Efficient Zero-Knowledge Proofs of Knowledge Without Intractability Assumptions

Ronald Cramer[1], Ivan Damgård[2], and Philip MacKenzie[3]

[1] Institute for Theoretical Computer Science
ETH Zurich, 8092 Zurich
cramer@inf.ethz.ch
[2] Aarhus University, BRICS
[3] Information Sciences Research Center, Bell Laboratories
600 Mountain Ave., Murray Hill, NJ 07974-0636
philmac@research.bell-labs.com

Abstract. We initiate the investigation of the class of relations that admit extremely efficient perfect zero knowledge proofs of knowledge: constant number of rounds, communication linear in the length of the statement and the witness, and negligible knowledge error. In its most general incarnation, our result says that for relations that have a particular three-move honest-verifier zero-knowledge (HVZK) proof of knowledge, and which admit a particular three-move HVZK proof of knowledge for an associated commitment relation, perfect zero knowledge (against a general verifier) can be achieved essentially for free, even when proving statements on several instances combined under under monotone function composition. In addition, perfect zero-knowledge is achieved with an optimal 4-moves. Instantiations of our main protocol lead to efficient perfect ZK proofs of knowledge of discrete logarithms and RSA-roots, or more generally, q-one-way group homomorphisms. None of our results rely on intractability assumptions.

1 Introduction

1.1 The Problem and the Motivation

Suppose a prover P to would like to efficiently convince a verifier V about knowledge of some secret, for instance, a discrete logarithm or an RSA root. More formally, we want to prove knowledge of a secret witness to a public value over some given relation. Of course, a trivial way to do this would be to simply reveal the secret. However, this is useless in a cryptographic context – we want a solution allowing P to keep the secret to himself. More concretely, we want an efficient zero-knowledge proof of knowledge for the given relation.

In this paper we characterize a class of relations where such a zero-knowledge proof of knowledge can be built without making any computational assumptions, and at negligible extra cost (communication and number of moves) compared to the trivial non zero-knowledge solution. Related results in this vein were shown before in [CD98], but for the case of proofs of language membership (concretely, for the boolean-circuit satisfiability problem).

Before describing our results in more detail, we emphasize that the fact that we use no intractability assumptions implies that soundness and zero-knowledge of our protocols hold independently of the hardness of any problem, *including even the problems for which we build our proofs of knowledge.* Even if this is nice from a theoretical point of view, one could argue that in practice, there is no use in making a zero-knowledge proof of knowledge for a problem that does not satisfy an intractability assumption: if anyone can compute the prover's secret, there is not any sense in trying to hide it.

There are two answers to this objection: first it implicitly assumes that computational problems fall in two classes, where you have on one side problems for which an intractability assumption holds, and on the other side easy problems. This is in fact not the case. Cryptographic assumptions require that the problem be hard on average, whereas problems that are hard in the worst case, but often easy, fall in neither of the two classes. For instance, graph isomorphism seems to be one such case. Thus there could well be cases where one would like a zero-knowledge proof for a problem, even though one might be reluctant to base an intractability assumption on it.

Secondly, it should be noted that a protocol with no assumptions is much easier to use as subprotocol in a larger construction (and this is indeed an important type of application of ZK proofs). Let us elaborate on this: In any multiplayer protocol it is an obvious advantage if we can prove it secure against cheating by player A, even if A is computationally unbounded. Indeed many cryptographic tasks do allow for solutions where some (though not all!) players can be unbounded. However, if in constructing such a solution, we use as subprotocol a ZK proof that only works if one of the players, say A, has bounded computing power, then nothing we build on top will allow us to prove the overall protocol secure against an unbounded A. So we see that using subprotocols with no assumptions allows more design freedom in deciding which players should be protected against unconditionally.

1.2 Our Results

Our results apply to relations admitting a proof of knowledge of a special form: a prover P can convince in perfect zero-knowledge an *honest* verifier using a 3-move protocol, where P sends the first message, V sends a random challenge, and P answers the challenge. We assume that, whereas a truthful P can answer any challenge, a cheating prover can answer at most one. We call this a Σ-protocol (a precise definition follows below). There are several known protocols of this form, e.g., [Sch89] for discrete logs and [GQ88] for RSA roots. These protocols have the efficiency we are after: constant round and communication a constant factor larger than the length of the secret. But they cannot be proved to be zero-knowledge against a dishonest verifier - at least not by any known resettable simulation technique [GK96b].

Our main result is as follows: it was observed in [Dam89,FS89] that any Σ-protocol for a relation R leads to existence of a commitment scheme, which in turn naturally defines a new relation R' (see below for details). Loosely speaking, R' consists of pairs in which one component is a commitment and the other is a string opening that commitment. We consider the case where both R and R' have Σ-protocols. This class includes the cases of discrete log, RSA roots, and in general any relation built from a q-one-way function [CD98].[1] For any such relation, we obtain a 4-move protocol that is perfect zero-knowledge in general *without making any computational assumptions*. Note that this in particular means - somewhat surprisingly - that we do not need to assume that the commitment scheme associate to R is secure. We just use it as a building block in the protocol.

Our results come at the price of one extra move and a small constant factor of communication compared to the Σ-protocols we start from. The 4 moves is optimal for protocols that are black-box zero-knowledge [GK96b]. Additionally, we obtain a 6-move protocol with an even smaller constant factor of communication overhead.[2]

When instantiated for concrete problems like discrete log and RSA, we get very practical perfect zero-knowledge protocols which naturally have many applications, including improving the efficiency of many distributed cryptographic protocols (e.g., [GRR98,FMY98]). As an example of the practicality of these protocols, consider the problem of discrete logs in Z_p^* (p prime) where $|p|$ denotes the bit length of p. We present a 4-round ZK proof of knowledge for this relation that communicates only $9|p|$ bits, Additionally it only requires 10 exponentitations (6 by V, 4 by P), so there is only a small constant factor of computation overhead, also. Our 6-move version of this protocol communicates $8|p|$ bits and requires only 7 exponentiations (4 by V, 3 by P). The knowledge error for both proofs is simply $1/q$, where q is the order of the generator used. As another example, we mention one case where our protocols are more efficient than any previous ones, with or without intractability assumptions, namely the case of proving knowledge of an RSA root where the public exponent is a small prime, say 3. If we try to apply Guillou-Quisquater [GQ88] directly to this situation, we can only get negligible error probability if we iterate their protocol many times because the error probability of one instance is $1/3$ in this case. By constrast, a variant of our construction allows to give a perfect ZK proof of knowledge of an RSA third root with the same asymptotic efficiency as for RSA roots with large public exponent. Specifically, to achieve knowledge error 3^{-t} for an RSA modulus n, our protocol runs in 4 rounds and communicates only $11 \log |n| + 5t$ bits, as compared to $\Omega(t)$ rounds and $\Omega(t \log |n|)$ bits for the iterated GQ protocol.

