theorem with probability $1/q$, one can efficiently get an implicit representation of a circuit with probability $1/q'$ that is polynomially related to $1/q$. This essentially means that one can efficiently get a *candidate* implicit representation of a witness, with non-negligible probability when $1/q$ is non-negligible. However, there is no guarantee that the candidate implicit representation is actually correct (one can not simply test the circuit asking for all bits of the witness since the size of a witness can be superpolynomial).

Furthermore, there is an additional subtlety in the way this weak proof of knowledge

property is defined. For any polynomial q, there is guaranteed to be a polynomial time extractor that can extract candidate representations of the witness from any prover that proves a theorem with probability $1/q$, and this extractor is guaranteed to succeed with the related probability $1/q'$; however, the choice of this extractor and its running time may depend on q. This has two disadvantages: (1) in order to use an extractor, we must first determine which q we are interested in, and (2) an efficient extractor is required to exist only for polynomials q, while nothing is required when q is super polynomial. (In fact, in the construction given in [BG02], the running time of the extractor is roughly q^d where d is some constant greater than 1, therefore when q is superpolynomial the running time of the extractor increases quickly and its expected running time is not polynomial). As a result of these disadvantages, converting a weak proof of knowledge system into a proof system with the standard proof of knowledge property is very non-trivial, even when one is happy with the extraction of an implicit representation of the witness. (This is because the standard proof of knowledge property requires that, regardless of the success probability of the adversarial prover, the extractor must run in expected polynomial time.)

In this paper we remove the above additional restriction of the *weak* proof of knowledge property and replace it with a more standard proof of knowledge property, requiring that an extractor outputs in expected polynomial time a correct implicit representation of the witness. The only caveat is that, while we require that the extracted implicit representation must correspond to some true witness, we do allow it to contain a few errors, as long as those errors are unlikely to be noticed by any polynomial time process. (Note that if the witness is polynomial sized, then this still implies the standard proof of knowledge property.) We will say that an argument system is an argument of "quasi" knowledge if it enjoys this property.

Finally, we construct a constant-round witness-indistinguishable universal argument of quasi-knowledge under standard complexity-theoretic assumptions.

2 Universal Arguments of Quasi Knowledge

A universal argument [BG02] is an interactive argument system for proving membership in **NEXP** that satisfies a weak proof of knowledge property. For the formal definition, see Appendix A.3. We will use the following result.

Theorem 1 ([BG02, Bar04b], informal) *Suppose there exists a hash function ensemble that is collision-resistant against polynomial-sized circuits. Then there exists a universal argument system that is constant-round, public-coin and witness indistinguishable.*

Moreover, there exists a construction for which there exists a weak proof of knowledge extractor satisfying two additional properties. First, the construction of this extractor is parameterized by a lower bound α on the success probability of the adversarial prover $P_n^\star$. There exists a fixed constant d (independent of P^*) such that the resulting extractor, that we denote by $E^{P_n^\star}(\alpha,\cdot,\cdot)$ has expected running time at most $(\frac{n}{\alpha})^d$ for any n-bit instance y.

Furthermore, it has the property that there is a fixed known polynomial $q^\star$ such that for any polynomial-sized prover $P_n^\star$ that succeeds in causing verifier $V(y)$ to accept with probability $q(n)$ for an n-bit instance y, and for any $\alpha < q(n)$, the probability that the extractor $E^{P_n^\star}(\alpha,y,\cdot)$ succeeds in extracting bits of a valid witness is at least $\frac{\alpha}{q^\star(n)}$. For details of the construction and proof, see the proof of Lemma A.2.5 in [Bar04b].

2.1 A New Notion: Quasi-Knowledge

As described in Section 1.1, we aim to construct a universal argument with a more standard proof of knowledge property. The resulting proof of knowledge will resemble the standard proof of knowledge property with two exceptions.

The first is that the extractor will produce only an implicit representation of a witness. This is necessary since UAs are used to prove statements that are not necessarily in NP, therefore the length of the witness may not be polynomial, but still we want our extractor to run in (expected) polynomial time. We formalize this saying that the extractor on input i will produce a candidate for the i-th bit of the witness w.

The second difference is that even when the extractor is successful, we do not require that the extracted witness be perfectly correct. We may allow some small fraction of the extracted bits to be incorrect, as long as this will have a negligible effect when used in any polynomial-time process. We formalize this by saying that for any (potentially adversarially and adaptively generated) polynomial sized set of indices I, it must be the case that with all but negligible probability, all of the associated bits produced by the extractor will be correct with respect to some valid witness.

Thus, our definition requires an extractor which satisfies two properties. The first is that, for any PPT (and potentially adversarial) sampling algorithm S which chooses a set of indices I, with all but negligible probability over the choice of random tape r, the extractor E_r with oracle access to $P^\star$ will output a set of bits $\{w_i\}_{i\in I}$ that are consistent with some valid witness w. Note that we allow S to choose these indices adaptively, after querying E on indices of it's choice. This allows for the fact that in a larger application, which positions of the witness need to be extracted may depend on the values of the bits which have been seen so far.

The second property requires that the expected running time of E be at most a polynomial factor (in $|y|$) larger than $\frac{1}{p(|y|)}$ where $p(|y|)$ is the success probability of the adversarial prover P^*. There is a slight issue here in that, even if the *expected* running time of $E(y,i)$ was guaranteed to be polynomial for each i, it might still be possible that for any choice of random tape r, an adversarial and adaptive sampling algorithm S could find at least one index that causes $E_r(y,i)$ to run in super-polynomial time. Thus we also require that the running time be independent of i; this implies that the expected running time of E on any set of inputs (even those that are chosen adversarially and adaptively) will also be at most a polynomial factor larger than $\frac{1}{p(|y|)}$. For our application this will be particularly useful, as it implies that E can be converted into a circuit which implicitly describes the witness and whose expected size is at most $\frac{\mathsf{poly}(|y|)}{p(|y|)}$. (We will see more details in Section 4.)

Definition of universal argument of quasi knowledge (UAQK). We construct a universal argument $\langle P(\cdot,\cdot), V(\cdot)\rangle$ such that the efficient verification, completeness and computational soundness properties hold as usual, but the weak proof of knowledge property is replaced as follows.

Definition 1 *A universal argument system $\langle P(\cdot,\cdot), V(\cdot)\rangle$ for the universal language L_U is a universal argument of* quasi knowledge *(UAQK) if there exists an algorithm E and negligible functions ν_1, ν_2 such that for any $y \in L_U \cap \{0,1\}^n$, and for any polynomial-size circuit family $\{P_n^\star\}_{n\in N}$ there exists a polynomial* poly *such that the following two properties hold.*

1. *For any family* $\{\mathsf{S}_n\}_{n\in N}$ *of polynomial-sized circuits, for sufficiently large* n*, for any* $y \in L_U \cap \{0,1\}^n$*, if* $\text{Prob}[\ \mathtt{out}_V(\langle P_n^\star, V(y)\rangle) = 1\ ] > \nu_1(n)$*, then*

$$\text{Prob}_r[\ I \leftarrow \mathsf{S}_n^{E_r^{P_n^\star}(y,\cdot)}(y); \{w_i \leftarrow E_r^{P_n^\star}(y,i)\}_{i\in I} : \exists \hat{w} = \hat{w}_1, \ldots \hat{w}_s, (\hat{w}, y) \in R_U \wedge (w_i = \hat{w}_i \ \ \forall i \in I)\ ] \geq 1 - \nu_2(n).$$

where the probability is over the choice of the coins r *used by* E *(indeed by* E_r *we mean a run of* E *with coins* r*). Note that the same coins are used for all* $i \in I$*.*[1]

2. *For any* $n \geq 2$*, for any* $y \in L_U \cap \{0,1\}^n$*: The running time of* $E^{P_n^\star}(y,i)$ *is independent of the choice of* i*, and if we let* $p = \text{Prob}[\ \mathtt{out}_V(\langle P_n^\star, V(y)\rangle) = 1\ ] > 0$*, then the expected running time of* $E^{P_n^\star}(y,\cdot)$ *is at most* $\frac{\mathsf{poly}(n)}{p}$*, where again the expectation is over the choice of the coins* r *used by* E*.*

Note that if L_U is a language with polynomial-size witnesses, then this property implies proof of knowledge property [FFS88]. The proof of knowledge extractor will first run $\langle P_n^\star, V(y)\rangle$ once: if V rejects, it will output $\perp$, otherwise it will choose a random string r to be used as randomness to run $E_r^{P^\star}(y,i)$ for each bit i of the witness, and then output the result when the obtained witness is correct, aborting otherwise. This extractor will have overwhelming success probability when $p(y)$ is non-negligible, and expected running time $p(y) \cdot \frac{|y|^c}{p(y)} = |y|^c$, which is clearly polynomial.

Note also that we could have given a definition that explicitly requires the extractor to run in expected polynomial time, essentially resembling the definitions of the proof of knowledge [FFS88] and weak proof of knowledge [BG02] properties. However, we prefer the above formulation as it makes clear that we can differentiate between (item 1) runs on which the extractor inadvertently produces a bad witness (recall that when the witness is not polynomial-sized, it may not be possible to efficiently identify invalid witnesses) and (item 2) runs on which the extractor aborts (e.g., when the interaction with $P_n^\star$ produces an invalid proof or when the extractor gives up in order to keep polynomial is expected running time).

2.2 CRHFs ⇒ Constant-Round UAQK

Our approach will be to construct a UAQK out of a regular UA (i.e., enjoying just the weak proof of knowledge property). As mentioned in Section 1.1, there are two difficulties when using the weak proof of knowledge property: (1) we must somehow estimate a lower bound on the success probability of the prover in order to determine which extractor to use, and (2) the running time of the extractor may grow faster than what we would like as compared with the success probability of the prover (here we will use the UA construction of [BG02], which gives an extractor with running time $(\frac{|y|}{\alpha})^d$ where α is a lower bound on the success probability of $P^\star$).

The first issue we will deal with within the design of our extractor; it essentially requires balancing the accuracy of the estimate with the need to compute it in expected polynomial time. In order to address the second issue, we will attempt to increase the success probability

[1] Here we assume for simplicity that if a witness w is expanded by appending any sequence of zeros, the resulting value is still a valid witness, so that $\hat{w}_i$ is always defined for all $i \in I$.

of the adversarial prover. Essentially, we will run some instances of that UA sequentially, and accept only if all instances are accepted by the verifier. Then, if the prover succeeds in all instances with probability p, we will argue that there must be at least one instance in which it succeeds with significantly higher probability. If we use this instance, then we can run the extractor with a higher lower bound, and thus obtain a more efficient extractor.

The resulting construction goes as follows. We first ask the prover to commit to a Merkle hash of its witness, and then we run the UA several times sequentially. In each UA, the prover proves both that the statement is true, and that the witness used is consistent with the given commitment, i.e., it proves weak knowledge of a witness and an authentication chain for each bit of the witness. (This will allow us to verify that parts of the witness are consistent with the initial commitment without having to extract the entire witness.)

The actual UAQK. Our construction uses as a subprotocol a universal argument $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$ satisfying the conditions described in Theorem 1 for the universal language L_U. Let $\ell = d+2$ where d is the value of the constant defined in Theorem 1 for this UA. We construct a UAQK $\langle P_{new}(y,w), V_{new}(y)\rangle$ for language L_U as depicted in Fig. 1.

2.3 Proving the Quasi-Knowledge Property

Intuition. At a high level, the proof proceeds as follows: Let p be the probability that the adversarial prover $P^\star$ convinces V_{new} to accept. Then for $j = 1, \ldots, \ell$, let p_j be the probability that V_{new} accepts the j-th internal UA instance conditioned on the event that it accepted all previous instances. Then the key observation is that $p = \prod_{j=1}^{\ell} p_j$, so we are guaranteed that for some j^*, $p_{j^*} \geq p^{1/\ell}$. Now, if we could estimate p_{j^*}, and identify a UA instance where $P^\star$ succeeds with probability roughly p_{j^*}, then we could run the UA extractor with this lower bound, and obtain an extractor with success probability roughly $\frac{p^{1/\ell}}{q(|y|)}$ (here q is the polynomial referred to as $q^\star$ in the discussion in Theorem 1), and running time roughly $(\frac{|y|}{p_{j^*}})^d = \frac{|y|^d}{p^{d/\ell}}$. Then, because we need an extractor with overwhelming success probability, we will run this extractor approximately $\frac{q(|y|)}{p^{1/\ell}}$ times, to ensure that at least one run will produce bits of a valid implicit witness.[2] The final result will have success probability nearly 1, and running time roughly $\frac{|y|^d}{p^{d/\ell}} \cdot \frac{q(|y|)}{p^{1/\ell}} = \frac{|y|^d q(|y|)}{p^{(d+1)/\ell}} = \frac{|y|^d q(|y|)}{p}$ (the last equality follows from our choice of ℓ). [3]

The main challenge in this process is finding a UA instance where $P^\star$ has success probability roughly $p^{1/\ell}$, and estimating this probability. Essentially, for each j, we want to find a starting state where $P^\star$ is about to begin the j-th UA, and where $P^\star$'s success probability in that proof is roughly p_j; then we need to identify and take the best of these states (i.e.,

[2] There is some subtlety here in that we must guarantee that we can always recognize which set of extracted bits is the correct one (the one which is consistent with some valid witness). To do this we make use of the prover's initial commitment to the witness - if the extracted bits are consistent with the initial commitment, then we assume that they are the correct ones.

[3] In fact this is slightly inaccurate, as we also need to guarantee, even when we run this extractor many times and boost $P^\star$'s success probability a bit more, none of these times takes too long. We do this by estimating a reasonable upper bound for the running time of E based on our estimate of P^*'s success probability, and stopping E early if it runs more than this number of steps. As a result, we have to run E a few more times, and we set $\ell = d + 2$.

Players: prover P_{new}, verifier V_{new}.
Common input: the statement $y \in L_U$, such that $|y| = n$.
Input of P_{new}: witness w for $y \in L_U$.
Tools: a family $\{\mathcal{H}_n\}$ of CRHFs, a statistically hiding trapdoor commitment scheme $(\mathsf{SHTGen}, \mathsf{SHCom}, \mathsf{SHVer}, \mathsf{SHTCom}, \mathsf{SHTDec})$, a constant-round statistical ZKAoK $\langle P_{szk}(\cdot,\cdot), V_{szk}(\cdot)\rangle$, a constant-round WI universal argument $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$ (as defined in Theorem 1) with parameter d.

1) $V_{new} \to P_{new}$: V_{new} picks $h \stackrel{R}{\leftarrow} \{\mathcal{H}_n\}$, computes $(\eta, t) \stackrel{R}{\leftarrow} \mathsf{SHTGen}(1^n)$ and sends (h, η) to P_{new}.

2) $V_{new} \leftrightarrow P_{new}$: V_{new} runs P_{szk} to prove to P_{new} running V_{szk} knowledge of a trapdoor t for η.

3) $P_{new} \to V_{new}$: P_{new} encodes the witness w as follows: Let $s = |w|$ be the length of the original witness w, and let $w = w_1, \ldots, w_s$ be the bitwise representation of w. Then P_{new} uses h to form a Merkle hash tree over all the bits of w treating each bit as a leaf node in the tree. Let *root* be the root of this tree, and for bit w_i of witness w, let $auth_i$ be the appropriate authentication chain (i.e., all of the values on the path from w_i to *root*, and all of the siblings of those values).

   Next P_{new} computes $(C_s, \mathsf{dec}_s) \leftarrow \mathsf{SHCom}_\eta(s)$ and $(C_{root}, \mathsf{dec}_{root}) \leftarrow \mathsf{SHCom}_\eta(root)$.

   Finally, P_{new} forms the new witness $w' = s \circ \mathsf{dec}_s \circ root \circ \mathsf{dec}_{root} \circ w_1 \circ auth_1 \circ \ldots \circ w_s \circ auth_s$. Now, let $y' = C_s \circ C_{root} \circ y$ and let L'_U be the language which accepts (y', w') iff $w' = s \circ \mathsf{dec}_s \circ root \circ \mathsf{dec}_{root} \circ w_1 \circ auth_1 \circ \ldots \circ w_s \circ auth_s$ such that (1) $s < 2^{|y|}$, (2) (s, dec_s) is a valid decommitment for C_s and $(root, \mathsf{dec}_{root})$ is a valid decommitment for C_{root} with parameter η, (3) for all $i = 1 \ldots s$, $auth_i$ is a valid authentication chain for w_i with respect to *root* and hash function h, and (4) $w = w_1 \ldots, w_s$ is such that $(y, w) \in L_U$.

   P_{new} sends (C_s, C_{root}) to the verifier.

4) $P_{new} \leftrightarrow V_{new}$:

   For $j = 1, \ldots, \ell = d + 2$, sequentially:

   4.j) P_{new} and V_{new} run the universal argument $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$ for this new language L'_U. P_{new} will run $P_{UA}(y', w')$ while V_{new} will run $V_{UA}(y')$, where $y' = C_s \circ C_{root} \circ y$.

   V_{new} outputs 1 iff all instances of $V_{UA}(y')$ accept.

Figure 1: Universal Argument of Quasi Knowledge $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$.

p_{j^*}). We do this as follows: First, for each j, we record many states for $P^\star$ where $P^\star$ has successfully completed the first $j-1$ UAs and has a reasonable chance of completing the next UA. In Appendix B we show that with overwhelming probability, one of these states will be such that $P^\star$ has probability at least $p_j/2$ of successfully completing the next UA, and at the

same time that the time required to collect all these states is at most $O(\mathsf{poly}(|y|)/p)$. Next, we attempt to identify one such state, and to estimate the corresponding success probability. We do this by running $P^\star$ from each state many times, and counting how long it takes for $P^\star$ to complete $|y|$ proofs starting from that state. We interleave the counting for all of the states corresponding to the j-th proof to ensure that we can stop as soon as one has resulted in $|y|$ successful proofs. (This allows us to avoid running for too long just because one candidate state was bad.) Finally, we consider the best states for $j = 1, \ldots, \ell$, select the one with the highest estimated success probability, and use it as described above.