Our methods generalize to provide perfect zero-knowledge proofs of partial knowledge, allowing P to, for example, prove that he knows at least t out of $n > t$

[1] Actually this class includes random self-reducible languages also (i.e., our result encompasses all the languages considered in [BMO90,SKS91]), but we are mainly concerned with those languages with efficient (linear communication) Σ-protocols.

[2] Due to space limitations, the 6-move protocol can only be found in the full version.

secrets, in particular without releasing any information about which secrets are known. This may be seen as a generalization of the work by Cramer, Damgård and Schoenmakers [CDS94] who provide efficient witness hiding protocols for the same type of problems. In this context, our work shows that the stronger property of perfect zero-knowledge can be obtained at the cost of one extra move in the protocol. Monotone compositions of ZK were also studied in [SCP93] and [SCPY94], but their protocols are not as efficient as ours, in fact we exhibit example cases, where our protocols are super polynomially more efficient.

1.3 Related Work

There are several earlier results on building general zero-knowledge proofs from honest-verifier zero-knowledge protocols of this type. A well-known technique that acheives constant round protocols (which was only proven secure quite recently [GK96a]) is to let V commit to his challenge before P sends his first message. Apart from the fact that this would require a computational assumption, the method only seems to work for proofs of language membership. Damgård [Dam93] shows a method without computational assumptions, which however leads to a non-constant number of moves. Subsequent results, such as [GSV98], also lead to a non-constant number of moves, and actually do not apply to proofs of knowledge.

Also, there are methods for building constant-round protoocols from scratch: Bellare, Micali and Ostrovsky [BMO90] show that 5-move perfect zero-knowledge proofs are possible for any random self-reducible relation.[3] Saito, Kurosawa and Sakurai [SKS91] improve this to 4-moves. Feige and Shamir [FS89] show a general method for constructing constant-round ZK proofs of knowledge. Their 4-move protocol relies on the hardness of discrete log, and their 5-move protocol relies on the existence of a one-way function. The protocols work for any NP relation, but do not lead to practical protocols. Bellare, Jakobsson, and Yung [BJY97] construct a four-round (computational) proof of knowledge for any NP-relation that is computationally ZK, and relies only on the existence of a one-way function. Again, this is not a practical protocol. None of these results achieve linear communication.

1.4 Road map to the paper

Section 3 presents most of the notation, concepts, definitions, and model we use. Section 4 presents the 4-move perfect ZK proof-of-knowledge protocol. Section 5 presents security proofs for these protocols. Section 6 present some examples of how the general protocol can be instantiated. Section 7 presents an extension to monotone compostitions.

[3] They prove this for language membership, but it is not hard to show that their protocol is also a proof of knowledge.

2 Definitions of Proofs of Knowledge and Perfect Zero-Knowledge

For a *Proof of Knowledge* (P, V) for some binary relation $R = \{(\alpha, \beta)\}$ we use the definition of Bellare and Goldreich [BG92]. This definition includes the usual *completeness* condition which says that a prover indeed knowing what he claims he knows is accepted by the verifier. For convenience we (informally) state the technical soundness condition.

There is a function $\kappa : \{0,1\}^* \rightarrow [0,1]$ (*knowledge error*) and a probabilistic expected polynomial time oracle machine E (*knowledge extractor*) such that the following holds.

Let P be an arbitrary prover(not necessarily following the protocol!), claiming to know a witness β for a given public string α, and let $\epsilon(\alpha)$ denote P's success probability. Then E, having rewindable black-box access to the prover P, either outputs some witness β for α, or a special halting-symbol. Furthermore, the probability that E outputs a witness is greater than or equal to $\epsilon(\alpha) - \kappa(\alpha)$.

In our context *Perfect Zero-Knowledge* is defined by means of the usual black-box formulation [Gol95], but with perfect indistinguishablity of conversations produced by the expected polynomial time univeral simulator, having rewindable black-box access to the verifier, and the conversations resulting from "real life" interactions between prover and verifier.

3 Definition and Basic Theory of Σ-protocols

We overview the basic definitions and properties of our primitive Σ-protocols. Let a Σ-protocol (A, B) be a three move interactive protocol between a probabilistic polynomial-time prover A and a probabilistic polynomial-time verifier B, where the prover acts first. The verifier is only required to send random bits as a challenge to the prover.

More precisely, let $R = \{(\alpha, \beta)\}$ be a binary relation and assume that for some given polynomial $p(\cdot)$ it holds that $|\beta| \leq p(|\alpha|)$ for all $(\alpha, \beta) \in R$. Furthermore, let R be testable in polynomial time, and let R^* denote the collection of strings α such that, for some string β, $(\alpha, \beta) \in R$. The string β is called a witness for α. For some $(\alpha, \beta) \in R$, the common input to both players is α while β is private input to the prover. For such given α, let (a, c, z) denote the conversation between the prover and the verifier. To compute the first and final messages, the prover invokes efficient algorithms $a(\cdot)$ and $z(\cdot)$, respectively, using (α, β) and random bits as input. Using an efficient predicate $\phi(\cdot)$, the verifier decides whether the conversation is accepting with respect to α. The relation R, the algorithms $a(\cdot)$, $z(\cdot)$ and $\phi(\cdot)$ are public. The length of the challenges is denoted t_B, and we assume that t_B only depends on the length of the common string α.

In the present context, we will assume that all Σ-protocols we are given satisfy the following security properties. First, (A, B) satisfies a strong flavour of knowledge soundness: Let (a, c, z) and (a, c', z') be two conversations, that are accepting for some given α. If $c \neq c'$, then $\alpha \in R^*$ and, on input α and those two conversations, we can efficiently compute β such that $(\alpha, \beta) \in R$. This is called *special soundness*, and the pair of accepting conversations (a, c, z) and (a, c', z') with $c \neq c'$ is called a *collision* .

It can be shown by the results of Damgaard and Pfitzmann [DP] that a Σ-protocol (A, B) with special soundness is a *proof of knowledge* in the sense of Bellare and Goldreich [BG92], with *knowledge error* 2^{-t_B}. [4]

Finally, we assume (A, B) satisfies *special honest verifier zero knowledge* (special HVZK). This means that we are given a (probabilistic polynomial time) simulator M that on input $\alpha \in R^*$ generates accepting conversations with the exactly same distribution as when A and B execute the protocol on common input α (and A is given a witness β for α), and B indeed honestly chooses its challenges uniformly at random. The simulator is special in the sense that it can additionally take a random string c as input, and output an accepting conversation for α where c is the challenge.

A simple but important fact (see [CDS94]) is that if a Σ-protocol is HVZK, the protocol is perfectly *witness indistinguishable* (WI) [FS89]. Although HVZK by itself is defined with respect to a very much restricted verifier, i.e. an honest one, this means that if for a given instance α there are at least two witnesses β, then even an arbitrarily powerful and malicious verifier cannot distinguish which witness the prover uses.

Examples of Σ-protocols can be based on one-way group homomorphisms, claw-free pairs of trapdoor permutations, random self-reducible languages, and q-one-way group homomorphisms [CD98], which are very attractive from an efficiency point of view.

3.1 Partial Proofs

A General Theorem The following material is relevant to Section 7. Let $f : \{0,1\}^n \rightarrow \{0,1\}$, $f \not\equiv 0, 1$, be a *monotone function*[5], i.e. for all $x \leq y$ we have $f(x) \leq f(y)$.

The *dual* f^* of f is defined as $f^*(x) = f(x \oplus \mathbf{1}) \oplus 1$, where $\mathbf{1}$ denotes the all-one string. This is a monotone function as well.

There is a natural connection between monotone functions and monotone access structures from secret sharing: Γ_f is the collection of all sets $A \subset \{1, \ldots, n\}$ such that $f(A) = 1$, i.e. f applied to the characteristic bit string of A is equal to 1.

Consider an efficient perfect secret sharing scheme $\mathcal{S}$ with access structure Γ_f. Write $\{0,1\}^t$ for its key-space (the set from which secrets are chosen). It is

[4] This is a non-trivial observation, but by standard rewinding techniques one can already verify the slightly weaker property that there is a knowledge extractor that runs in expected time polynomial in the running time of the prover and $1/(\epsilon - 2^{-t_B})$, where ϵ is the prover's success probability.

[5] Usual Boolean ordering: $a \geq b$ iff for all bit positions where a has 1, the corresponding position in b is 1 as well.

efficient in the sense that all operations take time polynomial in n (the number of players) and t. Say that each player receives a number of strings from $\{0,1\}^t$ as share in a secret. Then the total number of strings dealt to all players is denoted size($\mathcal{S}$), the *size* of $\mathcal{S}$.

We say that $\mathcal{S}$ has *completion*[6] if the following holds. Let $A \notin \Gamma_f$ be arbitrary and consider the distribution D_A of shares resulting from $\mathcal{S}$ for players in A. By the condition on A and the perfectness of $\mathcal{S}$, D_A is independent of the secret dealt by the dealer. Completion is satisfied if there exists an efficient probabilistic algorithm that takes as input a random sample from D_A and an arbitrary secret $s \in \{0,1\}^t$, and outputs a full set of shares consistent with s, with distribution identical to a full set of shares of s generated by the honest dealer in $\mathcal{S}$.

Let $\mathcal{F} = \{f_n\}_{n>0}$ denote a collection of efficiently computable monotone functions f_n on n bits. Let R be a binary relation as before. For all $f_n \in \mathcal{F}$ and for all l, consider the collection of all tuples $\alpha = (f_n, \alpha_1, \ldots, \alpha_n)$ such that $\alpha_i \in \{0,1\}^l$ for $i = 1 \ldots n$. Let $\beta = \{\beta_j\}_{j\in I} \subset \{0,1\}^*$ be given where $I \subset \{1,\ldots,n\}$ and $|\beta_j| \leq p(|\alpha_j|)$ for $j \in I$. Then $(\alpha, \beta) \in R_{\mathcal{F}}$ if and only if $f_n(I) = 1$ and $(\alpha_j, \beta_j) \in R$ for $j \in I$. Note that by our assumptions on R and $\mathcal{F}$, the composite binary relation can be tested efficiently. The results from [CDS94] imply the following theorem (see also [Cra96] for the full version)

Theorem 1. *Let a Σ-protocol (A, B) for relation R be given, satisfying special honest verifier zero-knowledge and special soundness. Let $\mathcal{F} = \{f_n\}$ be a family of efficiently computable monotone functions. Assume that there is an efficient perfect secret sharing scheme $\mathcal{S}$ with completion for the* dual *family $\mathcal{F}^* = \{f_n^*\}$ with key-space $\{0,1\}^{t_B}$.*

Then there exists a Σ-protocol $(\overline{A}, \overline{B})$ for relation $R_{\mathcal{F}}$ satisfying special honest verifier zero-knowledge and special soundness. The size $t_{\overline{B}}$ of the challenges is equal to t_B. The total communication complexity of $(\overline{A}, \overline{B})$ is size($\mathcal{S}$) times that of (A, B) plus t_B bits.

Examples of secret sharing schemes satisfying our requirements include *all* efficient linear secret sharing schemes, so in particular Shamir's scheme for the threshold access structure.

A Special Instance: Proof of "OR" In our results to follow, we need a particular, simple instance of the main theorem from [CDS94].

What we use is a slight generalization of a corollary in [CDS94] which enables a prover, given two values (x_1, x_2), corresponding relations (R_1, R_2), and corresponding 3-move Σ-protocols $((A_1, B_1), (A_2, B_2))$, to present a 3-move Σ-protocol (A_{or}, B_{or}) for proving knowledge of a w such that either $(x_1, w) \in R_1$ or $(x_2, w) \in R_2$.

We will describe the protocol assuming the challenges from (A_1, B_1) and (A_2, B_2) are of the same length. This can easily be generalized, as long as the challenge length in the combined protocol is at least as long as the challenges from either protocol. The protocol consists of (A_1, B_1) and (A_2, B_2) running in

[6] This notion corresponds to "semi-smooth" in [CDS94]

parallel, but with the verifier's challenge c split into $c = c_1 \oplus c_2$ by P, who uses c_1 as the challenge for (A_1, B_1), and c_2 as the challenge for (A_2, B_2).

The protocol for A_{or} is as follows: Without loss of generality, say A_{or} knows w such that $(x_1, w) \in R_1$. Let M_2 be the simulator for S_2. Then A_{or} runs $M_2(x_2)$ to generate (m, e, z). It sends the first message of (A_1, B_1), along with m as the first message of (A_2, B_2). On challenge c, it chooses $c_2 = e$, and $c_1 = c \oplus e$. It is able to provide the final response in (A_1, B_1) because it knows w, and the final response in (A_2, B_2) is simply z.