Our extractor. To summarize, our extractor works as follows:

1. For each $j = 1, \ldots, \ell$:
   (a) Collect $n = |y|$ candidate states, where $P^\star$ has successfully completed the first $j-1$ proofs and has a reasonable chance of successfully completing the next one.
   (b) Repeatedly run the next UA from all the above n states, and identify the state $beststate_j$ that is the first one that reaches n accepting executions of the next $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$. Let m_j be the number of such executions from that state.
2. Given the above m_j for $j = 1, \ldots, \ell$, let $\hat{j}$ be the index of the UA where $m_{\hat{j}}$ is minimal. $\frac{n}{2m_{\hat{j}}}$ is an estimate of a lower bound on the adversarial prover's success probability $p_{\hat{j}}$ in state $beststate_{\hat{j}}$.
3. Run approximately $q(|y|)/p_{\hat{j}}$ instances of the UA extractor. Return the result that agrees with the initial commitment sent by the prover.

A more formal description of the extractor and a more detailed analysis can be found in the Appendix B. We also note that for technical reasons, we also need a statistically hiding trapdoor commitment scheme and a statistical zero-knowledge argument of knowledge of the trapdoor, which help us to prove the witness indistinguishability of our construction.

Theorem 2 *Under the assumption that a family of collision-resistant hash functions exists, then the protocol depicted in Fig. 1 is a constant-round witness-indistinguishable UAQK.*

Similarly to our construction, one can also obtain a zero-knowledge UAQK by relying on our techniques on top of the zero-knowledge UA of [Bar04a].

3 Secure Database Commitments

We consider the notion of a secure database commitment such that each element x appears at most once. As in [MRK03], we will consider a formulation that captures both sets and databases. For a set S, we can define $Db[x]$ to be 1 if x appears in the set and $\perp$ if it does not. More generally, for a database of pairs (x, y) (such that each x appears in at most one pair), we define $Db[x]$ to be y if a pair (x, y) appears in the database, and $\perp$ if no pair $(x, \cdot)$ appears. Since the difference between the two primitives is almost cosmetic, we will use the terms of sets and database interchangeably. We will assume that each element x belongs to $\{0,1\}^L$ and L is polynomial in the security parameter k. Thus, by the requirements

above, this means that the database contains at most 2^L entries. We will make no other requirement on the size of the database.

The functionality in question is $\mathcal{F}_{DbCom}$ and is the natural extension of the standard commitment functionality, but in $\mathcal{F}_{DbCom}$ we must hide the size of the committed message. Furthermore, we will allow for partial openings, in which a verifier can choose a value x, and the the prover will reveal the value of $Db[x]$. (These partial openings must hide all other information about the database, thus there are also similarities with the zero-knowledge functionality.)

Ideal functionality $\mathcal{F}_{DbCom}$. The database commitment functionality consists of two phases: one in which a prover commits to a database, and a second in which the prover answers queries about that database. Here we will generalize this definition to say that the prover can commit to any database for which he knows an implicit representation. In particular, we will allow the prover to commit to a circuit which evaluates the desired database: it takes as input a string x, and returns $Db[x]$ (which will be $\perp$ if x is not in the database).

The formal specification is given in Fig. 2. For simplicity we assume that all queries made by the honest verifier V are fixed in advance. The definition can be generalized to adaptive queries, where the next query depends on the output of the previous ones. Our construction will satisfy this stronger definition as well.

Remark. Note that our definition of $\mathcal{F}_{DbCom}$ does not exclude the possibility that a player is able to commit to a very large (e.g., superpolynomial sized) set, for which he only knows some implicit representation. However, we argue that this does not violate any intuitive security requirement of secure database commitments since the critical requirement is that the prover is committed to some database (whose size is not a priori upperbounded by any fixed polynomial) at the end of the commitment phase, and that it must answer correctly according to this database during the query phase. The construction that we present in Section 4 will require that the honest sender knows an explicit representation of the database he is committing to (i.e., the honest sender runs on input a list of pairs representing the database it wants to commit to).[4] Obtaining a scheme where a real-world honest prover is allowed to have as input a polynomial-sized implicit representation that corresponds to a super-polynomial number of elements is also an interesting direction, and we defer it to future research.

Defining security. In the ideal execution, when initiated with input, parties interact with $\mathcal{F}_{DbCom}$. The adversary Sim has control of a party and thus can deviate from the prescribed computation, while the other player H_P follows the prescribed ideal computation. We denote by $IDEAL^{\mathcal{F}_{DbCom}}_{H_P,\mathsf{Sim}}(k,x,z)$ the output of Sim and H_P where Sim runs on input an auxiliary input z, and uniform randomness, while H_P runs on input x. Moreover we denote by $\{IDEAL^{\mathcal{F}_{DbCom}}_{H_P,\mathsf{Sim}}(k,x,z)\}$ the corresponding ensemble of distributions, with $k \in \mathcal{N}$ and $z \in \{0,1\}^*$.

In the real execution parties run a given protocol π. Let H_P be the honest party and $\mathcal{A}$ be the adversary, which has control of the other party. We denote by $EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)$

[4] Note that an explicit representation can always be efficiently converted into an implicit representation $\mathcal{C}_{Db}$.

FUNCTIONALITY $\mathcal{F}_{DbCom}$

Players: Prover P, Verifier V.
Input of P: circuit $\mathcal{C}_{Db}$.
Input[a] **of V:** $x_1, \ldots, x_m$ with $m = \mathsf{poly}(k)$.
Computation of $\mathcal{F}_{DbCom}$:

1. Upon receiving $(\texttt{Commit}, \mathcal{C}_{Db})$ from P, if $(\texttt{Commit}, \mathcal{C}_{Db})$ was already received or if $\mathcal{C}_{Db}$ does not describe a circuit with the appropriate number of input and output wires then ignore, otherwise record $(\mathcal{C}_{Db})$, send $(\texttt{Receipt})$ to V.

2. Upon receiving $(\texttt{Query}, x)$ from V, if $(\texttt{Commit}, \mathcal{C}_{Db})$ was previously received from P, then send $(\texttt{Query}, x)$ to P, otherwise ignore.

3. Upon receiving $(\texttt{Open}, 1, x)$ from P, if $(\texttt{Query}, x)$ was not previously sent to P then ignore, otherwise evaluate the circuit $\mathcal{C}_{Db}$ on input x to obtain output $Db[x]$, and send $(\texttt{Open}, x, Db[x])$ to V.

4. Upon receiving $(\texttt{Open}, 0, x)$ from P, if $(\texttt{Query}, x)$ was not previously sent to P then ignore, otherwise send $(\texttt{Open-Abort}, x)$ to V.

5. Upon receiving $(\texttt{Halt})$ from V, if the same message was previously received from V then ignore, otherwise send $(\texttt{Halt})$ to P.

Computation of P:

1. Upon activation, send $(\texttt{Commit}, \mathcal{C}_{Db})$ to $\mathcal{F}_{DbCom}$.

2. Upon receiving $(\texttt{Query}, x)$ from $\mathcal{F}_{DbCom}$, send $(\texttt{Open}, 1, x)$ to $\mathcal{F}_{DbCom}$.

3. Upon receiving $(\texttt{Halt})$ from $\mathcal{F}_{DbCom}$ go to Output.

Computation of V:

1. Upon receiving $(\texttt{Receipt})$ from $\mathcal{F}_{DbCom}$ send $(\texttt{Query}, x_1)$ to $\mathcal{F}_{DbCom}$.

2. Upon receiving $(\texttt{Open}, x_i, y_i)$ from $\mathcal{F}_{DbCom}$, if $0 < i < m$ then send $(\texttt{Query}, x_{i+1})$ to $\mathcal{F}_{DbCom}$.

3. Upon receiving $(\texttt{Open}, x_m, y_m)$ or $(\texttt{Open-Abort}, x)$ from $\mathcal{F}_{DbCom}$, send $(\texttt{Halt})$ to $\mathcal{F}_{DbCom}$ and go to Output.

Output:

P: Output $(x_1, \ldots, x_t)$ where $t = \mathsf{poly}(k)$ and x_i for $0 < i \leq t$ is such that $(\texttt{Query}, x_i)$ is the i-th message received from $\mathcal{F}_{DbCom}$.

V: Output $((x_1, y_1), \ldots, (x_{m'}, y_{m'}))$, where $x_1, \ldots, x_m$ was the verifier's input, and each y_i for $0 < i \leq m'$ was part of a message $(\texttt{Open}, x_i, y_i)$ received from $\mathcal{F}_{DbCom}$ while y_i is the empty string in case $(\texttt{Open-Abort}, x_i)$ has been received from $\mathcal{F}_{DbCom}$.

[a] We stress that inputs can also be computed adaptively.

Figure 2: Ideal computation of functionality $\mathcal{F}_{DbCom}$.

the output of $\mathcal{A}$ and H_P where $\mathcal{A}$ runs on input an auxiliary input z, and uniform randomness, while H_P runs protocol π on input x and uniform randomness. Moreover we denote by $\{EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)\}$ the corresponding ensemble of distributions, with $k \in \mathcal{N}, x \in \{0,1\}^{\mathsf{poly}(k)}$ and $z \in \{0,1\}^*$.

Definition 2 *A protocol π securely realizes the functionality $\mathcal{F}_{DbCom}$ if for any real-world adversary $\mathcal{A}$ there exists an ideal-world adversary* Sim *such that for every sufficiently large k, for every $x \in \{0,1\}^{\mathsf{poly}(k)}$ and for every $z \in \{0,1\}^*$, $\{EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)\}$ and $\{IDEAL^{\mathcal{F}_{DbCom}}_{H_P,\mathsf{Sim}}(k,x,z)\}$ are computationally indistinguishable.*

4 Constant-Round Secure Database Commitments

In this section we present a constant-round protocol that securely realizes the $\mathcal{F}_{DbCom}$ functionality. Our construction uses a constant-round zero-knowledge argument of knowledge (ZKAoK) for all NP, a constant-round WI universal argument of quasi knowledge, a 2-round trapdoor commitment scheme, and a zero-knowledge set scheme with two additional properties. As all of these ingredients can be instantiated through CRHFs, this gives a secure realization of $\mathcal{F}_{DbCom}$ based only on CRHFs. An additional feature of our protocol is that the amortized round complexity of a proof is optimal (i.e., proofs are actually non-interactive).

Zero-knowledge sets. We start by reviewing a major building block for our construction: zero-knowledge sets. At a high level, a non-interactive zero-knowledge set scheme is composed of four algorithms as follows: an algorithm ZKSSetup, which generates some parameters, an algorithm ZKSCom, which commits to a database in a size independent way, an algorithm ZKSProve, which allows the owner of the database to prove that a particular query response was consistent with the committed database, and an algorithm ZKSVerify for verifying such proofs. We will also use "ZKS proof system" to refer to the interaction between ZKSProve and ZKSVerify.

We will use non-interactive zero-knowledge sets as a building block in our construction of secure database commitment. In Appendix A.4 we give a definition for zero-knowledge sets which is similar to that in [CHL+05]. We also define a somewhat stronger hiding property (which we call *special zero knowledge*), in which zero knowledge holds for all valid parameters output by the simulator set-up algorithm, and the simulated commitment is an honest commitment to an empty database. In Appendix C we will discuss how to achieve these properties based on CRHFs. We finally stress that even though non-interactive zero-knowledge sets have been defined in the common reference string model, we will use them in the standard model by using interaction.

Intuition for our protocol. The basic idea behind this construction is fairly straightforward: we give a concise commitment to the database (using the ZKS commitment algorithm), and use a universal argument of quasi knowledge to prove knowledge of a valid opening. Then we can use the ZKS proof system to respond to queries.

However, we need an additional property from the ZKS proof system to make this work. The issue is that when we are dealing with a corrupt prover, we need to be able to extract a circuit representation of the committed database from the UAQK (by which we mean a

circuit which on input x returns $Db[x]$). On the other hand, recall that the UAQK only allows us to query the witness one bit at a time. Thus, we need the ZKS to have the additional property that the committer can generate a well-formed witness string. In particular, we need to guarantee that the witness will be in a format such that we can efficiently extract $Db[x]$ for any x given only the UAQK extractor which allows queries to individual bits of the witness. Furthermore, in order to argue that the responses in the query phase must be consistent with the extracted database, we also need to be able to efficiently extract a ZKS proof for each x. This must again be done by just querying the circuit representation. Thus, we will define the notion of a ZKS scheme that *allows local witnesses* to capture these requirements.

Definition 3 *A ZKS proof system* $(\mathsf{ZKSSetup}, \mathsf{ZKSCom}, \mathsf{ZKSProve}, \mathsf{ZKSVerify})$ allows local witnesses *if there exist additional polynomial-time deterministic algorithms* $\mathtt{FormWitness}, \mathsf{TMVer}, \mathsf{Eval}, \mathsf{PfGen}$ *such that the following properties hold:*

Witness verifiability. For any sufficiently large k, for all polynomial sized Db, and random strings r,

$$\Pr[\mathtt{ZKSPAR} \leftarrow \mathsf{ZKSSetup}(1^k); zks \leftarrow \mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r); \\ w \leftarrow \mathtt{FormWitness}(\mathtt{ZKSPAR}, Db, r) : \mathsf{TMVer}(\mathtt{ZKSPAR}, zks, w) = 1] = 1.$$

Local evaluation. There exists a polynomial q such that for any sufficiently large k, for any $x \in \{0,1\}^k$ and $w \in \{0,1\}^{2^k}$, $\mathsf{Eval}(x, w)$ only accesses $q(k)$ bits of w and runs in time polynomial in k.

Local verification. There exists a polynomial q' such that for any sufficiently large k, $x \in \{0,1\}^k$ and $w \in \{0,1\}^{2^k}$, $\mathsf{PfGen}(x, w)$ only accesses $q'(k)$ bits of w and runs in time polynomial in k. Moreover for any $x \in \{0,1\}^k$ and $w \in \{0,1\}^{2^k}$ and polynomial sized zks,

$$\Pr[\mathtt{ZKSPAR} \leftarrow \mathsf{ZKSSetup}(1^k) : \mathsf{TMVer}(\mathtt{ZKSPAR}, zks, w) = 1 \wedge \\ \wedge \mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x, \mathsf{Eval}(x, w), \mathsf{PfGen}(x, w)) = 0] = 0.$$

The intuition of the above definition is that there must exist a procedure `FormWitness` that transforms the witness while still preserving the content of the database; TMVer verifies that this transformation has been correctly computed. Moreover one can extract the membership or non membership of an element and the corresponding proof by accessing only a polynomial number of bits of this witness, according to the algorithms Eval and PfGen. The reason why we allow w to be of superpolynomial size is that the adversary can bypass the procedure `FormWitness` and produce a circuit that implicitly represent a superpolynomial sized witness. The interesting property of our definition is that even in this case, Eval and PfGen will be efficient.

Previous ZKS schemes allow local witnesses. We now show that standard constructions of ZKS ([MRK03, CHL$^+$05, CDV06]) have this property. We will use ideas from the ZKS construction of [MRK03, CHL$^+$05]. We summarize only the main properties we need.

The constructions work by building a tree. For each x in the database, we parse x as bits $b_1, \ldots, b_L$. Then at position $b_1, \ldots, b_L$ in the tree (where $b_i = 0$ denotes the left branch and

$b_i = 1$ denotes the right branch at level i), we store a commitment v_x to the corresponding y (all commitments here use a special commitment scheme). We construct the tree from the bottom up. For every ancestor $x' = b_1, \ldots, b_i$ of such an x, we compute a commitment to a hash of the two children: $v_{x'} = \mathsf{Com}(h(v_{x'||0}||v_{x'||1}))$. If one of the children has not yet been specified (it is the root of a subtree with no leaves in the database), we set that child node to $\mathsf{Com}(\perp)$.

Then a proof for value $(x, y) \in Db$ can be formed by producing all sibling and ancestor nodes and openings for all ancestor commitments. A proof for value $x \notin Db$ is more complex, but it can be formed by first finding the root of the largest empty subtree containing x. The value of this root node (call it $x_\perp$) should be $\perp$. Then the proof will include all sibling and ancestor nodes of $x_\perp$, special openings of all ancestor commitments, and an additional value which can be constructed given x and the randomness used to form the commitment at $x_\perp$.

More formally there exists a procedure PfGen^* such that for every empty subtree, there exists a polynomial sized string *info*, such that for all leaves x in this subtree, $\mathsf{PfGen}^*(x, \mathit{info})$ produces output identical to the part of the output of $\mathsf{ZKSProve}$ corresponding to such a subtree. It is also possible to efficiently verify that *info* is a legitimate string for running $\mathsf{PfGen}^*(x, \mathit{info})$ on any leaf x belonging to the empty subtree.

We now sketch the necessary algorithms as defined in Definition 3:

`FormWitness` : will first form the ZKS tree as described above. Then we will transform this into a single linear witness. For each nonempty, non-$\perp$ node x in the tree we will form an entry consisting of the value of v_x, the values of its children, the opening of the commitment v_x, and the positions in this witness string where the entries for each of the children nodes begin. For each $\perp$ node, we form an entry with the commitment $v_\perp$ and the extra *info* necessary to open the commitment. The witness will begin with the bit position of the entry corresponding to the root commitment.

TMVer: will traverse the entire tree (stopping at the roots of empty subtrees), and verify all commitments and hashes. This algorithm can easily be described by a polynomial sized TM.

$\mathsf{Eval}(x)$: will follow the path through the tree corresponding to x, beginning with the root and continuing until it reaches either $\perp$ or a leaf node with value y, and return the result.

$\mathsf{PfGen}(x, w)$: will follow the path through the tree corresponding to x until it reaches an empty subtree or a leaf. If it is a leaf, it will return all of the ancestor and sibling nodes and the openings, as described above. If it is an empty subtree, it reads the accompanying value *info*, runs $\mathsf{PfGen}^*(x, \mathit{info}) \to \pi$ and returns all the ancestor and sibling nodes and openings, followed by π. In both cases the output for an honestly generated witness will be identical to the output of $\mathsf{ZKSProve}$.

Our construction. We will use a constant-round zero-knowledge argument of knowledge (ZKAoK) $\langle \mathcal{P}(\cdot,\cdot), \mathcal{V}(\cdot) \rangle$, a constant-round WI UAQK $\langle \mathsf{UAP}(\cdot,\cdot), \mathsf{UAV}(\cdot) \rangle$, a special ZKS $(\mathsf{ZKSSetup}, \mathsf{ZKSCom}, \mathsf{ZKSProve}, \mathsf{ZKSVerify})$ which allows local verification with zero knowledge simulator $(\mathsf{ZKSSimSetup}, \mathsf{ZKSSimCom}, \mathsf{ZKSSimProve})$ and a 2-round trapdoor commitment scheme $(\mathsf{TGen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{TDec}, \mathsf{Ver})$. A description of our scheme is found in Fig. 3.