3.2 Commitments using Σ-protocols

For relation $R = \{(x, w)\}$ and Σ-protocol (A, B), consider the relation $R' = \{((x, a), (c, z))\}$ which holds if (a, c, z) is an accepting conversation for (A, B) on (public) input x. We call R' a commitment relation for (A, B), since given an input x for which a party A does not know a witness, there is a commitment protocol for A that works as follows:

1. Say A wishes to commit to a value c. A runs the special simulator M for (A, B) with input x and challenge c, producing accepting conversation (a, c, z) for (A, B).
2. A commits to c by publishing a.
3. A opens the commitment by revealing (c, z).
4. Anyone can verify this by testing if $\phi(x, a, c, z)$ outputs "accept"

It is easy to see that the commitment is binding as long as A does not know a witness for x.

4 General Protocol

Say a relation R has a Σ-protocol (A, B) with t-bit challenges. Assume the commitment relation R' for (A, B) *also* has a Σ-protocol (call it (A', B')) satisfying special soundness and special HVZK with t-bit challenges.[7] Then we show how to construct a 4-move perfect ZK proof of knowledge with knowledge error 2^{-t} for R that does not rely on any intractability assumptions, and whose communication complexity is twice the communication complexity of (A', B') plus the communication complexity of (A, B) plus the number of bits in a commitment. We describe it as a 6-move protocol, but the second and third moves of the first

[7] Since the special soundness and special HVZK properties are not affected by parallel executions of a protocol, if there exists a Σ-protocol for R' with t'-bit challenges for $t' < t$, then a Σ-protocol with t''-bit challenges ($t'' = t'\lceil t/t' \rceil \geq t$) can be constructed using parallel executions of the t'-bit challenge protocol. On the other hand, if there exists a Σ-protocol for R' with t'-bit challenges for $t' > t$, then a Σ-protocol with t-bit challenges can be created by simply allowing the prover to choose the remaining $t' - t$ bits of the challenge, and then sending those bits along with z to the verifier. Again, special soundness and SHVZK are preserved.

part can be combined with the first and second moves of the second part, as in [FS89].

In the protocol, say a prover P wishes to prove to V knowledge of a witness w for x, i.e., a w such that $(x, w) \in R$.

Part 1 Using the commitment relation R' and the input x, V commits to value e and proves knowledge of this value using (A', B'). This proof is WI, so does not reveal any information about e. If V does not give an accepting proof, P halts. Otherwise, say the commitment is (x, m).

Part 2 Let R_{or} be the relation over input pairs $((\alpha_1, \alpha_2), (\beta_1, \beta_2))$ where either $(\alpha_1, \beta_1) \in R$ or $(\alpha_2, \beta_2) \in R'$. Using the Σ-protocol (A_{or}, B_{or}) from Section 3.1, P gives a WI proof that it knows either a witness for x, or how to open the commitment (x, m). An honest prover can do this since it knows a witness for x.

Based on the x for which the prover claims to know a witness, the verifier provides a perfectly hiding commitment and a perfect WI proof [FS89] that he can open it. Note that we set it up so that a witness for x can be computed efficiently given any two distinct openings of the commitment.

Next, the prover gives a perfect WI proof of knowledge that he can open the commitment or knows the witness he claims. So why should this convince the verifier that, unconditionally and not relying on intractability assumptions, the prover knows the claimed witness?

This is by the existence of the following knowledge extractor. First, K sets up the commitment and runs, as an honest verifier, the protocol with the prover. Intuitively, by rewinding the prover during the second part of the proof, K either extracts a witness for x, in which case we are done, or extracts an opening of the commitment. In the latter case, and assuming for simplicity that there is an overwhelming number of different ways to open a commitment, we know by the witness indistinguishability of K's proof of knowledge of an opening that with very high probability the opening extracted from the prover is different from the one known to K. With two different ways to open the commitment, K can find a witness for x and we are done.

The black-box simulation for this protocol intuitively works by rewinding the verifier the simulator extracts an opening for the commitment and thus can give the perfect WI proof to the verifier that he knows the witness or can open the commitment. Thus we achieve perfect zero knowledge.

4.1 A Remark on Computational ZK Proofs of Knowledge

We emphasize that our method allows us to construct perfect ZK proofs of knowledge from Σ-protocols without making any complexity assumptions, *not even on the underlying problem itself.* If we were willing to do that, one could construct a ZK proof of knowledge for any Σ-protocol as follows. Assume there is an invulnerable generator G [FS89] for relation R and assume R has a Σ-protocol. Then given problem instance y, where P claims to know a solution, V

makes instances z_1, z_2 using G and shows that he knows a solution to z_1 or or z_2. This will naturally be witness hiding, since there are at least two witnesses. The P proves she knows a solution to y or z_1 or z_2. The zero-knowledge property is obtained immediately, and soundness follows by a standard argument, assuming P is poly time and G is invulnerable.

5 Security of the general protocols

Theorem 2 is proved in the appendix, while the proof of Theorem 3 is omitted due to space limitations.

Theorem 2. *If relation R has a Σ-protocol (A, B) with t-bit challenges, and commitment relation R' associated with (A, B) also has a Σ-protocol (A', B') with t-bit challenges, then R has a 4-move perfect ZK proof of knowledge with no intractability assumptions, with knowledge error at most 2^{-t}, and with communication complexity twice that of (A', B') plus that of (A, B) plus the number of bits in a commitment.*

Theorem 3. *If relation R has a Σ-protocol (A, B) with t-bit challenges, and commitment relation R' associated with (A, B) also has a Σ-protocol (A', B') with t-bit challenges, then R has a 6-move perfect ZK proof of knowledge with no intractability assumptions, with knowledge error at most 2^{-t}, and with communication complexity twice that of (A, B) plus that of (A', B') plus the number of bits in a commitment.*

6 Instantiations of the General Protocol

6.1 q-one-way group homomorphisms (q-OWGH)

Definition q-OWGH's are introduced in [CD98]. We briefly review the definition.[8]

Let q be a fixed prime, and let H and G be finite Abelian groups, with efficient basic operations (as usual: random sampling, equality testing, group operations). Let $f : H \to G$ be a one-way group homomorphism (easy to compute on elements from its domain, but hard to invert on a random element from its image).