Security Parameter: k.
Input to P: $Db = ((x_1, y_1), \ldots, (x_s, y_s))$ where $s = \mathsf{poly}(k)$, $x_i, y_i \in \{0,1\}^L$ for $0 < i \leq s$.
Input to V: $x'_1, \ldots, x'_m$, where $m = \mathsf{poly}(k)$ and $x'_i \in \{0,1\}^L$ for $0 < i \leq m$.
Commitment Phase:

1. $V \to P$: set $(\mathtt{ZKSPAR}, \mathtt{ZKSTRAP}) \leftarrow \mathsf{ZKSSimSetup}(1^k)$, $(\mathtt{crs}, \mathtt{aux}) \leftarrow \mathsf{TGen}(1^k)$ and send $(\mathtt{ZKSPAR}, \mathtt{crs})$ to P.

2. $V \leftrightarrow P$: V proves knowledge of $\mathtt{ZKSTRAP}, \mathtt{aux}$ (i.e., V and P run $\mathcal{P}((\mathtt{ZKSPAR}, \mathtt{crs}), (\mathtt{ZKSTRAP}, \mathtt{aux}))$ and $\mathcal{V}((\mathtt{ZKSPAR}, \mathtt{crs}))$ respectively). If P rejects the proof, then it aborts.

3. $P \to V$: pick $r \in \{0,1\}^k$, set $(c, \mathsf{dec}) = \mathsf{Com}(\mathtt{crs}, zks = \mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r))$ and send c to V.

4. $P \leftrightarrow V$: P execute $\mathtt{FormWitness}(\mathtt{ZKSPAR}, Db, r)$ to generate witness w. Let $\mathsf{UAP}, \mathsf{UAV}$ be a UAQK that proves quasi knowledge of a witness of the form $zks||\mathsf{dec}||w$, where zks, dec is a valid opening for commitment c under $\mathtt{crs}$, and w is such that $\mathsf{TMVer}(\mathtt{ZKSPAR}, zks, w)$ accepts. Then P will execute the code of UAP on input $c||\mathtt{ZKSPAR}||\mathtt{crs}$ with witness $zks||\mathsf{dec}||w$ while V executes the code of UAV on input $c||\mathtt{ZKSPAR}||\mathtt{crs}$. If UAV rejects, V aborts.

5. $P \to V$: open the commitment c by sending zks, dec to V. V runs $\mathsf{Ver}(\mathtt{crs}, c, \mathsf{dec}, zks)$ and if the output is 0, it aborts.

Query/Answer Phase:
For $i = 1, \ldots, m$ do:

6. $V \to P$: send x'_i to P.

7. $P \to V$: send $(y'_i = Db[x'_i], \pi = \mathsf{ZKSProve}(\mathtt{ZKSPAR}, x'_i, Db[x'_i], Db, r))$ to V. V outputs (x'_i, y'_i) if $\mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x'_i, y'_i, \pi) = 1$ and aborts otherwise.

Figure 3: Our scheme for secure database commitments.

Round optimality. Simply by inspection one can observe that the query/proof phase is non-interactive (optimal).

Security. The protocol depicted in Fig. 3 securely realizes the $\mathcal{F}_{DbCom}$ functionality, i.e., it is a constant-round protocol for secure database commitment. For lack of space the proof is presented in Appendix D.

Theorem 3 *If $\langle \mathcal{P}(\cdot,\cdot), \mathcal{V}(\cdot) \rangle$ is a ZKAoK, $(\mathsf{ZKSSetup}, \mathsf{ZKSCom}, \mathsf{ZKSProve}, \mathsf{ZKSVerify})$ is a special zero-knowledge set scheme which allows local witnesses, $\langle \mathsf{UAP}(\cdot,\cdot), \mathsf{UAV}(\cdot) \rangle$ is a WI UAQK and $(\mathsf{TGen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{TDec}, \mathsf{Ver})$ is a 2-round trapdoor commitment scheme, then the protocol depicted in Fig. 3 securely realizes the $\mathcal{F}_{DbCom}$ functionality.*

Corollary 1 *Under the assumption that there exists a family of CRHFs, then there exists*

an efficient protocol which securely realizes the $\mathcal{F}_{DbCom}$ functionality. This follows from the fact that all of the primitives mentioned in Theorem 3 can be realized based on CRHFs.

References

[ADCT11] Giuseppe Ateniese, Emiliano De Cristofaro, and Gene Tsudik. (If) size matters: Size-hiding private set intersection. In *PKC 2011*, volume 6571, pages 156–173. Springer, 2011.

[Bar01] Boaz Barak. How to Go Beyond the Black-Box Simulation Barrier. In *Proceedings of the 42nd Symposium on Foundations of Computer Science, (FOCS '01)*, pages 106–115. IEEE Computer Society Press, 2001.

[Bar04a] Boaz Barak. *Non-Black-Box Techniques in Cryptography.* PhD thesis, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, 2004.

[Bar04b] Boaz Barak. *Non-Black-Box Techniques in Cryptography, Ph.D. Thesis.* Weizmann Institute of Science, 2004.

[Bel02] Mihir Bellare. A note on negligible functions. *Journal of Cryptology*, 15:271–284, 2002.

[BG02] Boaz Barak and Oded Goldreich. Universal Arguments and Their Applications. In *IEEE Conference on Computational Complexity (CCC '02).* IEEE Computer Society Press, 2002.

[CDV06] Dario Catalano, Yevgeniy Dodis, and Ivan Visconti. Mercurial commitments: Minimal assumptions and efficient constructions. In *Third Theory of Cryptography Conference, TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 120–144. Springer, 2006.

[CFM08] Dario Catalano, Dario Fiore, and Mariagrazia Messina. Zero-knowledge sets with short proofs. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 433–450. Springer, 2008.

[CHL+05] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In *Advances in Cryptology – Eurocrypt '05*, volume 3494 of *Lecture Notes in Computer Science*, pages 422–439. Springer-Verlag, 2005.

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *J. Cryptology*, 1(2):77–94, 1988.

[Fis01] Marc Fischlin. Trapdoor commitment schemes and their applications, phd thesis. http://www.mi.informatik.uni-frankfurt.de/research/phdtheses/mfischlin.dissertation.2001.pdf, 2001.

[FS90] Uriel Feige and Adi Shamir. Zero-Knowledge Proofs of Knowledge in Two Rounds. In *Advances in Cryptology – Crypto '89*, volume 435 of *Lecture Notes in Computer Science*, pages 526–544. Springer-Verlag, 1990.

[GM06] Rosario Gennaro and Silvio Micali. Independent zero-knowledge sets. In *ICALP*, volume 4052 of *Lecture Notes in Computer Science*, pages 181–234. Springer, 2006.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. *SICOMP*, 18(6):186–208, 1989.

[GMW87] Oded Goldreich, Silvio Micali, and Avi. Wigderson. How to Play any Mental Game - A Completeness Theorem for Protocols with Honest Majority. In *19th ACM Symposium on Theory of Computing (STOC '87)*, pages 218–229, 1987.

[Gol04] Oded Goldreich. *Foundations of Cryptography - Volume II - Basic Applications*. Cambridge press, 2004.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *Advances in Cryptology - Crypto '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 201–215. Springer-Verlag, 1996.

[HT98] Satoshi Hada and Toshiaki Tanaka. On the existence of 3-Round zero-knowledge protocols. In *Advances in Cryptology – Crypto '98*, volume 1462, 1998.

[IP07] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594. Springer, 2007.

[LY10] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 499–517. Springer, 2010.

[MRK03] Silvio Micali, Michael Rabin, and Joe Kilian. Zero-knowledge sets. In *44th IEEE Symposium on Foundations of Computer Science (FOCS '03)*, pages 80–91, 2003.

[PX09] Manoj Prabhakaran and Rui Xue. Statistically hiding sets. In *CT-RSA*, volume 5473 of *Lecture Notes in Computer Science*, pages 100–116. Springer, 2009.

A Definitions and Tools in Details

A polynomial-time relation R is a relation for which it is possible to verify in time polynomial in $|x|$ whether $R(x, w) = 1$. Let us consider an NP-language L and denote by R_L the corresponding polynomial-time relation such that $x \in L$ if and only if there exists w such that $R_L(x, w) = 1$. We will call such a w a *valid witness for* $x \in L$. A *negligible* function $\nu(k)$ is a non-negative function such that for any constant $c < 0$ and for all sufficiently

large k, $\nu(k) < k^c$. We will call a positive function *non-negligible* if it is not a negligible function. We will call a positive function *overwhelming* if it can be described as $1 - \nu$ for some negligible function nu. We will denote by $\text{Prob}_r[\ X\ ]$ the probability of an event X over coins r.

Indistinguishability.

Definition 4 *Two ensembles of distributions $X = \{X_k\}$ and $Y = \{Y_k\}$ ranging over $\{0,1\}^{\mathsf{poly}(k)}$ are* computationally indistinguishable *if for any polynomial-sized circuit D there exists a negligible function ν such that*

$$\left|\text{Prob}[\ \alpha \leftarrow X_s : D(s,\alpha) = 1\ ] - \text{Prob}[\ \alpha \leftarrow Y_s : D(s,\alpha) = 1\ ]\right| < \nu(k).$$

Chernoff Bounds. We use the following versions of Chernoff Bounds.

Let $X_1, \ldots, X_n$ be independent Poisson trials with $\text{Prob}[\ X_i = 1\ ] = p_i$. Then if X is the sum of all X_i and if μ is $E[X]$, for any $\delta \in (0,1]$:

$$\text{Prob}[\ X < (1-\delta)\mu\ ] < \Big(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big)^{\mu}.$$

Let $X_1, \ldots, X_n$ be independent Poisson trials with $\text{Prob}[\ X_i = 1\ ] = p_i$. Then if X is the sum of all X_i and if μ is $E[X]$, for any $\delta > 0$:

$$\text{Prob}[\ X > (1+\delta)\mu\ ] < \Big(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\Big)^{\mu}.$$

Markov's inequality. Let X be a non-negative discrete random variable, and let a be a positive real number, then

$$\text{Prob}[\ X \geq a\ ] \leq \frac{E(X)}{a}.$$

A.1 Commitment Schemes

We now give definitions for several notions of commitment schemes. For readability we will use "for all m" to mean any possible message m of length polynomial in the security parameter. We start with the standard notion of commitment scheme with its two main variants (i.e., unconditionally binding and unconditionally hiding). Note that all definitions will use a commitment generator function that outputs the commitment parameters.

Definition 5 $(\mathsf{Gen}, \mathsf{Com}, \mathsf{Ver})$ *is a* commitment scheme *(CS, for short) if:*

- **efficiency:** Gen, Com *and* Ver *are polynomial-time algorithms;*
- **completeness:** *for all m it holds that*

$$\text{Prob}\Big[\mathtt{crs} \leftarrow \mathsf{Gen}(1^k); (\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}(\mathtt{crs}, m) : \mathsf{Ver}(\mathtt{crs}, \mathsf{com}, \mathsf{dec}, m) = 1\Big] = 1;$$

- **binding:** *for any polynomial-time algorithm* $\texttt{committer}$ *there is a negligible function* ν *such that for all sufficiently large* k *it holds that*

$$\text{Prob}\Big[\texttt{crs} \leftarrow \mathsf{Gen}(1^k); (\mathsf{com}, m_0, m_1, \mathsf{dec}_0, \mathsf{dec}_1) \leftarrow \texttt{committer}(\texttt{crs}) : m_0 \neq m_1 \text{ and } \mathsf{Ver}(\texttt{crs}, \mathsf{com}, \mathsf{dec}_0, m_0) = \mathsf{Ver}(\texttt{crs}, \mathsf{com}, \mathsf{dec}_1, m_1) = 1\Big] \leq \nu(k);$$

- **hiding:** *for any polynomial-time algorithm* $\texttt{receiver}$ *there is a negligible function* ν *such that for all* m_0, m_1 *where* $|m_0| = |m_1|$ *and all sufficiently large* k *it holds that* $\text{Prob}\Big[(\texttt{crs}, \mathsf{aux}) \leftarrow \texttt{receiver}(1^k); b \leftarrow \{0,1\}; (\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}(\texttt{crs}, m_b) : b \leftarrow \texttt{receiver}(\mathsf{com}, \mathsf{aux})\Big] < \frac{1}{2} + \nu(k)$.

If the binding property holds with respect to a computationally unbounded algorithm $\texttt{committer}$*, the commitment scheme is said* statistically binding*; if instead, the hiding property holds with respect to a computationally unbounded algorithm* $\texttt{receiver}$*, the commitment scheme is said* statistically hiding*. When* $\texttt{crs}$ *is clear from context, we often say "*m, dec *is a valid opening for* com*" to mean that* $\mathsf{Ver}(\texttt{crs}, \mathsf{com}, \mathsf{dec}, m) = 1$.

We now give the definition of trapdoor commitment scheme. In particular we strengthen the computational indistinguishability of the computed commitments so that it holds even in case the distinguisher takes as input the trapdoor information. This allows one to use the same commitment parameters for any polynomial number of commitments (and actually all our results hold in this stronger setting).

Definition 6 $(\mathsf{TGen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{TDec}, \mathsf{Ver})$ *is a* trapdoor *commitment scheme (TCS, for short) if* $\mathsf{TGen}(1^k)$ *outputs a pair* $(\texttt{crs}, \mathsf{aux})$*,* $\mathsf{Gen}_{\texttt{crs}}$ *is the related algorithm that restricts the output of* Gen *to the first element* $\texttt{crs}$*,* $(\mathsf{Gen}_{\texttt{crs}}, \mathsf{Com}, \mathsf{Ver})$ *is a commitment scheme and* TCom *and* TDec *are polynomial-time algorithms such that:*
- trapdoorness: *for all* m *and all pairs* $(\texttt{crs}, \mathsf{aux})$ *given in the output space of* TGen*, the probability distribution*

$$\{(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}(\texttt{crs}, m) : (\texttt{crs}, \mathsf{aux}, \mathsf{com}, \mathsf{dec}, m)\}$$

is statistically indistinguishable from

$$\{(\mathsf{com}', \mathsf{aux}_{\mathsf{com}'}) \leftarrow \mathsf{TCom}(\texttt{crs}, \mathsf{aux}); \mathsf{dec}' \leftarrow \mathsf{TDec}(\mathsf{aux}_{\mathsf{com}'}, m) : (\texttt{crs}, \mathsf{aux}, \mathsf{com}', \mathsf{dec}', m)\}$$

even when $\texttt{receiver}$ *and the distinguisher have access to* aux.

A.2 Interactive proof/argument systems with efficient prover strategies.

An *interactive proof (resp., argument) system* for a language L is a pair of probabilistic polynomial-time interactive algorithms P and V, satisfying the requirements of *completeness* and *soundness*. Informally, completeness requires that for any $x \in L$, at the end of the interaction between P and V, where P has as input a valid witness for $x \in L$, V rejects

with negligible probability. Soundness requires that for any $x \notin L$, for any (resp., any polynomial-sized) circuit $P^\star$, at the end of the interaction between $P^\star$ and V, V accepts with negligible probability. We denote by $\mathtt{out}_V(\langle P(x,w), V(x)\rangle)$ the output of the verifier V when interacting on common input x with prover P that also receives as additional input a witness w for $x \in L$. Moreover we denote by $\mathtt{out}_V(\langle P^\star, V(x)\rangle)$ the output of the verifier V when interacting on common input x with an adversarial prover $P^\star$. Given a security parameter k, we will use inputs x of length n where $n = \mathsf{poly}(k)$.

Formally, we have the following definition.

Definition 7 *A pair of interactive algorithms $\langle P(\cdot,\cdot), V(\cdot)\rangle$ is an* interactive proof (resp., argument) system *for the language L, if V runs in probabilistic polynomial-time and the following properties hold.*

Completeness: for every $x \in L$ and for every NP *witness w for $x \in L$*

$$\text{Prob}[\ \mathtt{out}_V(\langle P(x,w), V(x)\rangle) = 1\ ] = 1.$$

Soundness (resp. computational soundness): for every (resp., every polynomial-sized) circuit family $\{P_n^\star\}_{n\in N}$ there exists a negligible function $\nu(\cdot)$ such that

$$\text{Prob}[\ \mathtt{out}_V(\langle P_n^\star, V(x)\rangle) = 1\ ] < \nu(|x|).$$

for every $x \notin L$ of size n.

Since all protocols we give are actually argument (rather than proof) systems, we will now focus on argument systems only. Therefore even when we use the word "proof", we will actually mean "computationally-sound" proof. Also from now on we assume that all interactive algorithms are probabilistic polynomial-time, including the honest prover.

Arguments of knowledge. Informally, an argument system is an argument of knowledge if for any probabilistic polynomial-time interactive algorithm $P^\star$ there exists a probabilistic algorithm called the extractor, such that 1) the expected running time of the extractor is polynomial-time regardless of the success probability of $P^\star$; 2) if $P^\star$ has non-negligible probability of convincing a honest verifier for proving that $x \in L$, where L is an NP language, then the extractor with overwhelming probability outputs a valid witness for $x \in L$.

Zero knowledge. The classical notion of zero knowledge has been introduced in [GMR89]. In a zero-knowledge argument system a prover can prove the validity of a statement to a verifier without releasing any additional information. This concept is formalized by requiring the existence of an expected polynomial-time algorithm, called the *simulator*, whose output is indistinguishable from the view of the verifier.

We start by defining the concept of a view of an interactive Turing machine. Let A and B be two interactive Turing machines that run on common input x and assume that A and B have additional information z_A and z_B. We denote by $\mathsf{View}_B^A(x, z_A, z_B)$ the random variable describing the *view of* B; that is, B's random coin tosses, internal state sequence, and messages received by B during its interaction with A.

We are now ready to present the notion of a zero-knowledge argument.

Definition 8 *An interactive argument system $\langle P(\cdot,\cdot), V(\cdot)\rangle$ for a language L is computational (resp., statistical, perfect)* zero-knowledge *if for all polynomial-time verifiers $V^\star$, there exists an expected polynomial-time algorithm S such that the following ensembles are computationally (resp., statistically, perfectly) indistinguishable:*

$$\{\mathsf{View}^P_{V^\star(z)}(x,w)\}_{x\in L, w\in W(x), z\in\{0,1\}^\star} \text{ and } \{S(x,z)\}_{x\in L, z\in\{0,1\}^\star}.$$

Witness Indistinguishability Let $\Pi = \langle P(\cdot,\cdot), V(\cdot)\rangle$ be an argument system for language L. A *witness indistinguishability adversary* V' for Π receives as input $x \in L$, $w^0, w^1 \in W(x)$ and auxiliary information z.