Definition 1. *f is a q-one-way group homomorphism, if given f and an arbitrary x in f's image, one can efficiently compute $\tilde{w} \in H$ such that $f(\tilde{w}) = x^q$.*

Lemma 1. *Given f, z, x, $0 < i < q$ with $f(z) = x^i$, one can efficiently compute w with $f(w) = x$.*

To prove the lemma, let i, j, β, z, $\tilde{w}$ be such that $ij = 1 + \beta q$, $f(z) = x^i$, $f(\tilde{w}) = x^q$. Then $f(z^j) = x^{1+\beta q}$. Hence, $f(z^j \tilde{w}^{-\beta}) = x$, and define $w = z^j \tilde{w}^{-\beta}$.

[8] We also remove a redundant requirement from the definition given in [CD98].

Σ-protocols First we give the basic Σ-protocol from [CD98]. A proves knowledge of an f-preimage w of $x \in G$. It starts by A choosing $p \in_R H$ and sending $a = f(p)$ to B. Next, B chooses $c \in_R \{0, \ldots, q-1\}$ and sends c to A. Finally, A sends $z = pw^c$ to B, who checks that $f(z) = ax^c$ (These computations are performed in the group G).

The proof that this basic protocol, which is clearly a generalization of the RSA-protocol from Guillou/Quisquater [GQ88] and the DL-protocol from Schnorr [Sch89], is a Σ-protocol with satisfies completeness, special soundness, and special HVZK is given in Lemma 25 from [CD98]. Note that it is in fact a proof of knowledge with knowledge error $1/q$. (See [CD98] for the proofs that q-OWGH's and the corresponding Σ-protocols can be constructed for RSA and Discrete Logarithms.) It remains to be shown that this Σ-protocol admits a Σ-protocol for the commitment relation.

This means that we have to provide a Σ-protocol for A to prove knowledge of a (e, s) for (x, m), where (m, e, s) is an accepting conversation for $x \in G$ in the basic Σ-protocol for q-OWGH's. i.e. A has to prove knowledge of e, s such that $f(s)x^{-e} = m$. The Σ-protocol follows:

1. A chooses $\rho \in_R H$ and $\sigma \in_R \{0, \ldots, q-1\}$ and sends $a = f(\rho)x^{\sigma}$ to B.
2. B chooses $c \in_R \{0, \ldots, q-1\}$ and sends c to A.
3. A has to find $z_1 \in H$ and $z_2 \in \{0, \ldots, q-1\}$ such that $f(z_1)x^{z_2} = am^c$. A computes $\tilde{w}$ such that $f(\tilde{w}) = x^q$ and integers $0 \leq \alpha < q$ and β such that $\sigma - ec = \alpha + \beta q$. A sends $z_1 = \rho s^c \tilde{w}^{\beta}$ and $z_2 = \alpha$ to B, who checks that $f(z_1)x^{z_2} = am^c$.

In the general protocol, the $\oplus$ operation for the challenge bits in the "OR" protocol may be replaced by addition mod q.

Assuming that the members of groups G and H can be represented with l bits, the communication complexity of the basic Σ-protocol for q-OWGHs is $3l$, and that of the commitment Σ-protocol for q-OWGHs is $4l$. Therefore, the communication complexity of the 4-move ZK proof of knowledge for q-OWGHs is $12l$, and that of the 6-move ZK proof of knowledge for q-OWGHs is $11l$.

6.2 Optimized Discrete-Log

Figure 1 gives an optimized version of the general protocol for discrete logarithms (DL-1). The proof of knowledge of either how to open the commitment, or a discrete log of X, is actually combined, with B being the first message of both proofs, and z and r taking the place of the challenge split and last messages of both proofs. It's communication complexity is $9|p|$ and it has a total of 10 exponentiations.

6.3 RSA with prime exponents

By using the protocol for q-one-way group homomorphisms, we can directly implement a ZK proof of knowledge of RSA decryption when the public exponent

V		P
$i, j, i', j' \in_R Z_q$ $h \leftarrow g^i X^j \bmod p$ $h' \leftarrow g^{i'} X^{j'} \bmod p$		
	$\xrightarrow{h, h'}$	
		$c, p, r \in_R Z_q$ $B \leftarrow g^{p+xcr} h^r \bmod p$
	$\xleftarrow{c, B}$	
$i'' \leftarrow ic + i' \bmod q$ $j'' \leftarrow jc + j' \bmod q$ $k \in_R Z_q$		
	$\xrightarrow{i'', j'', k}$	
		Check $g^{i''+xj''} \stackrel{?}{\equiv} h^c h' \bmod p$ $z \leftarrow p + kx \bmod q$
	$\xleftarrow{r, z}$	
Check $g^{z+ir} \stackrel{?}{\equiv} X^{k-(c+j)r} B \bmod p$		

Fig. 1. DL-1: Given a public value X, P proves that it knows the discrete log x of X.

is a prime q. However, the soundness error is $1/q$, which implies that for small q, the protocol might need to be repeated a number of times to achieve negligible soundness error. However, we can also get an efficient 4-move protocol for any small prime RSA exponent q with soundness error q^{-t} assuming that the prover knows a q^tth root of the encryption. (which, for instance, the prover could find if it knew the the factorization of the RSA modulus n). For concreteness, let us assume the public exponent is 3, although the protocol will work for any prime public exponent.

First we give a 3-move Σ-protocol with two parameters, t and t', which has soundness error 3^{-t} and proves knowledge of a $3^{t'+1}$th root. It requires that A knows a $3^{t+t'}$th root. Assume the public input is $y \in Z_n^*$, and that A knows x, where $x^{3^{t+t'}} \equiv y \bmod n$.
Protocol:

1. A chooses $r \in_R Z_n^*$ and sends $m \equiv r^{3^{t+t'}} \bmod n$ to B.
2. B chooses $e \in Z_{3^t}$ and sends e to A.
3. A sends $z \equiv rx^e \bmod n$, and B checks that $z^{3^{t+t'}} = my^e \bmod n$.

In a way, A is attempting to prove that he knows a $3^{t+t'}$th root, but ends up proving that he knows a $3^{t'+1}$th root.