V' interacts with machine $P^\star$ that has a bit $b \in \{0,1\}$ wired-in. $P^\star$ receives as input (x, w^0, w^1) and executes the code of the honest prover P on input (x, w^b). For $b \in \{0,1\}$, we denote by $\mathsf{WIExpt}^b_{P,V'}(x, w^0, w^1, z)$ the random variable describing the output of V' when interacting on input (x, w^0, w^1, z) with prover $P^\star$ running on input (x, w^0, w^1) and b is the wired-in bit of $P^\star$.

Definition 9 *Argument system $\Pi = \langle P(\cdot,\cdot), V(\cdot)\rangle$ for the language L is* witness indistinguishable *if for all probabilistic polynomial-time witness indistinguishability adversaries V' there exists a negligible function ν such that for all $x \in L$, all pairs of valid* NP *witnesses w^0, w^1 for $x \in L$ and all $z \in \{0,1\}^\star$*

$$|\mathrm{Prob}[\ \mathsf{WIExpt}^0_{P,V'}(x, w^0, w^1, z) = 1\ ] - \mathrm{Prob}[\ \mathsf{WIExpt}^1_{P,V'}(x, w^0, w^1, z) = 1\ ]| < \nu(|x|).$$

A.3 Universal Arguments

A universal argument is an interactive argument system for proving membership in **NEXP**. We give now the formal definitions.

Definition 10 *Given a description of a non-deterministic Turing machine M, a string x and a number t we say that $(\langle M, x, t\rangle) \in L_U$ if M accepts the input x within t steps. We can also consider the witness-relation for L_U denoted by R_U: the pair $(\langle M, x, t\rangle, w)$ is in R_U if M is a two-input deterministic Turing machine that accepts the input (x, w) within t steps. Moreover, we define $T_M(x, w)$ to be the number of steps made by M on input (x, w).*

Definition 11 *[BG02] A* universal-argument system *is an argument system $\langle P(\cdot,\cdot), V(\cdot)\rangle$ such that:*

Efficiency: *there exists a polynomial q such that for any $y = \langle M, x, t\rangle$, the total time spent by the verifier strategy V, on input y, is at most $q(|y|)$. In particular, all messages exchanged in the protocol have length smaller than $q(|y|)$.*

Completeness: *for every $(\langle M, x, t\rangle, w) \in R_U$, $\mathrm{Prob}[\ \mathtt{out}_V(\langle P(\langle M, x, t\rangle, w), V(\langle M, x, t\rangle)\rangle) = 1\ ] = 1$. Moreover, there exists a polynomial q such that for all $(\langle M, x, t\rangle, w) \in R_U$, it is guaranteed that $P(\langle M, x, t\rangle, w)$ runs in at most $q(T_M(x, w)) \le q(t)$ steps.*

Soundness: *For every polynomial-size circuit family $\{P^\star_n\}_{n\in N}$ and every string $y = \langle M, x, t\rangle \in \{0,1\}^n \backslash L_U$, it holds that $\mathrm{Prob}[\ \mathtt{out}_V(\langle P^\star_n, V(y)\rangle) = 1\ ] < \nu(n)$ where ν is a negligible function.*

Weak Proof of Knowledge: *For every positive polynomial q there exists a positive polynomial q' and a probabilistic polynomial-time algorithm E, called the* extractor*, such*

that the following holds. For every polynomial-size circuit family $\{P_n^\star\}_{n\in N}$*, and every* $y = \langle M, x, t\rangle \in \{0,1\}^n$*, if* $\text{Prob}[\ \mathtt{out}_V(\langle P_n^\star, V(y)\rangle) = 1\ ] \geq 1/q(|y|)$ *then*

$$\text{Prob}[\ \exists w = w_1 \cdots w_s \in R_U(y)\ \textit{s.t.}\ \forall i \in \{1,\ldots,s\}, E^{P_n^\star}(y,i) = w_i\ ] \geq 1/q'(|y|),$$

where $R_U(y)$ *is defined as* $\{w : (y,w) \in R_U\}$*,* $E^{P_n^\star}(\cdot,\cdot)$ *denotes that* E *has oracle access to* $P_n^\star$ *and the above probability is over the coins used by* E*.*

Loosely speaking, a universal argument is a computationally sound proof system for **NEXP**.

A.4 Zero-Knowledge Sets

In [MRK03] zero-knowledge sets and elementary databases were defined. We now present a definition for zero-knowledge sets which is similar to that in [CHL+05] but includes a special hiding property, in which zero knowledge holds for all valid parameters output by the simulator set-up algorithm. We also require that the simulated commitment is an honest commitment to an empty database. These additional properties can be achieved still using CRHFs which is the minimal assumption for ZK sets.

Definition 12 *Algorithms* $\mathsf{ZKSSetup}, \mathsf{ZKSCom}, \mathsf{ZKSProve}, \mathsf{ZKSVerify}$ *constitute a special zero-knowledge set scheme if they satisfy the following properties:*

Completeness. *For all databases* Db *and for all* x*,*

$$\Pr[\mathtt{ZKSPAR} \leftarrow \mathsf{ZKSSetup}(1^k); zks \leftarrow \mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r); \pi \leftarrow \mathsf{ZKSProve}(\mathtt{ZKSPAR}, x, Db[x], Db, r) : \mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x, Db[x], \pi) = 1] = 1.$$

Soundness. *For all* x *and for all probabilistic polynomial-time malicious provers* $P^\star$*,*

$$\Pr[\mathtt{ZKSPAR} \leftarrow \mathsf{ZKSSetup}(1^k); (zks, y, y', \pi, \pi') \leftarrow P^\star(\mathtt{ZKSPAR}) : y \neq y' \wedge \mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x, y, \pi) = 1 \wedge \mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x, y', \pi') = 1]$$

is negligible in k *(note that* y *and* y' *can be strings or* $\perp$*).*

Special Zero-Knowledge. *There exists algorithms* $\mathsf{ZKSSimSetup}, \mathsf{ZKSSimProve}$ *such that: (1) the distributions* $\{(\mathtt{ZKSPAR}, \mathtt{ZKSTRAP}) \leftarrow \mathsf{ZKSSimSetup}(1^k) : \mathtt{ZKSPAR}\}$ *and* $\{\mathtt{ZKSPAR} \leftarrow \mathsf{ZKSSetup}(1^k) : \mathtt{ZKSPAR}\}$ *are indistinguishable, and (2) for all* $(\mathtt{ZKSPAR}, \mathtt{ZKSTRAP}) \leftarrow \mathsf{ZKSSimSetup}$*, for all probabilistic polynomial-time malicious verifiers* $\mathcal{A}$*, the absolute value of the following difference is negligible in* k*.*

$$\Pr[(Db, state_A) \leftarrow \mathcal{A}(\mathtt{ZKSPAR}, \mathtt{ZKSTRAP}); zks \leftarrow \mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r) : \mathcal{A}^{\mathsf{ZKSProve}(\mathtt{ZKSPAR}, \cdot, Db[\cdot], Db, r)}(state_{\mathcal{A}}) = 1] -$$
$$\Pr[(Db, state_A) \leftarrow \mathcal{A}(\mathtt{ZKSPAR}, \mathtt{ZKSTRAP}); zks \leftarrow \mathsf{ZKSCom}(\mathtt{ZKSPAR}, \{\}, r) : \mathcal{A}^{\mathsf{ZKSSimProve}(\mathtt{ZKSTRAP}, \cdot, Db[\cdot], r)}(state_{\mathcal{A}}) = 1].$$

B Proof of Theorem 2

Theorem 2 *Under the assumption that a family of collision-resistant hash functions exists, then the protocol depicted in Fig. 1 is a constant-round WI UAQK.*

We prove this statement in two steps. First, we demonstrate that the protocol is a witness indistinguishable universal argument, then we show that it satisfies the quasi-knowledge property.

B.1 $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$ Is a Universal Argument

Theorem 4 *Under the assumption that a family of collision-resistant hash functions exists, the protocol $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$ depicted in Fig. 1 is a constant-round WI universal argument.*

Proof. Efficiency and completeness follow from the corresponding properties of $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$, which are preserved under sequential composition and from the corresponding properties of the commitment scheme and of the zero-knowledge argument of knowledge. The fact that CRHFs suffice, follow from Theorem 1 which provides the underlying constant-round witness indistinguishable universal argument with special properties under CRHFs, from the fact that constant-round statistical zero-knowledge arguments of knowledge exist for all NP under CRHFs, and from the fact that constant-round statistically hiding trapdoor commitments can be instantiated using CRHFs (see Appendix C).

Witness indistinguishability. We prove the witness indistinguishability of $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$, by contradiction. We assume that there exists a pair of different witnesses w_0, w_1 such that there exists a verifier $V^\star_{new}$ and a distinguisher D such that D distinguishes an execution of $\langle P_{new}(y,w_0), V^\star_{new}(y)\rangle$ from an execution of $\langle P_{new}(y,w_1), V^\star_{new}(y)\rangle$. We now show that hybrid games connect the two executions and therefore the success of D produces a contradiction of one of the underlying complexity-theoretic assumptions.

Game 1: this is the game where the prover is $P_{new}(y,w_0)$. Here the witness w_0 is used both for producing the commitments C_s, C_{root} of s, *root* and for producing the actual witness to be used in the executions of $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$.

Game 2: this is similar to Game 1, but we also require that $P_{new}(y,w_0)$ runs the extractor of the proof of knowledge of the trapdoor given by $V^\star_{new}$. The trapdoor however is not used by $P_{new}(y,w_0)$. It is easy to see that a distinguisher between Game 2 and Game 1 would contradict the proof of knowledge (specifically, the fact that the success probability of the extractor essentially corresponds to the success probability of the prover) property of the statistical zero-knowledge argument of knowledge in which the verifier proves knowledge of the trapdoor.

Game 3: this is similar to Game 2, but we also require that commitments are performed by using the trapdoor. Notice that commitments are compatible (in the statistical sense) both with the ones of $P_{new}(y,w_0)$ and with the ones of $P_{new}(y,w_1)$. The experiment continues still using w_0 for producing a witness in $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$. It is easy to see that a distinguisher between Game 3 and Game 2 would contradict the statistical trapdoorness of C_s, C_{root}.

Game 4: this is similar to Game 3, but this time w_1 is used for producing a witness in $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$. It is easy to see that a distinguisher between Game 3 and Game 4 would contradict the witness indistinguishability of $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$.

Game 5: this is similar to Game 4, but this time w_1 is used both for producing the commitments C_s, C_{root} of $s, root$ and for producing the actual witness to be used in the executions of $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$. It is easy to see that a distinguisher between Game 5 and Game 4 would break the statistical trapdoorness of the commitment scheme.

Game 6: this is similar to Game 5, but the trapdoor is not used and the commitments therefore correspond to w_1. It is easy to see that a distinguisher between Game 6 and Game 5 would break the statistical trapdoorness of the commitment scheme.

Game 7: this is similar to Game 6, but the trapdoor is not extracted. It is easy to see that a distinguisher between Game 7 and Game 6 would contradict the proof of knowledge (specifically, the fact that the success probability of the extractor essentially corresponds to the success probability of the prover) property of the statistical zero knowledge argument of knowledge in which the verifier proves knowledge of the trapdoor.

Since Game 7 corresponds to an execution of $\langle P_{new}(y, w_1), V^{\star}_{new}(y)\rangle$, we have shown that the existence of D contradicts the security of one of the complexity-theoretic assumptions used in our construction.

Soundness. Assume by contradiction that an adversary $P^{\star}$ can prove to V_{new} a false statement y with non-negligible probability p. By succeeding with probability p in $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$, $P^{\star}$ also succeeds with probability at least p in proving the first UA $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$. The instance of this UA is y' and is different than y, but it still includes the truthfulness of y. Indeed, the instance y' proves that the instance y is correct *and* moreover, one can also extract bits of the witness in expected polynomial time. Therefore, if y is a false instance then y' is a false instance too. It is therefore immediate the reduction from an adversary $P^{\star}$ that breaks the soundness of $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$ for a false instance y of its choice to an adversary $P^{\star}_{UA}$ that breaks the soundness of the underlying universal argument $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$ for a false instance y' of its choice. □

B.2 $\langle P_{\boldsymbol{new}}(\cdot,\cdot), V_{\boldsymbol{new}}(\cdot)\rangle$ Is a Universal Argument of Quasi Knowledge

We focus here on showing that the new argument system is an argument of quasi knowledge.

Theorem 5 *Under the assumption that a family of collision-resistant hash functions exists, the protocol $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot)\rangle$ depicted in Fig. 1 is an argument of quasi knowledge.*

PROOF. We must show that there exists an extractor E_{new} which satisfies the conditions of Definition 1. (Recall that E_{new} on input y, running with randomness r and with access to a polynomial-sized circuit $P^{\star}$ aims to provide an implicit representation of the witness.) We first present some notation and a couple useful lemmas based on Chernoff bounds, then describe our extractor E_{new} in detail in Fig. 4, and finally show in Propositions 1 and 2 that it satisfies the properties described in Definition 1.

Notation: We let $P^{\star} \in \{P^{\star}_n\}_{n\in N}$ be a potentially adversarial prover, y be an input statement, and $n = |y|$. Let $\sigma(n)$ be the combined bit length of $s, \mathsf{dec}_s, root, \mathsf{dec}_{root}, w_i, auth_i$. (Note that this must be polynomial.) We also let $T_{new}(n)$ be the running time of the UAQK

verifier V_{new}. As this is guaranteed to be polynomial in n (by definition of UA), we let c_{new} be a constant such that $T_{new}(n)$ is $O(n^{c_{new}})$.

Recall that step 4 of our UAQK protocol runs a UA protocol ℓ times sequentially. We will define the output of step $4.j$ in an execution of our UAQK protocol to be ACCEPT iff V_{UA} outputs 1 in step $4.j$.

For simplicity of notation, in what follows we denote by $p(n, y)$ the success probability of $P^\star$ when interacting with $V_{new}(y)$ where the probability is over the honest verifier's random coins. (We will omit n, y when it is clear from context.) Furthermore, let $p_1(n, y)$ be the probability that when we run the honest V_{new} algorithm with prover $P^\star$, the output of step 4.1 is ACCEPT. For $j = 2, \ldots, \ell$, let $p_j(n, y)$ be the probability that when we run V_{new} with $P^\star$, the output of step $4.j$ will be ACCEPT, conditioned on the event that the output of steps 4.1 - $4.(j-1)$ was also ACCEPT. Note that V_{new} accepts iff all ℓ instances of V_{UA} accept, i.e. if steps 4.1 - $4.\ell$ output ACCEPT, therefore $p(n, y) = \prod_{j=1}^{\ell} p_j(n, y)$ for all n, y.

Useful lemmas. The following lemmas will be useful in the analysis of our extractor.

Lemma 1 *Let E be an experiment which succeeds with probability p. Then if we run independent trials of this experiment $2n/p$ times, the probability that we obtain fewer than n successes is less than $(\frac{e}{2})^{-n}$.*

PROOF. Let X_i be a variable which is 1 if the ith experiment succeeds, and 0 if it fails, and let $X = \sum_{i=1}^{2n/p} X_i$. We want to show that $\text{Prob}[\ X < n\ ] < (\frac{e}{2})^{-n}$. Chernoff bounds tell us that if μ is $E[X]$, then for any $\delta \in (0, 1]$:

$$\text{Prob}[\ X < (1-\delta)\mu\ ] < \Big(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big)^{\mu}$$

In our case $E[X] = p \cdot 2n/p = 2n$. So if we consider $\delta = 1/2$, then we get

$$\text{Prob}[\ X < (1/2) \cdot 2n\ ] < \Big(\frac{e^{-1/2}}{(1/2)^{(1/2)}}\Big)^{2n}$$

$$\implies \text{Prob}[\ X < n\ ] < (\frac{e}{2})^{-n}$$

as desired. □

Lemma 2 *Let E be an experiment which succeeds with probability p. Then if we run independent trials of this experiment $\frac{n}{2p}$ times, the probability that we obtain at least n successes is less than $2(\frac{4}{e})^{-n/2}$.*

PROOF. Let X_i be a variable which is 1 if the ith experiment succeeds, and 0 if it fails, and let $X = \sum_{i=1}^{n/2p} X_i$. We want to show that $\text{Prob}[\ X > n-1\ ] < (\frac{4}{e})^{-n/2}$. Chernoff bounds tell us that if μ is $E[X]$, then for any $\delta > 0$:

$$\text{Prob}[\ X > (1+\delta)\mu\ ] < \Big(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\Big)^{\mu}$$

In our case $E[X] = p \cdot \frac{n}{2p} = \frac{n}{2}$. So if we consider $\delta = \frac{2(n-1)}{n} - 1$, then we get

$$\text{Prob}[\ X > \frac{2(n-1)}{n} \cdot \frac{n}{2}\ ] < \Big(\frac{e^{\frac{2(n-1)}{n}-1}}{(\frac{2(n-1)}{n})^{(\frac{2(n-1)}{n})}}\Big)^{n/2}$$

$$\implies \text{Prob}[\ X > n-1\ ] < \Big(\frac{e}{\frac{2(n-1)}{n}}\Big)^{\frac{2(n-1)}{n}\cdot n/2} e^{-n/2}$$

$$\implies \text{Prob}[\ X > n-1\ ] < \Big(\frac{en}{2(n-1)}\Big)^{n-1} e^{-n/2}$$

$$\implies \text{Prob}[\ X > n-1\ ] < \Big(\frac{n}{n-1}\Big)^{n-1}\Big(\frac{e}{2}\Big)^{n-1} e^{-n/2}$$

$$\implies \text{Prob}[\ X > n-1\ ] < e \cdot \Big(\frac{e}{2}\Big)^{-1}\Big(\frac{e^2}{4}\Big)^{n/2} e^{-n/2}$$

$$\implies \text{Prob}[\ X > n-1\ ] < 2\Big(\frac{e}{4}\Big)^{n/2}$$

as desired. Here the second to last implication follows because $(1+1/x)^x < e$ for all $x \geq 1$. □

Now we argue that E_{new} satisfies the UAQK property described in Definition 1. We must show that (1) ${E_{new}}^{P^\star}(y,i)$ has running time that is independent of i, and expected running time $\frac{\mathsf{poly}(n)}{p}$, for some fixed polynomial poly, and (2) that there is some negligible function ν such that whenever $p > \nu(n)$, then for any polynomial-time generated set of indices I, with overwhelming probability $1-\nu(n)$, $w_i \leftarrow {E_{new}}^{P^\star}(y,i)$ is such that all outputs $\{w_i\}_{i\in I}$ are consistent with some valid witness w.

Outline of the proof. The high level structure of our proof will be as follows. First we will analyze the running time of our extractor. We will argue in Claim 1 that by Chernoff bounds it will not take too long to find n successful states, so step 1a will have the appropriate expected running time. Then we argue, again by Chernoff bounds, that the expected value of $m_{\hat{j}}$ (Lemma 4), and so the expected running time of step 1b will also be as desired (Claim 2). Finally, some arithmetic shows that as long as all values $m_{\hat{j}}$ are not too large, the running time of step 2 will also be appropriate (Claim 3). Combining these three claims proves that the running time is as desired (Proposition 1).

Next, we analyze the success probability of our extractor. First, we argue by Markov's inequality and the properties of the UA considered in Theorem 1, that if $\alpha = \frac{n}{2m_{\hat{j}}}$ is in fact a lower bound for the success probability of $P^\star$, then with overwhelming probability we will get at least one random tape r such that E_r both finishes within γ steps and produces valid bits when we try to extract w_i and its associated authentication chain (Claim 5). Next, we show again by Chernoff bounds that with overwhelming probability, α will be a good lower bound for the success probability of $P^\star$ (Claim 6). Then we use these claims to argue that, there a negligible function such that whenever $p > \nu(n)$ is non-negligible, E_{new} will produce correct output on all indices chosen by S with probability $1-\nu(n)$ (Claims 7,

${E_{new}}^{P^\star}(y,i)$.
For $j = 1 \ldots \ell$ repeat the following steps.

1. Run $\langle V_{new}(y), P^\star \rangle$, pausing after step 4.$(j-1)$ to record the state of P^* and V_{new} at that point. If steps 4.1 - 4.j all output ACCEPT, then store the recorded state as $state_1^{(j)}$. Rewind $P^\star$ and run $\langle V_{new}(y), P^\star \rangle$ again from the beginning (using new random coins for V_{new}), and continue this process until n states $state_1^{(j)}, \ldots, state_n^{(j)}$ have been collected.

   If at any point the above step requires running the UAQK protocol more than $(\frac{e}{2})^n$ times, then immediately halt and output $\perp$.

2. Restart $\langle V_{new}(y), P^\star \rangle$ from state $state_1^{(j)}$ (which corresponds to the end of step 4.$(j-1)$ of the protocol), and complete the protocol (using new random coins for steps 4.j - 4.ℓ). Record whether step 4.j outputs ACCEPT, then repeat the same task with $P^\star$ and V_{new} starting from states $state_2^{(j)}, \ldots, state_n^{(j)}$. Restart P^* and V_{new} again from $state_1^{(j)}$, and continue this process, keeping track of how many times we restart from $state_1^{(j)}$. For each $u = 1, \ldots, n$, keep a counter which records how many times $state_u^{(j)}$ results an execution which outputs ACCEPT on step 4.j. When the first of these counters reaches n, store the associated state $state_u^{(j)}$ as $beststate_j$. Also store as value m_j the number of times we had to restart P^* from state $state_1^{(j)}$.

   If at any point the above step requires running the UAQK protocol more than $(\frac{e}{2})^{n/\ell}$ times, then immediately halt, and output $\perp$.

Let $\hat{j}$ be such that $m_{\hat{j}}$ is minimized. Let E be an extractor (given by the weak proof of knowledge property) for $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot) \rangle$, and let d, q be the associated constant and polynomial as described in Theorem 1.[a] Compute $\alpha = \frac{n}{2m_{\hat{j}}}$, $\Theta = 4 \cdot m_{\hat{j}} \cdot q(n)$, and $\gamma = 2(2m_{\hat{j}})^{\ell-1} q(n)$. Let $P^\star_{best}$ represent $P^\star$ restarted from state $beststate_{\hat{j}}$. Parse the rest of the random tape r into Θ values $r_1, \ldots r_\Theta$, and repeat the following process Θ times.
For each iteration $\theta = 1 \ldots \Theta$: For each of the $\sigma(n)$ values for x representing one of the bit positions of $s, \mathsf{dec}_s, root, \mathsf{dec}_{root}, w_i, auth_i$, use coins r_θ for E and run $E^{P^\star_{best}}(\alpha, y', x)$ for exactly γ steps[b]. If for any of the $\sigma(n)$ values, E does not halt in that time, consider the output of this iteration to be $\emptyset$. Otherwise, verify the extracted $s, \mathsf{dec}_s, root, \mathsf{dec}_{root}$ with respect to C_s, C_{root}, and the extracted $w_i, auth_i$ with respect to $root$. If the verification succeeds, consider the output of this iteration to be w_i, otherwise consider it to be $\emptyset$.
Once all iterations have been completed, if any two iterations produced $w_i \neq w_i^*$ such that $w_i \neq \emptyset$ and $w_i^* \neq \emptyset$, or if all iterations produced $\emptyset$, then output $\emptyset$. Otherwise output the unique $w_i \neq \emptyset$ produced in one or more iterations.

[a] Recall that d is the constant which describes how the choice of α affects the running time of the resulting extractor E, and q is the polynomial which relates the success probabilities of $P^\star$ and E.

[b] If E runs for less time, we just wait until the equivalent amount of time has passed.

Figure 4: Extractor ${E_{new}}^{P^\star}(y,i)$ for the Universal Argument of Quasi Knowledge $\langle P_{new}(\cdot,\cdot), V_{new}(\cdot) \rangle$.

8). This follows from collision resistance of the hash function, the binding property of the commitment scheme, and statistical zero knowledge property of the proof system used to prove knowledge of the commitment trapdoor (and this is where we need S to be polynomial time). Combining these claims demonstrates the necessary success probability (Proposition 2).

Running time of the proof of quasi-knowledge extraction procedure. We begin by analyzing the running time of E_{new}.

Proposition 1 *There exists a polynomial* poly *such that for all $n \in \mathbb{N}$, for any $y \in L_U \cap \{0,1\}^n$, any $i \in \{0,1\}^n$, and any circuit $P_n^\star$, the running time of ${E_{new}}^{P^\star}(y,i)$ is independent of i and if $p(n) = \text{Prob}[\ \mathtt{out}_V(\langle P_n^\star, V_{new}(y)\rangle) = 1\ ] > 0$ then the expected value of this running time is at most $\frac{\mathsf{poly}(n)}{p(n)}$.*

Proof.

First we define some additional notation, and prove two lemmas about the values m_j and $state_u^{(j)}$ that E_{new} computes in step 1 that we will use in our analysis below.
Additional Notation: First, for any initial state $state^{(j)}$ corresponding to the end of step $4.(j-1)$, let $\rho_{state^{(j)}}$ be the probability that when we restart the protocol from state $state^{(j)}$, step $4.j$ outputs ACCEPT. Second, recall that E_{new} may halt in step 1a or 1b if it has to run the UAQK too many times. For simplicity of notation, if for any j, E_{new} halts before setting m_j, we will consider m_j to be $\lceil (\frac{e}{2})^{n/\ell}/n \rceil$.

Lemma 3 *For any $j \in 1,\ldots,\ell$ and any $u \in 1,\ldots n$, let $state_u^{(j)}$ be generated as described in step 1a of Fig. 1. Then* $\text{Prob}[\ \mathsf{max}_u(\rho_{state_u^{(j)}}) < p_j/2\ ]$ *is at most 2^{-n}.* [5]

Proof.

Fix a value of j. Let $D^{(j)}$ be the distribution over states that results from choosing random coins for the verifier and outputting the state *state* of $P^\star$ after step $4.(j-1)$ of the protocol. Let $E_1,\ldots,E_j$ be the event that steps 4.1, ..., 4.j resp. output ACCEPT during a given run of the UAQK. Now, the process in step 1a of the extractor E_{new} can be described as choosing $state \leftarrow D^{(j)}$ conditioned on $E_1,\ldots,E_j$. Let bad be the set of states such that $\rho_{state} < p_j/2$, and let $D^{(j)}$ be the distribution over states that results from step 1a of E_{new}.

Now, for any $u \in 1,\ldots,n$:

$$
\begin{aligned}
& \text{Prob}_{state \leftarrow D_{1a}^{(j)}}[\ \rho_{state_u^{(j)}} < p_j/2\ ] \\
= \ & \text{Prob}_{state \leftarrow D_{1a}^{(j)}}[\ state \in \mathsf{bad}\ ] \\
= \ & \text{Prob}_{state \leftarrow D^{(j)}}[\ state \in \mathsf{bad} | E_1 \wedge \ldots \wedge E_j\ ] \\
= \ & \text{Prob}_{state \leftarrow D^{(j)}}[\ state \in \mathsf{bad} \wedge E_j | E_1 \wedge \ldots \wedge E_{j-1}\ ] / \text{Prob}_{state \leftarrow D^{(j)}}[\ E_j | E_1 \wedge \ldots \wedge E_{j-1}\ ] \\
= \ & \text{Prob}_{state \leftarrow D^{(j)}}[\ E_j | state \in \mathsf{bad} \wedge E_1 \wedge \ldots \wedge E_{j-1}\ ] \cdot \text{Prob}_{state \leftarrow D^{(j)}}[\ state \in \mathsf{bad} | E_1 \wedge \ldots \wedge E_{j-1}\ ] \\
& / \text{Prob}_{state \leftarrow D^{(j)}}[\ E_j | E_1 \wedge \ldots \wedge E_{j-1}\ ] \\
\leq \ & \text{Prob}_{state \leftarrow D^{(j)}}[\ E_j | state \in \mathsf{bad} \wedge E_1 \wedge \ldots \wedge E_{j-1}\ ] / \text{Prob}_{state \leftarrow D^{(j)}}[\ E_j | E_1 \wedge \ldots \wedge E_{j-1}\ ] \\
< \ & p_j/2/p_j \\
= \ & 1/2
\end{aligned}
$$

[5] See notation section at the beginning of the proof of this section for a definition of p_j.

where the first equality follows by the definition of bad, the second equality follows by the definitions of $D_{1a}^{(j)}$ and $D^{(j)}$, the third and fourth equalities follow by the rule of multiplication when the intersection of two events is considered, the first inequality follows by the observation that $\text{Prob}_{state \leftarrow D^{(j)}}[\ state \in \mathsf{bad} | E_1, \ldots, E_{j-1}\ ]$ is at most 1, and the second inequality follows by definition of bad and the definition of p_j.

Thus, if we consider the maximum over $u = 1, \ldots, n$, we get that for any j:

$$\begin{aligned} & \text{Prob}[\ max_u(\rho_{state_u^{(j)}}) < p_j/2\ ] \\ = \ & \text{Prob}[\ (\rho_{state_1^{(j)}} < p_j/2) \wedge (\rho_{state_2^{(j)}} < p_j/2) \wedge \cdots \wedge (\rho_{state_n^{(j)}} < p_j/2)\ ] \\ = \ & \prod_{u=1}^{n} \text{Prob}[\ \rho_{state_u^{(j)}} < p_j/2\ ] \\ < \ & 1/2^n. \end{aligned}$$

□

Lemma 4 *There exists a fixed constant c_m (independent of $P^\star$, y) such that if m_j is computed as described above, then for all j, $E[m_j^\ell]$ will be at most n^{c_m}/p_j^ℓ.*

PROOF.

We prove the claim in two steps: the first step shows that with overwhelming probability, for all j, E_{new} will compute values for m_j such that $m_j \leq \mathsf{min}_u(2n/\rho_{state_u^{(j)}})$. Then we can use Lemma 3 to argue that when the states are chosen according to the process in step 1a, this value is not too large.

We analyze m_j as follows: by Chernoff bounds (see Lemma 2) and our above notation, it follows that for any state $state^{(j)}$, the probability that E_{new} restarts the UAQK from state $state^{(j)}$ a total of $2n/\rho_{state^{(j)}}$ times without obtaining n successes is less than $(\frac{e}{2})^{-n}$. We consider a total of n states ($state_u^{(j)}$ for $u = 1 \ldots n$), so by the union bound the probability that for any of these states $state^{(j)}$ we could run $P^\star$ from $state^{(j)}$ a total of $2n/\rho_{state^{(j)}}$ times without obtaining n successes is at most $(\frac{e}{2})^{-n}n$.

Recall that E_{new} repeatedly runs the UAQK starting from state $state_1^{(j)}, \ldots, state_n^{(j)}$ and stops as soon as one state $state_u^{(j)}$ results in n successful proofs. Thus, by our analysis, with probability at least $1 - (\frac{e}{2})^{-n}n$, the final m_j value will be at most $\mathsf{min}_u(2n/\rho_{state_u^{(j)}})$.

So we only need to consider $\mathsf{min}_u(2n/\rho_{state_u^{(j)}})$ and show that with high probability, this value is not too large. Recall that p_j is the probability that when E_{new} runs V_{new} with $P^\star$, step $4.j$ outputs ACCEPT, conditioned on the event that steps $4.1, \ldots 4.(j-1)$ output ACCEPT. We showed in Lemma 3 that given that we use the process described in step 1a to sample states $state_u^{(j)}$, we are guaranteed that $\text{Prob}[\ max_u(\rho_{state_u^{(j)}}) < p_j/2\ ]$ is at most 2^{-n}. This means that with probability at least $1 - 2^{-n}$, we will get $\mathsf{min}_u(2n/\rho_{state_u^{(j)}}) \leq 4n/p_j$.

Now we combine this with the above and observe that this implies that with probability at least $1 - 2^{-n} - (\frac{e}{2})^{-n}n > 1 - 2(\frac{e}{2})^{-n}n$, it holds that $m_j \leq \mathsf{min}_u(2n/\rho_{state_u^{(j)}}) \leq 4n/p_j$. Furthermore, when this does not hold, the value of m_j will be at most $\lceil (\frac{e}{2})^{n/\ell}/n \rceil$. ($E_{new}$ will run the UAQK at most $(\frac{e}{2})^{n/\ell}$ times, but m_j only counts the number of times we restart from state $state_1^{(j)}$. E_{new} repeatedly loops over $state_1^{(j)}, \ldots, state_n^{(j)}$, so it starts

from state $state_1^{(j)}$ at most $\lceil (\frac{e}{2})^{n/\ell}/n \rceil$ times.) Thus, we conclude that $E[m_j^\ell] \leq (4n/p_j)^\ell + 2(\frac{e}{2})^{-n} n * (\lceil (\frac{e}{2})^{n/\ell}/n \rceil)^\ell < (4n/p_j)^\ell + 4 < 2^3 (n/p_j)^\ell$, and we can easily set $c_m = 3 + \ell$ so that $E[m_j^\ell] \leq \frac{n^{c_m}}{p_j^\ell}$ for all $n \geq 2$. □

Now we are ready to analyze the running time E_{new}. We will consider the running time of each step of the extraction procedure, then combine the results to prove the proposition.

Claim 1 *The running time of E_{new} in step 1a is independent of i, and there exists a constant c_{1a} such that the expected running time of E_{new} in step 1a is at most $\frac{n^{c_{1a}}}{p(n)}$ for all $n \geq 2$.*

Proof. Note that the value i is not used in step 1a, so it cannot affect the running time. Thus, we focus on the second part of the claim.

For each value of j, E_{new} does the following: E_{new} runs V_{new} with $P^\star$ and checks to see whether steps 4.1 - 4.j output ACCEPT. Then E_{new} rewinds $P^\star$ to its initial state and then repeats until it gets n runs in which this occurs. Recall that, by assumption, when V_{new} interacts with $P^\star$, with probability p we are guaranteed that all of the steps 4.1 - 4.ℓ output ACCEPT (because this is exactly the condition where honest verifier algorithm V_{new} outputs 1). Thus, the probability that the first j steps output ACCEPT must be at least p. Then by Chernoff bounds (see Lemma 1), the probability that E_{new} repeats the process $2n/p$ times without obtaining n successes is less than $(\frac{e}{2})^{-n}$. Finally, even if E_{new} does repeat more than $2n/p$ times without n successes, we are guaranteed that it runs the protocol at most $(\frac{e}{2})^n$ times before aborting and outputting $\perp$.

Thus, the expected running time for each value of j is at most $2n/p \cdot T_{new}(n) + (\frac{e}{2})^{-n} \cdot (\frac{e}{2})^n \cdot T_{new}(n) \leq 3n/p \cdot T_{new}(n) < 3n^{c_{new}+1}/p$. We repeat this process for each value of $j \in \{1, \ldots, \ell\}$, so we conclude that the overall time spent in step 1a is at most $3\ell n^{c_{new}+1}/p$. If we let $c_{1a} = c_{new} + \log(3\ell)$, the claim follows. (Recall that ℓ is a fixed constant.) □

Claim 2 *The running time of E_{new} in step 1b is independent of i, and there exists a constant c_{1b} such that the expected running time of E_{new} in step 1b is $\frac{n^{c_{1b}}}{p(n)}$.*

Proof. Again, the value i is not used in this step, so we only need to bound the expected running time of E_{new}.

For each value of j, E_{new} goes as follows. E_{new} runs until one of the recorded states produces n successful proofs. At that point, it records as m_j, the number of times that it restarted with state $state_1^{(j)}$. Thus, for each j, E_{new} runs the UAQK at most nm_j times, and in total the expected running time in this step will be at most $\sum_{j=1}^{\ell} nE[m_j]T_{new}$ which is at most $n^{c_{new}+1} \sum_{j=1}^{\ell} E[m_j]$.

There is one additional detail, in that with some very small probability E_{new} halts before obtaining n successful proofs. We still have to upper bound the running time in this case so that we can argue about expected running time of this step. Note that in this case, E_{new} runs the UAQK at most $(\frac{e}{2})^{n/\ell}$ times, which means it restarts from state $state_1^{(j)}$ at most $\lceil (\frac{e}{2})^{n/\ell}/n \rceil$ times. We stated above that, if E_{new} halts before setting m_j, we will consider m_j to be $\lceil (\frac{e}{2})^{n/\ell}/n \rceil$, so we can guarantee that the running time of this step will always be bounded by $m_j n T_{new}$. Then we can say that $\sum_{j=1}^{\ell} nE[m_j]T_{new}$ is an upper bound on the expected running time in step 1b.