Lemma 2. *The protocol above is a Σ-protocol.*

Proof. Completeness is trivial. To prove special soundness, consider two accepting conversations (m, e, z) and (m, e', z') with $e \neq e'$. These give an equation

$(z/z')^{3^{t+t'}} = y^{e-e'} \bmod n$. Also, $\gcd(e-e', 3^{t+t'})$ is 3^s for some $s < t$, so by the Extended Euclidean Algorithm, i and j can be found such that $i(e-e')+j3^{t+t'} = 3^s$. Then the equation can be transformed into $((z/z')^i y^j)^{3^{t+t'}} \equiv y^{3^s} \bmod n$. Since raising to 3 is a permutation in Z_n^*, assuming 3 is a valid RSA public key for n, we immediately get a $3^{t'+1}$th root of y from this. Special HVZK can be shown by using the simulator that on input c chooses r randomly and outputs $(r/y^c, c, r)$.

We do not know how to construct a general commitment protocol for the above Σ-protocol. However, we will be able to substitute a different commitment protocol that works just as well for constructing an efficient 4-move ZK proof for cube roots. We describe the new ZK proof here.

Part 1 First V commits to a value b with commitment $C = y^b r^{3^{2t}} \bmod n$ and proves knowledge of a 3^tth root of C or Cy^{-1}, using the Σ-protocol above with $t' = t$, combined with the Proof of "OR" method of Section 3.1. This proof is WI, so does not reveal any information about b. If V does not give an accepting proof, P halts.

Part 2 Using Section 3.1, P proves knowledge of either (1) a cube root of y, (2) a cube root of C, or (3) a cube root of Cy^{-1}. An honest prover can do this since it knows a cube root of y.

As in Section 4, the protocol above can be made into a 4-round protocol by combining the second and third steps of the Σ-protocol of Part 1 with the first and second steps, respectively, of the Σ-protocol of Part 2.

Theorem 4. *The protocol above is a 4-move perfect ZK proof of knowledge of a cube root of y with no intractability assumptions, with knowledge error at most 3^{-t}, and with communication complexity five times that of the Σ-protocol for cube roots (i.e., the Σ-protocol above with $t' = 0$), plus the number of bits in a commitment.*

The proof is similar to that of Theorem 2, and is omitted due to space limitations.

We stress again that the protocol above generalizes to any small prime RSA public key. Also note that we can construct an efficient 6-move protocol for the same problem, as in Section 4.

7 Monotone Composition

We discuss a further application of our results. It's is now possible to combine Theorems 2 and Theorem 1:

Theorem 5. *Let relation R have a Σ-protocol (A, B) with t-bit challenges, and let the commitment relation R' associated with (A, B) also have a Σ-protocol (A', B') with t-bit challenges. Let $\mathcal{F} = \{f_n\}$ be a family of efficiently computable monotone functions. Assume that there is an efficient perfect secret* *sharing scheme $\mathcal{S}$ with completion for the* dual *family $\mathcal{F}^* = \{f_n^*\}$ with key-space $\{0,1\}^t$. Then relation $R_{\mathcal{F}}$ has a 4-move perfect ZK proof of knowledge with no intractability assumptions, with knowledge error at most 2^{-t}, and with communication complexity size($\mathcal{S}$) times that of (A', B') plus size($\mathcal{S}$) times that of (A, B) plus the number of bits in a commitment.*

To prove this theorem it sufficices to show that the protocol $(\overline{A}, \overline{B})$ for relation $R_{\mathcal{F}}$ guaranteed by 1 admits a Σ-protocol for its associated commitment relation $R'_{\mathcal{F}}$. This can be constructed from the Σ-protocol for the commitment relation R' of R, by invoking Theorem 1 on the commitment relation R' and its associated Σ-protocol (A', B'). A similar combination is possible with Theorem 3.

We now identify an interesting class of secret sharing schemes suitable for our purposes, namely *linear secret sharing schemes* (LSSS), and establish an example of a super-polynomial gap with earlier work.

Let K denote a finite field. In an LSSS each player receives some linear combination of the secret $s \in K$ and some random values $\rho \in K$ chosen by the dealer. The distribution of a secret in fact consists of the dealer selecting a random vector with s in its first coordinate and multiplying a public matrix with this vector to get the shares. In the public matrix, each row is associated with a player and a player may have several rows. In Shamir's scheme, for instance, the matrix is a Van der Monde-matrix.

The matrix has the property that the K-span of rows corresponding to players in a set A contains the vector $(1, 0, \ldots, 0)$ iff $A \in \Gamma_f$. It is in this sense that Karchmer and Wigderson [KW93] say that the matrix (with its assignments of players to rows), which they call *monotone span program* (MSP), computes the function f. The *size* of an MSP is the number of rows in the matrix, which is equal to the total number of field elements given to players in the corresponding LSSS.

It is easy to see an MSP $\mathcal{M}$ gives rise to efficient perfect secret sharing schemes with completion (the latter comes down to solving a system of linear equations and sampling a random element from its solution space). The time required for all operations is polynomial in $\log(|K|)$ and size($\mathcal{M}$).

In the following we set $K = GF(2)$ for simplicity. It's easy to see that to increase the key-space from $\{0,1\}$ to $\{0,1\}^t$ it suffices to execute the basic LSSS t times, and it will still be efficient with completion.

It is well known that if f is computed by an MSP of size m then f's dual f^* is also computed by an MSP of size m. Another relevant fact about MSP's is that all monotone functions f can be computed, and that monotone formula complexity of f is an upperbound on the minimal size of an MSP computing f. Furthermore, there are monotone functions which are computed by polynomial size MSP's (over $GF(2)$), but require super-polynomial monotone *circuits* (the ODDFACTOR function [BGK+96]) (and hence its dual ODDFACTOR* also has super-polynomial monotone circuit complexity).

With this in mind, we can now show that our protocol gives in some cases a superpolynomial efficiency improvement over earlier work, namely [SCPY94] on perfect ZK for monotone formula closure of Random Self-reducible Languages.

To see this, let the family $\mathcal{F}$ of monotone functions occurring in Theorem 5 be the family of ODDFACTOR functions, and choose as the relation R anything derived from a random self-reducible problem, for instance the Discrete Logarithm problem. Theorem 5 now provides an efficient protocol for the composite language derived from R and $\mathcal{F}$, while [SCPY94] will have super-polynomial complexity, since that result is polynomial in the monotone formula complexity of the monotone composition function, ODDFACTOR in our example.

References

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Advances in Cryptology—CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer-Verlag, 1993, 16–20 August 1992.

[BGK+96] László Babai, Anna Gál, János Kollár, Lajos Rónyai, Tibor Szabó, and Avi Wigderson. Extremal bipartite graphs and superpolynomial lower bounds for monotone span programs. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 603–611, Philadelphia, Pennsylvania, 22–24 May 1996.