So all that remains is to bound $\sum_{j=1}^{\ell} E[m_j]$. We showed in Lemma 4 that there exists a constant c_m such that $E[m_{\hat{j}}^{\ell}] \leq n^{c_m}/p_j^{\ell}$, which in particular implies that $E[m_{\hat{j}}] \leq n^{c_m}/p_j$. Putting this all together, we conclude that the total expected running time in step 1b will be $n^{c_{new}+1} \sum_{j=1}^{\ell} E[m_j] \leq n^{c_{new}+1+c_m}/p_j \leq n^{c_{new}+1+c_m}/p$ where the last relationship follows by definition of p_j and p. If we let $c_{1b} = c_{new} + 1 + c_m$, then our claim follows.

□

Claim 3 *The running time of E_{new} in step 2 is independent of i, and there exists a constant c_2 such that the expected running time of E_{new} in step 2 is $O(\frac{n^{c_2}}{p(n)})$.*

PROOF. In this step, E_{new} computes $\sigma = \sigma(n)$, $\Theta = 4m_{\hat{j}}q(n)$, and $\gamma = 2(2m_{\hat{j}})^{\ell-1}q(n)$, and runs $E^{P^{\star}_{best}}(\alpha, y, x)$ on $\sigma(n)$ values of x using Θ different random strings for each. In each of these executions, $E^{P^{\star}_{best}}$ runs for exactly γ steps. Thus, the total running running time in this step will be $\Theta\gamma\sigma = 4m_{\hat{j}}q(n) \cdot 2(2m_{\hat{j}})^{\ell-1}q(n) \cdot \sigma(n) = 4(2m_{\hat{j}})^{\ell}q(n)^2\sigma(n)$. Note that again, this quantity is independent of i, so we just have to show that in expectation it will be $\frac{n^{c_2}}{p(n)}$ for some fixed constant c_2.

Recall that E_{new} computes $m_{\hat{j}}$ by taking $\mathsf{min}_j m_j$ and so $E[m_{\hat{j}}^{\ell}] \leq \mathsf{min}_j E[m_j^{\ell}]$. Thus, by Lemma 4 above, we know that there exists constant c_m such that $E[m_{\hat{j}}^{\ell}] \leq \mathsf{min}_j n^{c_m}/p_j^{\ell}$. Recall also that by our definition, $p = \prod_{j=1}^{\ell} p_j$, which implies that $\mathsf{max}_j p_j^{\ell} \geq p$. This implies that $\mathsf{min}_j n^{\ell}/p_j^{\ell} \leq n^{\ell}/p$, and so $E[m_{\hat{j}}^{\ell}] \leq n^{c_m}/p$.

From above, we have that the expected running time for step 2 is $E[4(2m_{\hat{j}})^{\ell}q(n)^2\sigma(n)] = 2^{\ell+2}E[m_{\hat{j}}^{\ell}]q(n)^2\sigma(n)$. Adding the bounds on $E[m_{\hat{j}}^{\ell}]$ gives an upper bound of $2^{\ell+2}\sigma(n)q(n)^2n^{c_m}/p$. Both $\sigma(n)$ and $q(n)$ are guaranteed to be polynomial, so we can assume that there exists c_σ, c_q such that $\sigma(n)$ is at most n^{c_σ} and $q(n)$ is at most n^{c_q}. Thus, if we let $c_2 = c_\sigma + 2c_q + c_m + \ell + 2$, we get that the running time of this step is at most n^{c_2}/p as desired.

□

Combining the steps to prove the proposition: In each of these steps we have seen that the running time is independent of i. Adding the running times for steps 1 and 2, we get that the expected running time of E_{new} will be at most $n^{c_{1a}}/p + n^{c_{1b}}/p + n^{c_2}/p$ for constants c_{1a}, c_{1b}, c_2. If we let $c = c_{1a} + c_{1b} + c_2$, the proposition follows. □

Success probability of the extraction procedure.

Proposition 2 *There exist negligible functions ν_1, ν_2 such that for any polynomial-sized circuit families $\{P_n^{\star}\}_{n\in N}$ and $\{\mathsf{S}_n\}_{n\in N}$, for sufficiently large n, for any $y \in L_U \cap \{0,1\}^n$, if $P_n^{\star}$ succeeds with probability greater than $\nu_1(n)$, then with probability at least $1 - \nu_2(n)$, ${E_{new}}^{P^{\star}}$ produces bits consistent with a valid witness on all positions chosen (potentially adaptively) by S_n.*

PROOF.

For any circuit $P_n^{\star}$, and n-bit strong y, let $f_{\mathsf{accept}}(P_n^{\star}, y)$ be the probability that $P_n^{\star}$ causes the honest verifier V to accept on input y. For any pair of circuits $P_n^{\star}, \mathsf{S}_n$, and n-bit string y, let $f_{\mathsf{valid}}(P_n^{\star}, \mathsf{S}_n, y)$ be the probability that ${E_{new}}^{P^{\star}}$ produces bits consistent with a valid witness on all positions chosen (potentially adaptively) by S_n.

We will Proposition 2 by proving that the extractor satisfies the following property:

Definition 13 *For any polynomial-size circuit families $\{P_n^\star\}_{n\in N}$ and $\{\mathsf{S}_n\}_{n\in N}$, there exists negligible function ν such that for sufficiently large n, for any $y \in L_U \cap \{0,1\}^n$, $\mathsf{min}(f_{\mathsf{accept}}(P_n^\star, y), 1 - f_{\mathsf{valid}}(P_n^\star, \mathsf{S}_n, y)) \leq \nu(n)$.*

Claim 4 *The property described in Definition 13 holds with respect to a given extraction algorithm iff the first property in Definition 1 holds with respect to that extraction algorithm.*

Proof. Recall that the original definition of UAQK (Definition 1) required that there exist negligible functions ν_1, ν_2 such that for any poly-sized circuit families $\{P_n^\star\}_{n\in N}$ and $\{\mathsf{S}_n\}_{n\in N}$, for sufficiently large n, for any $y \in L_U \cap \{0,1\}^n$, if $f_{\mathsf{accept}}(P_n^\star, y) > \nu_1(n)$ then $f_{\mathsf{valid}}(P_n^\star, \mathsf{S}_n, y) \geq 1 - \nu_2(n)$. First, note that we can equivalently combine ν_1, ν_2 into a single negligible function ν. (If ν_1, ν_2 are both negligible, then $\mathsf{max}(\nu_1, \nu_2)$ is also a negligible function.) Next, note that the statement "if $f_{\mathsf{accept}}(P_n^\star, y) > \nu(n)$ then $f_{\mathsf{valid}}(P_n^\star, \mathsf{S}_n, y) \geq 1 - \nu(n)$" is equivalent to saying "$\mathsf{min}(f_{\mathsf{accept}}(P_n^\star, y), 1 - f_{\mathsf{valid}}(P_n^\star, \mathsf{S}_n, y)) \leq \nu(n)$." Finally, Definition 1 requires that the negligible functions work for all circuit families $\{P_n^\star\}, \{\mathsf{S}_n\}$, while Definition 13 allows a different function for each pair of circuit families. However, we can show these two notions are equivalent using the techniques from [Bel02]. □

Now we need only show that the extractor we described satisfies Definition 13.

We begin by defining some shorthand. Let ρ be the probability that the honest verifier accepts when running a UA $\langle P_{UA}(\cdot,\cdot), V_{UA}(\cdot)\rangle$ with $P_{best}^\star$, i.e., with $P^\star$ starting in state $beststate_{\hat{j}}$. Let $R_{good}(P_{best}^\star, \alpha, y)$ be the set of all random tapes r such that $E^{P_{best}^\star}(\alpha, y, \cdot)$ with coins r produces bits consistent with some valid witness. Let $R_{fast}(P_{best}^\star, \alpha, y, \gamma)$ be the set of all random tapes r such that $E^{P_{best}^\star}(\alpha, y, \cdot)$[6] with coins r runs in time less than γ. (We omit the parameters when they are clear from context and just refer to these sets as R_{good} and R_{fast}.)

For any $P_n^\star, y$, let $\mathsf{fail}_R(P_n^\star, y)$ be the event that when $P_{best}^\star$, α, and $r_1, \ldots, r_\Theta$ are computed according to $E_{new}^{P_n^\star}(y, \cdot)$, for all $\theta \in \{1, \ldots, \Theta\}$ it is the case that $r_\theta \notin R_{good} \cap R_{fast}$. Let $f_R(P_n^\star, y)$ be the probability that $\mathsf{fail}_R(P_n^\star, y)$ occurs (over the choice of E_{new}'s random tape).

Next, we prove a useful claim about R_{good} and R_{fast}.

Claim 5 *There exists a negligible function ν such that for all $\{P_n^\star\}$ and all $y \in \{0,1\}^n$, if we condition on the event that E_{new} computes $m_{\hat{j}}$ such that $m_{\hat{j}} \geq \frac{n}{2\rho}$, then the probability of $\mathsf{fail}_R(P_n^\star, y)$ is at most $\nu(n)$.*

Proof.

Recall that E_{new} computes $\alpha = \frac{n}{2m_{\hat{j}}}$. Thus, the conditioning event means that we consider the case where $\alpha \leq \rho$. Then, by the properties of the universal argument of Theorem 1 for each random choice of r_θ, with probability $\alpha/q(n)$ we are guaranteed that $E^{P_{best}^\star}(\alpha, y, x)$ (using r_θ as coins) produces bits w'_x of a valid witness w', i.e. $r_\theta \in R_{good}$.

We have to guarantee that for one of our r_θ, $E^{P_{best}^\star}(\alpha, y, x)$ both produces correct results and finishes within γ steps. Recall that E_{new} computes γ as $2 * (2m_{\hat{j}})^{\ell-1} q(n) =$

[6]Recall that by the properties considered in Theorem 1, the running time of the oracle machine $E^{\mathcal{O}}(\alpha, \cdot, \cdot)$ depends only on α

$2(n/\alpha)^{d+1}q(n)$. By the properties considered in considered in Theorem 1, we know that for all $y, P^\star$, the expected running time of $E^{P^\star_{best}}(\alpha, y, \cdot)$ is $(n/\alpha)^d$. Thus, by Markov's inequality, the probability (over choice of r_θ), that $E^{P^\star_{best}}(y, \cdot)$ will run in time more than $\gamma = 2(n/\alpha)^{d+1}q(n)$ is at most $(n/\alpha)^d/(2(n/\alpha)^{d+1}q(n)) = \alpha/(2nq(n))$. Or equivalently, we can say that the probability that $r_\theta \notin R_{fast}$ is at most $\alpha/(2nq(n))$.

Combining these we find that the probability that a random r_θ is in $R_{good} \cap R_{fast}$, i.e. that $E^{P^\star_{best}}(\alpha, y, \cdot)$ runs in time less than γ and produces bits of a valid witness, is at least $\rho' = \alpha/q(n) - \alpha/(2q(n)) = \alpha/(2q(n))$. We conclude that since we use $\Theta = 4m_{\hat{j}}q(n) = 2nq(n)/\alpha$ random choices of r_θ, then by Chernoff bounds, with all but negligible probability it is guaranteed that for at least one $\theta \in 1 \ldots \Theta$, we will have $r_\theta \in R_{good} \cap R_{fast}$, i.e. at least one of the resulting extractors $E^{P^\star_{best}}(\alpha, y, \cdot)$ with E using r_θ as coins, will always produce bits consistent with some valid witness and will finish in less than γ steps on all the values of x that we use. □

Claim 6 *There exists a negligible function ν such that for all $\{P^\star_n\}$ and all $y \in \{0,1\}^n$, with probability at least $1 - \nu(n)$, E_{new} computes $m_{\hat{j}} \geq \frac{n}{2\rho}$.*

Proof.

Recall that $m_{\hat{j}} = \mathsf{min}_j m_j$, where m_j is the number of times we had to restart $P^\star$, V_{new} from state $state_1^{(j)}$ before some state $state_u^{(j)} \in \{state_1^{(j)}, \ldots, state_n^{(j)}\}$ produced n successful runs (runs in which step 4.j output ACCEPT).

We begin by considering each value of j separately. For a fixed j, let us again use $\rho^{(j)}_{state}$ to denote the probability that step 4.j outputs ACCEPT when $P^\star$ and V_{new} start in state $state^{(j)}$ corresponding to the end of step 4.$(j-1)$. By Chernoff bounds (see Lemma 2), it is guaranteed that for any state, the probability that the number of repetitions is less than $\frac{n}{2\rho_{state}}$ and yet we obtain n successful runs is at most $2(\frac{4}{e})^{-n/2}$, which is clearly negligible. We run this experiment for n states $state_1^{(j)}, \ldots, state_n^{(j)}$ so by the union bound, the probability that for any of these states we obtain n successful runs in less than $\frac{n}{2\rho_{state}}$ repetitions is also negligible. This means that the probability that $m_j < \frac{n}{2\rho_{beststate_j}}$ (for any fixed j) is negligible.

Now, we run this process for each $j \in 1 \ldots \ell$. Since ℓ is a constant, again by the union bound, the probability that E_{new} finds any $m_j, beststate_j$ such that $m_j \leq \frac{n}{2\rho_{beststate_j}}$ is negligible. Thus it must hold that, for the chosen value $\hat{j}$ and state $beststate_{\hat{j}}$, with all but negligible probability it will be the case that $m_{\hat{j}} \geq \frac{n}{2\rho_{beststate_{\hat{j}}}} = \frac{n}{2\rho}$ as desired. □

Combining the two, we conclude that $f_R(P^\star_n, y)$ is negligible for all $\{P^\star_n\}$ and $\{y_n\}$, i.e. with all but negligible probability, it will be the case that E_{new} gets a random tape such that for at least one value of θ, $r_\theta \in R_{good} \cap R_{fast}$.

Now, if there exists $r_\theta \in R_{good} \cap R_{fast}$, that means that there exists at least one tape for which $E_{r_\theta}(\alpha, y, x)$ produces correct answers for all values of x. We argue that as long as this happens, we will be guaranteed that with overwhelming probability, E_{new} will produce correct answers on any sampleable set of inputs i. We show this based on the collision resistance of the hash function and the binding property of the commitment scheme. The intuition is that if this were not the case, then we could use E_{new} and S to build a polynomial time adversary for either the commitment scheme or the hash function.

Recall that, for each r_θ, E_{new} runs E_{r_θ} to extract all the bits of $s, \mathsf{dec}_s, root, \mathsf{dec}_{root}, w_i, auth_i$. If it finds one r_θ for which all the values extracted are consistent with one another and with C_{root} and C_s, then there are only two possibilities. If at some point it finds another r_θ which produces another consistent set of values $s', \mathsf{dec}'_s, root', \mathsf{dec}'_{root}, w'_i, auth'_i$ for which $w'_i \neq w_i$, then it outputs $\emptyset$. Otherwise it will output the value w_i. Note that, if $\mathsf{fail}_R(P^\star_n, y_n)$ does not occur, then there is one r_θ which will always produce consistent values, and these values will correspond to a valid witness w. Thus, the only case where E_{new} will not output bits of this witness is when there exists another r'_θ such that $E_{r'_\theta}$ also produces values which are consistent for some $i \in I$, but where $w_i \neq w'_i$. We argue that if S can find a set of bits I for which this happens with probability that is not negligible, then we can build a polynomial time adversary for either the hash function or the commitment scheme.

Let $\mathsf{fail}_{\mathsf{hash}}$ be the event that S produces a set I such that for some $i \in I$, there is some r_θ, r'_θ on E_{new}'s random tape such that E_{r_θ} produces values which are consistent but where $root = root'$ and $w_i \neq w'_i$. Let $f_{\mathsf{hash}}(P^\star_n, \mathsf{S}_n, y)$ be the probability that $\mathsf{fail}_{\mathsf{hash}}$ occurs (over the choice of E_{new}'s random tape).

Let $\mathsf{fail}_{\mathsf{comm}}$ be the event that S produces a set I such that for some $i \in I$, there is some r_θ, r'_θ on E_{new}'s random tape such that E_{r_θ} produces values which are consistent but where $root \neq root'$. Let $f_{\mathsf{comm}}(P^\star_n, \mathsf{S}_n, y)$ be the probability that $\mathsf{fail}_{\mathsf{comm}}$ occurs (over the choice of E_{new}'s random tape).

Finally, recall that $f_{\mathsf{accept}}(P^\star_n, y)$ is the probability that $P^\star$ causes the honest verifier V to accept on input y.

Claim 7 *Let $f(P^\star_n, \mathsf{S}_n, y) = \mathsf{min}(f_{\mathsf{hash}}(P^\star_n, \mathsf{S}_n, y), f_{\mathsf{accept}}(P^\star_n, y))$, and for each n, let $f^\star(P^\star_n, \mathsf{S}_n)$ be the maximum value of $f(P^\star_n, \mathsf{S}_n, y)$ over all n-bit $y \in L_U$. If $\mathcal{H}_n$ is a family of collision resistant hash functions secure against non-uniform adversaries, then $f^\star(P^\star_n, \mathsf{S}_n)$ is negligible.*

Suppose there exist polynomial sized circuit families $\{P^\star_n\}$ and $\{\mathsf{S}_n\}$ such that $f^\star(P^\star_n, \mathsf{S}_n)$ is not negligible. Then will we construct a polynomial sized circuit family $\{\mathcal{B}_n\}$ which attempts to break the collision resistance property of the hash function.

In what follows, we will construct a non-uniform adversary $\mathcal{B}$ which runs in expected polynomial time, and breaks the collision resistance property with some probability that is not negligible. However, it is straightforward to convert such an adversary into a family of circuits which has strict polynomial size and still succeeds with probability that is not negligible.

For each value of n, let y_n be an element of L_U such that $f(P^\star_n, \mathsf{S}_n, y)$ is maximized, and let $p_n = f_{\mathsf{accept}}(P^\star_n, y_n)$. Assume $\mathcal{B}$ is given $p_n, y_n, P^\star_n, \mathsf{S}_n$ as part of it's non-uniform advice string.

$\mathcal{B}$ will receive a hash function h^* from it's challenger. With probability p_n, it will abort immediately. (This will help guarantee expected polynomial time.)

Note that S gets to choose it's set I adaptively, based on E_{new}'s responses. Thus, if we want to use S to construct an adversary $\mathcal{B}$ for the collision resistance property, we need to ensure that $\mathcal{B}$ can provide responses which from the view of S are indistinguishable from what E_{new} would give. That means we need to somehow sample a state *beststate* from the same distribution that E_{new} uses. To do this we will essentially run the algorithm of E_{new}.

The only issue is that we need to embed the challenge function h^* into the first message of one of the instances of $\langle P^\star, V\rangle$ in step 1a. Then, we can argue that with at least non-negligible probability, this h^* will be the one used in *beststate*, and so when the event $\mathsf{fail}_{\mathsf{hash}}$ occurs, that will imply a collision in our hash function h^*.

This is a bit tricky because we don't know how many times $\langle P^\star, V\rangle$ will be restarted in step 1a. To deal with this we will first choose a positive integer λ from a distribution where each value λ' is chosen with probability $1/2^{\lambda'}$. (Essentially, this corresponds to a guess that there will be 2^λ restarts in step 1a.) Then we will pick in integer β uniformly at random from $\{1, \ldots, 2^\lambda\}$. (This corresponds to guessing which restart will lead to the chosen *beststate*.)

So, our final algorithm $\mathcal{B}$ will first choose λ and β as described above. Then it will run E_{new}, but in the βth restart in step 1a, it will use h^* instead of choosing a new h at random. (Note that this produces an identical distribution.) It will run S to adaptive choose inputs for this E_{new}. If at any point, in step 2 of E_{new}, there are a pair of tapes r_θ and $r_{\theta'}$ such that E produces sets of values $(s, \mathsf{dec}_s, root, \mathsf{dec}_{root}, w_i, auth_i)$ and $(s', \mathsf{dec}'_s, root', \mathsf{dec}'_{root}, w'_i, auth'_i)$ such that $root = root'$ and w_i, $auth_i$ and w'_i, $auth'_i$ both verify with respect to $root$ but $w_i \neq w'_i$, it will find and return the first pair of different values in the authentication chains $auth_i$ and $auth'_i$.

Now we have to argue that $\mathcal{B}$ runs in expected polynomial time, and that $\mathcal{B}$ succeeds in finding a collision with probability that is not negligible. First we consider the running time. Let $T_E(n)$ be the expected running time of $E_{new}^{P_n^\star}(y_n)$, and let $T_{\mathsf{S}}(n)$ be the (polynomial) size of S_n. Then, the expected running time of $\mathcal{B}$ will be at most $p(n)T_E(n)T_{\mathsf{S}}(n)$. (Since the probability that $\mathcal{B}$ aborts immediately is independent of the running time of $E_{new}^{P_n^\star}$.) Recall that by Proposition 1, $T_E(n) \leq \frac{n^c}{p(n)}$, so we can conclude that the expected running time of $\mathcal{B}$ will be $n^c T_{\mathsf{S}}(n)$ which is a polynomial.

To analyze $\mathcal{B}$'s success probability we note that, if it is in fact the case that the βth restart results in state *beststate*, then $\mathcal{B}$ will succeed with exactly the same probability that the event $\mathsf{fail}_{\mathsf{hash}}$ occurs in the real game. Thus, we have only to argue that with non-negligible probability, $\mathcal{B}$ guesses the correct value for β. First, let N be the number of restarts that occur in step 1a in a given run of the protocol. Now, the probability that $\mathcal{B}$ chooses λ such that $N \leq \lambda \leq 4N$ is at least $1 - \frac{1}{2N} - (1 - \frac{1}{N}) = \frac{1}{2N}$. (To see this, note that for any λ', the probability that $\lambda \geq \lambda'$ is between $1/2^{\lambda-1}$ and $1/2^{\lambda+1}$.) If $\mathcal{B}$ does choose λ in this range, then the probability that $\mathcal{B}$ chooses the right restart β, is at least $\frac{1}{4N}$. Thus, $\mathcal{B}$'s success probability is at least $\frac{1}{8N^2} f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n)$.

Finally, recall that we showed in Proposition 1 that E_{new} runs in expected time $T_E(n) \leq n^c / f_{\mathsf{accept}}(P_n^\star, y_n)$. Furthermore, by Markov's inequality, on any given run the probability that E_{new} runs in time greater than $2T_E(n)$ is at most $1/2$. So with probability at least $1/2$, N will be at most $2T_E(n)$ (since N must be less than the running time E_{new}). Thus, $\mathcal{B}$ will succeed with probability at least $\frac{1}{2}\frac{1}{32T_E(n)^2} f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n) = \frac{1}{64(n^c/f_{\mathsf{accept}}(P_n^\star, y_n))^2} f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n) = \frac{1}{64n^{2c}} f_{\mathsf{accept}}(P_n^\star, y_n)^2 f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n) \geq \frac{1}{64n^{2c}} \mathsf{min}(f_{\mathsf{accept}}(P_n^\star, y_n), f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n))^3$, which is not negligible as long as $f^\star(P_n^\star, \mathsf{S}_n) = \mathsf{min}(f_{\mathsf{accept}}(P_n^\star, y_n), f_{\mathsf{hash}}(P_n^\star, \mathsf{S}_n, y_n))$ is not negligible.

Claim 8 *If $p(n)$ is polynomial, S is polynomial time, and $\langle P_{szk}(\cdot,\cdot), V_{szk}(\cdot)\rangle$ is a statistical zero-knowledge argument of knowledge, and* $(\mathsf{SHTGen}, \mathsf{SHCom}, \mathsf{SHVer}, \mathsf{SHTCom}, \mathsf{SHDec}, \mathsf{SHTDec})$ *is a commitment scheme, then the event $\mathsf{fail}_{\mathsf{comm}}$ can occur with at most negligible probability.*

PROOF. This proof is very similar to the one above for Claim **??**. The only major difference is that in addition to providing the challenge commitment parameters to P^*, $\mathcal{B}$ also has to simulate an argument of knowledge of the corresponding trapdoor. However, because the argument is statistical zero knowledge, the resulting distribution is statistically indistinguishable from the one produced by the real E_{new}, and so the lemma follows by the same arguments as we used above. □

Thus, we conclude that there exist negligible functions $\nu_{\mathsf{hash}}, \nu_{\mathsf{comm}}, \nu_R$ such that for all n, for all y_n, $f_R(P^\star_n, y_n) < \nu_R(n)$, $\mathsf{min}(f_{\mathsf{accept}}(P^\star_n, y_n), f_{\mathsf{hash}}(P^\star_n, \mathsf{S}_n, y_n)) < \nu_{\mathsf{hash}}(n)$ and $\mathsf{min}(f_{\mathsf{accept}}(P^\star_n, y_n), f_{\mathsf{comm}}(P^\star_n, \mathsf{S}_n, y_n)) < \nu_{\mathsf{comm}}(n)$. Combining these, implies that for all y_n, $\mathsf{min}(f_{\mathsf{accept}}(P^\star_n, y_n), f_{\mathsf{hash}}(P^\star_n, \mathsf{S}_n, y_n) + f_{\mathsf{comm}}(P^\star_n, \mathsf{S}_n, y_n) + f_R(P^\star_n, y_n)) < \nu_{\mathsf{hash}}(n) + \nu_{\mathsf{comm}}(n) + \nu_R(n)$. If neither $\mathsf{fail}_{\mathsf{hash}}$ nor $\mathsf{fail}_{\mathsf{comm}}$ nor fail_R occurs, then the extractor succeeds on all the indices produced by S, so if we set $\nu = \nu_{\mathsf{hash}} + \nu_{\mathsf{comm}} + \nu_R$, this implies that $\mathsf{min}(f_{\mathsf{accept}}(P^\star_n, y_n), 1 - f_{\mathsf{valid}}(P^\star_n, \mathsf{S}_n, y_n)) < \nu(n)$ which is negligible as desired.

□

Thus, we have shown both propositions, which completes our proof of Theorem 5. □

C Constructing a Special ZKS Scheme on CRHFs

Recall that we require a zero-knowledge property which is stronger than the standard definition [MRK03] in two ways: (1) it holds even for all the parameters output by the simulator set-up algorithm, even when the adversary selects the randomness used for the set-up, and (2) the simulated commitment algorithm just runs the honest commitment algorithm on an empty database. Here we will show that we can construct such a ZKS scheme based on any family of collision-resistant hash functions.

Trapdoor commitments. We begin by identifying a trapdoor commitment scheme with the analogous properties. Here we consider a variant of the Feige and Shamir scheme [FS90], as described by Fischlin [Fis01]. This construction is based on a non-trapdoor commitment scheme, and a secure one-way function (OWF).

The idea is that we can use a OWF to identify a graph in which, with high probability, it is hard to find a Hamiltonian cycle (by Karp reduction to Directed Hamiltonian Cycle). This graph forms the parameters for the trapdoor commitment scheme (together with the parameters for the non-trapdoor scheme). To commit to 0, the committer applies a random permutation to the graph, and commits (using the non-trapdoor commitment scheme) to the entries in the resulting adjacency matrix. To commit to 1, the committer forms an adjacency matrix for a graph which has only a Hamiltonian cycle, and no other edges, and uses the basic commitment scheme to commit to the entries in this matrix. Finally, to open a commitment to 0, the committer opens all entries, and reveals the permutation, and to open to 1, the committer opens only the entries in the cycle. Note that a commitment that could be opened to both 0 and 1 would imply a Hamiltonian cycle on the original graph. On the other hand, a simulator who knew such a cycle could easily form a commitment to 0 and then open it to 1. Our simulated set-up algorithm will therefore derive a graph for which it can identify a cycle.

Now, we consider our stronger (i.e., statistically secure) properties. We will instantiate the underlying non-trapdoor commitment with a two-round statistically hiding commitment

(which can be built from CRHFs [HM96]), where the first round (receiver to committer) is treated as the set-up algorithm, and the second (committer to receiver) as the commitment itself. Because of the statistical hiding property, we get that for all parameters, the resulting commitment is statistically hiding, even given any auxiliary information. Moreover, this also achieves statistical trapdoorness.

Next note that any graph derived from a OWF as described above will have a cycle. Thus, for any parameters output by the simulation set-up algorithm, and for any relevant auxiliary information, the simulator's trapdoor decommitments will be identical to or indistinguishable from valid decommitments. Note also that the simulated committer can just form a commitment to 0, without knowing the trapdoor, and then later use the trapdoor to open this commitment to 0 or 1. Thus we obtain a trapdoor commitment scheme with properties analogous to those that we require from our ZKS instantiation.

Mercurial Commitments. At a high level, a mercurial commitment scheme is a trapdoor commitment scheme which allows for two different types of commitments (hard and soft commitments), and two different types of opening procedures (hard openings and soft openings). It has been shown [CHL$^+$05] that zero-knowledge sets can be built from any mercurial commitment scheme. Thus, our next step is to find a mercurial commitment scheme with strong zero-knowledge properties. The only mercurial commitment scheme given in [CHL$^+$05] that is built from general assumptions is based on non-interactive zero-knowledge proofs, which require trapdoor permutations. Thus, we instead look at the results of [CDV06], who present a mercurial commitment scheme based on any trapdoor commitment.

The basic construction proceeds as follows: The set-up algorithm generates the parameters for the trapdoor commitment scheme. A hard commitment to bit b is a pair of commitments, the first to b, the second to $1 - b$. A soft commitment is a pair of commitments to 0. A soft opening opens commitment b to 0, while a hard opening opens both commitments.

Note that, using the above trapdoor commitment scheme, (1) a simulator can form an honest soft commitment (a pair of trapdoor commitments to 0), and then use the trapdoor to open the component commitments to 0 or 1 as necessary, and (2) if the underlying trapdoor commitment is hiding for all parameters output by the simulated set-up algorithm, even when the adversary is given the trapdoor, then the same will hold for the resulting mercurial commitment scheme.

Zero-knowledge sets. Finally, we will see that we can build a satisfactory zero-knowledge sets scheme from the above mercurial commitment scheme, and a CRHF using the transformation of [CHL$^+$05] (which is itself a generalization of the construction of [MRK03]). We simply make the following observations about this transformation.

The ZKS set-up algorithm just runs the set-up algorithm of the underlying mercurial commitment scheme and identifies a hash function. The ZKS commitment algorithm forms a series of hard and soft commitments and combines them using a Merkle hash tree. A commitment to an empty database is a single soft commitment. The ZKS proof algorithm generates hard or soft openings for a combination of hard and soft commitments. (For details, see [CHL$^+$05].)

We note the following: (1) Our simulator can form a simulated ZKS commitment by forming a single soft commitment, and this will be equivalent to an honest commitment to an empty database. (2) Our ZKS simulator can generate the appropriate proofs by gener-

ating soft or hard openings for this commitment and subsequent commitments as necessary using the simulator for the mercurial commitment scheme. Since the underlying mercurial commitments are zero-knowledge for all parameters given the trapdoor, this ZKS simulator will also have those properties.

D Proof of Theorem 3

We now give the proof of Theorem 3.

Corrupt verifier. When the real-world verifier $\mathcal{A}$ is corrupted, our ideal-world simulator (i.e., the corrupt verifier in the ideal-world) upon receiving (`Receipt`) from $\mathcal{F}_{DbCom}$ will behave as follows:

Commitment Phase. Receive $(\texttt{ZKSPAR}, \texttt{crs})$ from $\mathcal{A}$ (step 1). Execute the code of $\mathcal{V}$ on input $(\texttt{crs}, \texttt{ZKSPAR})$ (step 2). If $\mathcal{V}$ rejects then abort. Pick $r \in \{0,1\}^k$, set $(c, \mathsf{dec}) = \mathsf{Com}(\texttt{crs}, zks = \mathsf{ZKSCom}(\texttt{ZKSPAR}, \{\}, r))$ and send c to $\mathcal{A}$ (step 3). Executes $\texttt{FormWitness}(\texttt{ZKSPAR}, \{\}, r)$ to generate witness w, and then executes the code of UAP on input $c||\texttt{ZKSPAR}||\texttt{crs}$ with witness $zks||\mathsf{dec}||w$ (step 4). Send (dec, zks) to $\mathcal{A}$ (step 5).

After the commitment phase has ended, run the extractor of the proof given by $\mathcal{A}$ in step 2, thus obtaining $\texttt{aux}, \texttt{ZKSTRAP}$.

Query/Answer Phase. For $i = 1, \ldots, m$ do: Receive x'_i from $\mathcal{A}$. Send x'_i to $\mathcal{F}_{DbCom}$ (step 6). Receive y'_i from $\mathcal{F}_{DbCom}$. Send $(y'_i, \pi = \mathsf{ZKSSimProve}(\texttt{ZKSTRAP}, x'_i, y'_i, r))$ to $\mathcal{A}$ (step 7).

Finally outputs what $\mathcal{A}$ outputs.

We will show that $EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)$ and $\{IDEAL^{\mathcal{F}_{DbCom}}_{H_P,\mathsf{Sim}}(k,x,z)\}$ are computationally indistinguishable based on the zero-knowledge property of the ZKS scheme, the WI property of the UAQK, and the trapdoorness property of the commitment scheme. Consider the following sequence of games.

Game Real: run the honest prover algorithm. The output is $EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)$.

Game Hybrid a: (*Trapdoor commitment*) We proceed as in **Game Real** with the following exception. We run step 1 and step 2 according to the honest prover protocol. Then we rewind to extract $(\texttt{aux}, \texttt{ZKSTRAP})$. In step 3, we generate $(c, \texttt{aux}_c) \leftarrow \mathsf{TCom}(\texttt{crs}, \texttt{aux})$ and send it to the adversary. Next, choose random r_1, and generate $zks_1 = \mathsf{ZKSCom}(\texttt{ZKSPAR}, Db, r_1)$ and $\mathsf{dec}_1 \leftarrow \mathsf{TDec}(\texttt{aux}_c, zks_1)$. Then we proceed with step 4-7 as in the honest protocol, using $\mathsf{dec} = \mathsf{dec}_1$ and $zks = zks_1$.

Let $\{Hyb^a_{\mathcal{A}}(k,x,z)\}$ be the output of the above experiment. By the trapdoorness property the two experiments are indistinguishable, i.e. $\{Hyb^a_{\mathcal{A}}(k,x,z)\} \approx EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)$.

Game Hybrid b: (*Simulated WI witness*) We proceed as in **Game Hybrid** a with the following exception. In step 4, we choose random r_2, and generate $zks_2 = \mathsf{ZKSCom}(\texttt{ZKSPAR}, \{\}, r_2)$ and $\mathsf{dec}_2 \leftarrow \mathsf{TDec}(\texttt{aux}_c, zks_2)$. Then we use a witness formed using zks_2, dec_2 and the empty database to run the UAQK. The rest of the game (steps 5-7) proceeds as in **Game Hybrid** a, using $\mathsf{dec} = \mathsf{dec}_1$ and $zks = zks_1$, and the real Db. Let $\{Hyb^b_{\mathcal{A}}(k,x,z)\}$ be the output of the above experiment.

Claim 9 *$\{Hyb^b_{\mathcal{A}}(k,x,z)\}$ and $\{Hyb^a_{\mathcal{A}}(k,x,z)\}$ are indistinguishable by the WI property of the UAQK.*

PROOF. If there exists a distinguisher between $\{Hyb^a_{\mathcal{A}}(k,x,z)\}$ and $\{Hyb^b_{\mathcal{A}}(k,x,z)\}$, then we can easily break the witness indistinguishability of the universal argument. We run step 1 and step 2, and then extract aux from the proof of knowledge. Then we use this to form commitment c and decommitments dec_1 to $zks_1 = \mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r_1)$ and dec_2 to $zks_2 = \mathsf{ZKSCom}(\mathtt{ZKSPAR}, \{\}, r_2)$. In step 3 we send c to V. Then we run $\mathtt{FormWitness}$ twice, with $(\mathtt{ZKSPAR}, Db, r_1)$ and with $(\mathtt{ZKSPAR}, \{\}, r_2)$ to obtain w_1, w_2. We give $zks_1||\mathsf{dec}_1||w_1$ and $zks_2||\mathsf{dec}_2||w_2$ to the WI challenger of the UAQK. Now, in step 4 it is sufficient to forward the messages to the external UAP that will use one of the two given witnesses. We complete the game using zks_1, Db, r_1 as in the honest algorithm. In one case this will be equivalent to $\{Hyb^a_{\mathcal{A}}(k,x,z)\}$ and in the other case it would be $\{Hyb^b_{\mathcal{A}}(k,x,z)\}$, thus the distinguisher will allow us to break the witness indistinguishability of UAP. □

Game Hybrid c:(*Simulated ZKS commitment and proofs*) We proceed as in **Game Hybrid** b with the following exception. In step 5, we choose a new random string r_3, and generate $zks_3 = \mathsf{ZKSCom}(\mathtt{ZKSPAR}, \{\}, r_3)$ and $\mathsf{dec}_3 = \mathsf{TDec}(\mathsf{aux}_c, zks_3)$, then we set $zks = zks_3$ and $\mathsf{dec} = \mathsf{dec}_3$. In step 7, we use $\mathsf{ZKSSimProve}(\mathtt{ZKSTRAP}, \cdot, \cdot, r_3)$ to compute each proof π. Let $Hyb^c_{\mathcal{A}}(k,x,z)$ be the output of the above experiment.