[BJY97] M. Bellare, M. Jakobsson, and M. Yung. Round-optimal zero-knowledge arguments based on any one-way function. In *Advances in Cryptology—EUROCRYPT 97*, volume 1233 of *Lecture Notes in Computer Science*, pages 280–305. Springer-Verlag, 1997.

[BMO90] Mihir Bellare, Silvio Micali, and Rafail Ostrovsky. Perfect zero-knowledge in constant rounds. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 482–493, Baltimore, Maryland, 14–16 May 1990.

[CD98] R. Cramer and I. Damgård. Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free? In Hugo Krawczyk, editor, *Advances in Cryptology—CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer-Verlag, 17–21 August 1998.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology—CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer-Verlag, 21–25 August 1994.

[Cra96] R. Cramer. *Modular Design of Secure yet Practical Cryptographic Protocols*. PhD thesis, CWI & Univ. of Amsterdam, November 1996.

[CRY89] *Advances in Cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990, 20–24 August 1989.

[CRY93] *Advances in Cryptology—CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*. Springer-Verlag, 22–26 August 1993.

[Dam89] Ivan Bjerre Damgård. On the existence of bit commitment schemes and zero-knowledge proofs. In CRYPTO'89 [CRY89], pages 17–27.

[Dam93] Ivan B. Damgård. Interactive hashing can simplify zero-knowledge protocol design without computational assumptions (extended abstract). In CRYPTO'93 [CRY93], pages 100–109.

[DP] I. Damgård and B. Pfitzmann. Sequential iteration of interactive arguments. journal version of ICALP'98 paper.

[FMY98] Y. Frankel, P. D. MacKenzie, and M. Yung. Robust efficient distributed rsa-key generation. In STOC'98 [STO98], pages 663–672.

[FS89] U. Feige and A. Shamir. Zero knowledge proofs of knowledge in two rounds. In CRYPTO'89 [CRY89], pages 526–545.

[GK96a] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for np. *Journal of Cryptology*, 9(3):167–189, Summer 1996.

[GK96b] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, February 1996.

[Gol95] Oded Goldreich. Foundations of cryptography (fragments of a book), February 1995.

[GQ88] Louis Claude Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *Advances in Cryptology—CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer-Verlag, 1990, 21–25 August 1988.

[GRR98] R. Genarro, M. O. Rabin, and Tal Rabin. Simplified vss and fast-track multiparty computations with applications to threshold cryptography. In *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing*, pages –, 1998.

[GSV98] O. Goldreich, Amit Sahai, and Salil Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In STOC'98 [STO98], pages 399–408.

[KW93] M. Karchmer and A. Wigderson. Characterizing non-deterministic circuit size. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 532–540, San Diego, California, 16–18 May 1993.

[Sch89] C. P. Schnorr. Efficient identification and signatures for smart cards. In CRYPTO'89 [CRY89], pages 239–252.

[SCP93] A. De Santis, G. Di Crescenzo, and G. Persiano. Secret sharing and perfect zero knowledge. In CRYPTO'93 [CRY93], pages 73–84.

[SCPY94] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula closure of SZK. In *35th Annual Symposium on Foundations of Computer Science*, pages 454–465, Santa Fe, New Mexico, 20–22 November 1994. IEEE.

[SKS91] Takeshi Saito, Kaoru Kurosawa, and Kouichi Sakurai. 4 Move perfect ZKIP of knowledge with no assumption. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology—ASIACRYPT '91*, volume 739 of *Lecture Notes in Computer Science*, pages 321–330, Fujiyoshida, Japan, 11–14 November 1991. Springer-Verlag. Published 1993.

[STO98] *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, Dallas, Texas, 23–26 May 1998.

Appendix A: Formal Definitions

Proofs of knowledge were defined in [BG92]. Much of the formalism below is from that paper.

Let R be a relation as defined in Section 3. Let $R(x) = \{y : (x, y) \in R\}$ be the *witness set* of x. Let $L_R = \{x : \exists y \text{ such that } (x, y) \in R\}$.

Let A, B be interactive functions (as defined in [BG92]). Let $A(x)$ denote the interactive function with input x. Let $A(x; B)$ be a random variable describing the output of A when interacting with B on common input x. Let $A_z(x;B)$ be a random variable describing the output of A with auxiliary input z when interacting with B on common input x, Let $\mathrm{tr}_{A,B}(x)$ denote a transcript of the interaction between A and B on common input x. For a verifier V, modeled as an interactive function, let $\mathrm{acc}_V(x)$ be the set of accepting transcripts for V on input x. For a probabilistic oracle machine C, let $C^{A(x)}(x)$ denote a a random variable describing the output of C with oracle $A(x)$ and input x, with the probability being over the random choices of M and A. (Technically, we can assume A's random bits are fixed [BG92].) We can assume that C can query the oracle for any part of a conversation, by supplying the appropriate prefix to the conversation. (In other words, C can arbitrarily "rewind" the interactive function A.)

We say that a function $f(n)$ is *negligible* if for any $poly(\cdot)$, there is an n' such that for $n > n'$, $f(n) < 1/poly(n)$.

Definition 2. *Let* $\kappa : \{0,1\}^* \to [0,1]$. *An* interactive proof of knowledge *system for a relation* R *is a pair of interactive functions* (V, P), *where* V *is computable in probabilistic polynomial-time, satisfying:*

1. *Completeness:* $\forall x \in L_R$, $\Pr(tr_{V,P}(x) \in acc_V(x)) = 1$
2. *Soundness: There exists a probabilistic expected polynomial-time oracle machine* E *(the* knowledge extractor*) such that* $\forall P' \forall x \in L_R$, *it is the case that* $E^{P'(x)}(x) \in R(x) \cup \perp$, *and* $\Pr(E^{P'(x)}(x) \in R(x)) \geq \Pr(tr_{V,P'}(x) \in acc_V(x)) - \kappa(x)$.

The $\perp$ symbol is the output of the oracle that indicates "failure."

All knowledge extractors that we consider satisfy the following slightly stronger soundness property, which basically states that the knowledge extractor can differentiate the prover's failure from its own failure.

- There exists a probabilistic expected polynomial-time oracle machine E such that $\forall P' \forall x$, it is the case that $E^{P'(x)}(x) \in R(x) \cup \perp \cup \triangle$, and
  - $\Pr(E^{P'(x)}(x) = \triangle) = \Pr(\mathrm{tr}_{V,P'}(x) \notin \mathrm{acc}_V(x))$,
  - $\Pr(E^{P'(x)}(x) =\in R(x)) \geq \Pr(\mathrm{tr}_{V,P'}(x) \in \mathrm{acc}_V(x)) - \nu(|x|)$, and
  - $\Pr(E^{P'(x)}(x) = \perp) \leq \nu(|x|)$.