Claim 10 $\{Hyb^c_{\mathcal{A}}(k,x,z)\} \approx \{Hyb^b_{\mathcal{A}}(k,x,z)\}$ *by the special ZK property of the ZKS scheme.*

PROOF. If there exists a distinguisher between $\{Hyb^b_{\mathcal{A}}(k,x,z)\}$ and $\{Hyb^c_{\mathcal{A}}(k,x,z)\}$ we can easily break the special zero-knowledge property of $(\mathsf{ZKSSetup}, \mathsf{ZKSCom}, \mathsf{ZKSProve}, \mathsf{ZKSVerify})$.

Our reduction proceeds as follows. We run step 1-2 of the commitment phase using the honest protocol. Then we rewind and extract $\mathtt{ZKSTRAP}, \mathsf{aux}$. We generate trapdoor commitment c and decommitment dec_2 to $zks = \mathsf{ZKSCom}(\mathtt{ZKSPAR}, \{\}, r_2)$, and use these for the UAQK. We then send the value $\mathtt{ZKSPAR}$ received in step 1 to our challenge oracle with database Db, and get back zks', which we send to the adversary in step 5 along with decommitment dec' (formed using aux_c). Then, in the query phase, for each query we query our challenge oracle, get back π_i, and send $Db[x_i], \pi_i$, to the adversary as step 7.

Note that if the oracle returns $\mathsf{ZKSCom}(\mathtt{ZKSPAR}, Db, r')$, and $\mathsf{ZKSProve}(\mathtt{ZKSPAR}, x_i, Db[x_i], Db, r')$, this will be identical to **Game Hybrid** b (with $r_1 = r'$). If instead the oracle returns $\mathsf{ZKSCom}(\mathtt{ZKSPAR}, \{\}, r')$ and $\mathsf{ZKSSimProve}(\mathtt{ZKSTRAP}, x_i, y_i, r')$, this will be identical to **Game Hybrid** c (with $r_3 = r'$). □

Game Hybrid d: (*Same simulated zks in UAQK as for ZKS proofs*) We proceed as in **Game Hybrid** c with the following exception. In step 4, we use $zks_3, \mathsf{dec}_3, r_3$ (i.e. the same values as in step 5-7) to form the witness for the UAQK. Let $\{Hyb^d_{\mathcal{A}}(k,x,z)\}$ be the output of the above experiment.

Claim 11 $\{Hyb^d_{\mathcal{A}}(k,x,z)\}$ *and* $\{Hyb^c_{\mathcal{A}}(k,x,z)\}$ *are indistinguishable by the WI property of the UAQK.*

The proof is very similar to the proof of Claim 9.

Game Ideal: run the simulator and the ideal functionality. Let $IDEAL_{\mathcal{A}}(k,x,z)$ be the distribution of the output. Note that **Game Ideal** is identical to **Game Hybrid** d. Thus, $\{IDEAL_{\mathcal{A}}(k,x,z)\} = \{Hyb^d_{\mathcal{A}}(k,x,z)\}$.

We conclude that $\{EXEC^{\pi}_{H_P,\mathcal{A}}(k,x,z)\} \approx \{IDEAL^{\mathcal{F}_{DbCom}}_{H_P,\mathsf{Sim}}(k,x,z)\}$, and our construction satisfies Definition 12 in the corrupt verifier case.

Corrupt prover. When the real-world prover $\mathcal{A}$ is corrupted, our ideal-world simulator (i.e., the ideal-world prover) will behave as follows.

Commitment Phase. Execute step 1-5 as described in the honest verifier protocol.

After commitment phase has ended successfully, and extract the circuit $\mathcal{C}_{Db}$ which implicitly determines the database. (This is the challenging step - we describe exactly how this is done below.) Send $\mathcal{C}_{Db}$ to the functionality.

Query/Answer Phase. When receiving a query x_i from the functionality, send it in step 6 to $\mathcal{A}$. When receiving a proof in step 7 for x_i from $\mathcal{A}$, if it is correct send $(\texttt{Open}, 1, x)$ to the functionality, otherwise send $(\texttt{Open}, 0, x)$ to the functionality.

Finally, output what $\mathcal{A}$ outputs.

Extracting $\mathcal{C}_{Db}$. We now describe in more detail how the simulator generates the circuit $\mathcal{C}_{Db}$. Let $\mathcal{C}_{\mathcal{A}}$ be a circuit that has the same behavior as $\mathcal{A}$. (Note that $\mathcal{C}_{\mathcal{A}}$ is guaranteed to have polynomial size.) Let $\mathcal{A}^{\star}$ be a PPT algorithm which behaves as follows: It runs step 1-3 of the honest verifier algorithm interacting with $\mathcal{C}_{\mathcal{A}}$, using the same coins that the simulator used. Then, $\mathcal{A}^{\star}$ runs the prover side of the UAQK in step 4 by forwarding all messages to and from $\mathcal{C}_{\mathcal{A}}$. Construct a corresponding circuit - we refer to the result as $P^{\star}_{\mathcal{A}}$.

By the definition of quasi knowledge, we know that there exists an algorithm E, which when run on random string R and with oracle access to a UAQK prover $P^{\star}_{\mathcal{A}}$ that succeeds with probability p, runs in expected time $\mathsf{poly}(k)/p$ (for some polynomial independent of $P^{\star}_{\mathcal{A}}$) and correctly extracts bits of a valid witness with overwhelming probability as long as p is large enough.

We construct two other algorithms based on $E^{P^{\star}_{\mathcal{A}}}_{R}$. The first, which we will call $Db^{\mathcal{A}^{\star}}$, will proceed as follows. Let y be the statement corresponding to $c||\texttt{ZKSPAR}||\texttt{crs}$. On input x, $Db^{\mathcal{A}^{\star}}$ will run Eval on input x. Whenever Eval tries to access the ith bit of w, the algorithm will run $E^{P^{\star}_{\mathcal{A}}}_{R}(y, i')$ and use the resulting bit as w_i. (Here we must adjust the position i and use the adjusted value i' to take into account the fact that the witness used for the UAQK is prepended with $zks||\mathsf{dec}$.) $Db^{\mathcal{A}^{\star}}$ will then output whatever Eval outputs.

The second algorithm, which we will call $\mathsf{Pf}^{\mathcal{A}^{\star}}$ will proceed similarly, but it will run PfGen rather than Eval. This algorithm is not actually used by the simulator, but will be necessary for our security arguments.

Note that the result is 2 deterministic oracle algorithms. Furthermore, we can compute a fixed upper bound on the running time of these algorithms: if we run $E^{P^{\star}_{\mathcal{A}}}_{R}(y, 0)$, that tells us the running time T_E of the extractor for this randomness[7], so the total running time will be $T_E \cdot |P^{\star}_{A}| \cdot q(k)$ (or $q'(k)$), where q, q' are the polynomials defined in the local witness property of the ZKS scheme. Then we convert these algorithms into the equivalent circuits: Suppose the running time of these algorithms is $O(t(k))$. Then there is a corresponding circuit with size $O(t(k)\log(t(k)))$ with identical behavior. Our simulator will generate the circuit corresponding to $Db^{\mathcal{A}^{\star}}$, which we will refer to as $\mathcal{C}_{Db}$, and send the result to $\mathcal{F}_{DbCom}$. (If this process takes longer than 2^k time, we abort.)

[7]Recall that the running time is the same for all i.

Claim 12 *The simulator described above runs in expected polynomial time.*

PROOF. Recall that the simulator begins by running $\mathcal{A}$ with the honest verifier, and only continues with extracting C_{Db} if steps 1-5 execute successfully. First, consider just those executions where the steps 1-3 are successful. After a given execution of step 1-3, let p be the probability that $\mathcal{A}$ will cause the UAV to accept. At this point, we know that probability $1-p$, the simulator will simply abort. If the simulator does not abort, then it will construct a circuit using the oracle algorithm $E_R^{\mathcal{A}^\star}(y,\cdot)$, which by the UAQK property will have running time k^c/p for some fixed constant c (which is independent of $\mathcal{A}^\star, y, R$).[8] As discussed earlier, we combine this with $\mathcal{A}^\star$ and with Eval, and the resulting algorithm will have fixed running time $k^c/p \cdot T_{\mathcal{A}}(k) \cdot q(k)$ (where q is as defined in the local witness property, and $T_{\mathcal{A}}$ is a polynomial describing the running time of $\mathcal{A}$). Thus, we can construct a corresponding circuit of size $k^{c'}/p\log(1/p)$ for appropriate constant c' (where again, c' is independent of $\mathcal{A}^\star, y, R$), and if this takes longer than 2^k time, we abort. Now, we can say that at the point when the first 3 steps have been successfully completed, and the adversary has probability p of successfully completing the rest of the protocol, the expected running time of the rest of the algorithm is $p \cdot \mathsf{min}(2^k, k^{c'}/p\log(1/p)) = \mathsf{min}(p\cdot 2^k, k^{c'}\log(1/p)) \leq k^{c'} \cdot k = k^{c'+1}$ for fixed constant c'. Note that c' does not depend on the adversarial prover $\mathcal{A}^\star$ or the statement y, which means it will be independent of the randomness used in steps 1-3. Thus, we can conclude that the expected running time of the second half of the algorithm is always $k^{c'+1}$. Finally, as running the first three steps takes only polynomial time, we can conclude that the overall expected running time is at most polynomial. □

Based on this, we can show that the output of the above simulator (i.e., the adversarial prover in the ideal game) is computationally indistinguishable from the output of the corrupt prover (i.e., the adversarial prover of the real game). Consider the following game:

Game Hybrid. Run as in the ideal-world game with the following two exceptions: Compute $\mathcal{A}^\star$ as described above, and in addition to using $E_R^{\mathcal{A}^\star}$ to compute $\mathcal{C}_{Db}$, also use it to extract the values zks, dec corresponding to the beginning of the witness. If the ZKS commitment zks produced by the extractor does not match the value sent in step 5, output "Commitment consistency error". If for any i, the response y_i sent by the prover in step 7 does not match the value produced by $\mathcal{C}_{Db}$, output "ZKS consistency error".

Claim 13 *For any $\mathcal{A}$ such that Game Hybrid outputs an error with negligible probability, then $\{EXEC^{\pi}_{\mathcal{A},H_V}(k,x,z)\} \approx \{IDEAL^{\mathcal{F}_{DbCom}}_{\mathsf{Sim},H_V}(k,x,z)\}$.*

Note that, if the simulator does not abort, and the game does not output an error, then the two games are identical. Thus, we need only show that simulator aborts with the same probability as the honest verifier. Note that, if neither of the above errors occurs, then the only time the simulator aborts when the honest verifier would not is when the process of generating the extractor takes more than 2^k time. However, as argued above, this process should take $O(\mathsf{poly}(k)/p\log(1/p))$ time, where p is the probability that $\mathcal{A}^\star$ will produce a valid proof. If this value is more than polynomial, that means p must be exponentially small. Thus, any run on which the extractor takes more than 2^k time is one where with

[8] Our statement y has length polynomial in k, so in fact the running time will be $\mathsf{poly}(k)^c/p$, but again, this polynomial is fixed and independent of $\mathcal{A}, y, R$, so we can just incorporate the extra exponent into c.

overwhelming probability step 4 of the commitment protocol will fail, and the honest verifier would also have aborted. Thus, the probability of aborting differs by at most a negligible amount, and we conclude that the two games are indistinguishable (as long as there are no errors).

Claim 14 *For all $\mathcal{A}$, the probability that Game Hybrid outputs "Commitment consistency error" is negligible by the binding property of the commitment scheme.*

PROOF. Suppose there exists an efficient $\mathcal{A}$ such that with non-negligible probability $\mathcal{A}$ cause Game Hybrid to output "Commitment consistency error", i.e., the *zks* extracted from the UAQK is different from the *zks* that the adversary sends in step 5. Then we can easily break the binding property of the commitment scheme. We are given challenge parameters `crs` for the commitment scheme. In step 1 we generate $\texttt{ZKSPAR} \leftarrow \mathsf{ZKSSimSetup}(1^k)$, and send ZKSPAR, `crs`. In step 2 we use the ZK simulator to simulate the proof. Then we run step 3-5 honestly, and receive c, zks, dec. Finally, we define $\mathcal{A}^\star$ as above, choose random R, and run $E_R^{\mathcal{A}^\star}(y, \cdot)$ on the first set of indices to extract zks', dec'. We output $c, zks, \mathsf{dec}, zks', \mathsf{dec}'$. Note that the resulting algorithm will run in expected polynomial time, by a similar argument to claim 12. By assumption $zks \neq zks'$ with non-negligible probability so this breaks the binding property of the commitment scheme.[9] □

Claim 15 *Let zks' be the value that the simulator extracts from the UAQK, and let $Db^{\mathcal{A}^\star}$, $\mathsf{Pf}^{\mathcal{A}^\star}$ be the algorithms produced by the above process. For any polynomial sized set of inputs $x_1, \ldots, x_m$, for any adversary $\mathcal{A}$, the probability that the simulator does not abort, and that the resulting $Db^{\mathcal{A}^\star}$, $\mathsf{Pf}^{\mathcal{A}^\star}$ will be such that for some $i \in 1, \ldots, m$, $\mathsf{ZKSVerify}(\texttt{ZKSPAR}, zks, x_i, Db^{\mathcal{A}^\star}(x_i), \mathsf{Pf}^{\mathcal{A}^\star}(x_i)) \neq 1$ is negligible.*

PROOF. Note that by the local witness property, as long as $E_R^{\mathcal{A}^\star}$ returns the correct values (i.e., bits consistent with some w such that $\mathsf{TMVer}(w) = 1$) for all positions queried by Eval and PfGen, it will be true that for all $i \in 1, \ldots, m$, $\mathsf{ZKSVerify}(\texttt{ZKSPAR}, zks, x_i, Db^{\mathcal{A}^\star}(x_i), \mathsf{Pf}^{\mathcal{A}^\star}(x_i)) = 1$.

Note that this process of querying bits of the witness based on Eval and PfGen together can be seen as a polynomial time sampling algorithm S. (Recall that, by the local witness property, they only access a polynomial number of bits in total.)

Let ν_1, ν_2 be the negligible functions defined in the definition of quasi knowledge.

Now, consider the situation after the simulator has run steps 1-3 of the commitment phase with $\mathcal{A}$. At this point, let p be the probability that the resulting $\mathcal{A}^\star$ successfully completes the proof. Then, if $p > \nu_1(k)$, the extractor will work as desired and we will obtain proofs that verify for all x_i with probability at least $1 - \nu_2(k)$. Otherwise, we abort with probability at least $1 - \nu_1(k)$. Thus, the probability that we do not abort, but do not produce $Db^{\mathcal{A}^\star}$, $\mathsf{Pf}^{\mathcal{A}^\star}$ that work for the positions queried by $\mathsf{Eval}, \mathsf{PfGen}$ is at most $\mathsf{max}(\nu_1(k), \nu_2(k))$, which is negligible. □

[9] The argument here gives an adversary which runs in expected polynomial time and breaks the binding probability with non-negligible probability, but it is straightforward to convert this into an adversary which runs in strict polynomial time.

Claim 16 *For all $\mathcal{A}$, the probability that Game Hybrid outputs "ZKS consistency error", but not "Commitment consistency error" is negligible by the soundness property of the ZKS scheme.*

PROOF. Suppose that there exists an efficient $\mathcal{A}$ such that this probability is non-negligible, i.e., with non-negligible probability, the *zks* extracted from the UAQK is the same as the *zks* that the adversary sends in step 5, but at least one of the responses sent by $\mathcal{A}$ in step 7 is not consistent with the extracted circuit $\mathcal{C}_{Db}$. Then we can break the soundness property of the ZKS protocol. We are given challenge parameters `ZKSPAR` for the zero-knowledge sets scheme. In step 1 we send `ZKSPAR` to the adversary along with the parameters `crs` for the commitment scheme. In step 2, we use the ZK simulator. After the commitment phase, we define $\mathcal{A}^\star$, choose a random R, run $E_R^{\mathcal{A}^\star}$ to extract *zks*, and use $E_R^{\mathcal{A}^\star}$ to construct $\mathcal{C}_{Db}, \mathcal{C}_{ZKSpf}$. (If the extracted value *zks* is different from the one received in step 5, we abort.) Then in the query/answer phase, each time the adversary sends a response (y_i, π_i) to query x_i, we compute $y_i' = \mathcal{C}_{Db}(x_i)$. If $\mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x_i, y_i, \pi_i) = 0$, we continue to the next query. Otherwise, if $y_i' \neq y_i$, we also compute $\mathsf{Pf}^{\mathcal{A}^\star}(x_i)$ to produce π_i', and output $(zks, x_i, y_i, \pi_i, y_i', \pi_i')$. Note that the resulting algorithm will run in expected polynomial time, by a similar argument to claim 12. By Claim 15, we are guaranteed that with overwhelming probability $\mathsf{ZKSVerify}(\mathtt{ZKSPAR}, zks, x_i, y_i', \pi_i') = 1$. Thus, with non-negligible probability we succeed in finding valid proofs for x_i and two different values y_i, y_i', which contradicts the soundness property of the ZKS.[10] □

We conclude that $\{EXEC^{\pi}_{\mathcal{A},H_V}(k,x,z)\} \approx \{IDEAL^{\mathcal{F}_{DbCom}}_{\mathsf{Sim},H_V}(k,x,z)\}$, and our construction satisfies Definition 12 in the corrupt prover case as well.

[10] See footnote 9.