In this case, $\triangle$ indicates that V would not accept the proof of P', and $\perp$ indicates that E failed, even though V accepted the proof of P'.

Two random variables S and S' are perfectly indistinguishable if the distributions of S and S' are identical.

Definition 3. *A proof system* (V, P) *is perfect zero knowledge over* R *if there exists a probabilistic expected polynomial-time oracle machine* M *(the* simulator*) such that for any probabilistic polynomial-time* V', *for any* $(x, y) \in R$, *and any auxiliary input* z *to* V', *the two random variables* $V'_z(x;P)$ *and* $M^{V'_z(x)}(x, z)$ *are perfectly indistinguishable.*

Taking away z from the inputs to M implies that M is a *black-box* simulator. All simulators in this paper are black-box.

Definition 4. *The proof system consisting of a pair of interactive functions (V, P), where both V and P are computed in probabilistic polynomial time, is witness indistinguishable (WI) over R if for any probabilistic polynomial-time V', any large enough input x, any $y_1, y_2 \in R(x)$, and for any auxiliary input z to V', $tr_{V'_z,P_{y_1}}(x)$ and $tr_{V'_z,P_{y_2}}(x)$ are perfectly indistinguishable.*

Appendix B: Proofs

Proof of Theorem 2

Proof. **Communication Complexity:** Straightforward.
Completeness: Straightforward.
Soundness: Let E_{or} be the extractor guaranteed by the special soundness property of (A_{or}, B_{or}) [DP]. We construct an extractor E as follows. Say the prover is P^*, and assume its bits are fixed.

1. E performs Part 1 like a true verifier, say using commitment pair $((x,m),(e,s))$. If P^* does not give a well-formed challenge, E outputs $\triangle$.
2. E runs E_{or} using oracle P^* (in its current state after Part 1, i.e., after receiving commitment m, and the proof transcript T from Part 1).
3. Based on the output of E_{or}, E performs different actions:
   (a) if E_{or} outputs $\triangle$, E outputs $\triangle$ and stops. (In this case, P^* failed to provide a valid proof.)
   (b) If E_{or} outputs $\perp$, E outputs $\perp$ and stops. (In this case P^* provided a valid proof, but E_{or} failed.)
   (c) If E_{or} outputs a witness w for x, E outputs w and stops.
   (d) If E_{or} outputs a witness (e', s') to commitment (x, m), with $e \neq e'$, E uses the two conversations (m, e, s) and (m, e', s') and the special soundness feature of (A, B) to generate a witness w for x. Then E outputs w and stops.
   (e) Otherwise E repeatedly performs steps 1 and 2 (sequentially) until either cases 3c or 3d would occur, at which point E finishes as indicated for that case. In step 1, instead of stopping if P^* does not provide a well-formed challenge, E simply rewinds and tries again.

It is obvious that $\Pr(E^{P^*(x)}(x) = \triangle) = \Pr(\mathrm{tr}_{V,P^*}(x) \notin \mathrm{acc}_V(x))$, and that $\Pr(E^{P^*(x)}(x) = \perp)$ (i.e., the soundness error of E) is bounded by the probability that E_{or} outputs $\perp$, which by the special soundness of (A_{or}, B_{or}), is at most 2^{-t} [DP].

To analyze the expected running time of E we use the following fact: given that a witness (e', s') for commitment (x, m) is extracted, the probability that $e = e'$ is 2^{-t}, since the protocol in Part 1 is witness indistinguishable [CDS94] and e is drawn randomly from a set of size 2^t. (We actually only use that the probability that $e = e'$ is at most $1/2$.)

Let q be the probability of E hitting case 3c, 3d, or 3e. Note that this probability is computed for E starting at step 1, and thus is averaged over all possible transcripts produced in step 1, including the possibility of E stopping. The probability of hitting case 3e is at most $q/2$, by the fact stated above, and therefore the probability of either case 3c or 3d occurring (implying success for E) is at least $q/2$. Thus the expected time of E is easily seen to be polynomial.

Zero-Knowledge: We construct a simulator SIM (that uses V^* as a black-box) as follows.

1. SIM plays the part of the P in Part 1 (i.e., the part of B' in (A', B')). If P does not accept the proof (i.e. V^* failed in its proof), SIM halts, like P would.
2. Otherwise SIM repeatedly runs the extractor E' for protocol (A', B') in Part 1 until E' does not output $\triangle$.
3. If E' outputs a witness to the commitment, SIM plays the part of the prover in the (A_{or}, B_{or}) in Part 2, which it can do since it has found a witness to the commitment.
4. If E' outputs $\perp$, SIM uses the simulator M_{or} to generate a conversation with a random challenge, and attempts to play the prover in the (A_{or}, B_{or}) in Part 2.
5. E' performs one of the following, depending on the challenge of V^*.
   (a) If V^* does not send a valid challenge, SIM stops, like a true prover would.
   (b) If V^* actually sends the challenge generated in the simulated conversation, SIM finishes the proof with the simulated responses.
   (c) If neither of the previous cases hold, then SIM rewinds V^* and generates conversations with random challenges until V^* actually sends the challenge generated in the simulated conversation, so that SIM can finish the proof.

Obviously SIM produces the same distribution as an honest prover (including when V^* misbehaves).

Say the probability of P accepting the proof of V^* in Part 1 is q'. Then the probability SIM reaches step 2 is q', and the extractor E' produces $\triangle$ with probability $1 - q'$. Thus, the overall expected number of rewinds in step 2 is constant.

Since (A', B') has special soundness, we can assume E' works with soundness error at most 2^{-t} [CD98]. Then the probability that the E' outputs $\perp$ (and thus SIM reaches step 4) is at most 2^{-t}. If E' does not output $\perp$, SIM can finish the protocol directly. Otherwise, SIM attempts to finish the protocol by guessing the challenge of V^*.

Let q be the probability that V^* sends a valid challenge, given that step 4 is reached. (Note that q will depend on the conversation transcript from step 1.) The total probability of reaching step 5c is at most $q2^{-t}$ (considering the probability of reaching step 4 is at most 2^{-t}). Moreover, the probability of V^* sending a valid challenge in (A_{or}, B_{or}) and SIM guessing it correctly is $q2^{-t}$. So the expected number of rewinds for SIM in step 5c is constant. Thus the total expected running time of SIM is polynomial.