CWI Quarterly Volume 8 (2) 1995, pp. 111 – 127

Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols

Ronald Cramer
Berry Schoenmakers
CWI, P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands
e-mail: {cramer, berry@cwi.nl}

Ivan Damgård
Matematisk Institut, Aarhus University, Ny Munkegade, DK-8000 Århus C, Denmark
e-mail: ivan@daimi.aau.dk

Suppose we are given a proof of knowledge $\mathcal{P}$ in which a prover demonstrates that he knows a solution to a given problem instance. Suppose also that we have a secret sharing scheme $\mathcal{S}$ on n participants. Then under certain assumptions on $\mathcal{P}$ and $\mathcal{S}$, we show how to transform $\mathcal{P}$ into a witness indistinguishable protocol, in which the prover demonstrates knowledge of the solution to a subset of n problem instances corresponding to a qualified set of participants. For example, using a threshold scheme, the prover can show that he knows at least d out of n solutions without revealing which d instances are involved. If the instances are independently generated, this can lead to witness hiding protocols, even if $\mathcal{P}$ did not have this property. Our transformation produces a protocol with the same number of rounds as $\mathcal{P}$ and communication complexity n times that of $\mathcal{P}$. Our results use no unproven complexity assumptions.

1. Introduction

In this work[1] we assume that we are given an interactive proof where the prover P convinces the verifier V that P knows some secret. Typically, the secret is the preimage under some one-way function of a publicly known piece of information. Thus the secret could be for example a discrete log or an RSA root. Such a proof is called a proof of knowledge [5], and can be used in practice to design identification schemes or signature systems.

[1] Partly done during Cramer's and Schoenmaker's visit at Aarhus University.

We assume in the following that the proof of knowledge has a special form in that the verifier only sends uniformly chosen bits. This is also known as a *public coin protocol*. For simplicity, we restrict ourselves to 3-round protocols, where the prover speaks first (generalization of our results to any number of rounds is possible). We also assume that the protocol is honest verifier zero-knowledge, i.e. the protocol does not reveal anything (for example about the prover's secret) to the honest verifier, but it is not necessarily secure against a cheating verifier.

Numerous protocols are known to satisfy the conditions described above. Concrete examples are Schnorr's discrete log protocol [12] and Guillou-Quisquater's RSA root protocol [8]. None of these protocols are known to be zero-knowledge or even witness hiding. In general, a parallelization of a sequential zero-knowledge proof [7] will often satisfy the conditions.

The second ingredient we need is a secret sharing scheme, i.e. a scheme for distributing a secret among a set of participants such that some subsets of them are qualified to reconstruct the secret while other subsets have no information about it. The collection of qualified subsets is called the access structure. The secret sharing scheme has to satisfy some properties which will be made more precise below. Shamir's secret sharing scheme [13] has the properties we need.

Our main result uses a proof of knowledge $\mathcal{P}$, an access structure Γ for n participants, and a secret sharing scheme $\mathcal{S}$ for the access structure dual to Γ to build a new protocol, in which the prover shows that he knows some subset of n secrets. More precisely, we fix a correspondence between secrets and participants in Γ, and P shows that he knows a set of secrets corresponding to a qualified set in the access structure of Γ (see Section 3 for details on access structures). The protocol is witness indistinguishable, i.e. the prover reveals no Shannon information about which qualified subset of secrets he knows.

As a corollary, we obtain a general method for improving the security of honest verifier zero-knowledge protocols. Of course, honest verifier zero-knowledge is a weak property, and it is much easier to design protocols that are honest verifier zero-knowledge, than to get more general security properties. On the other hand, honest verifier zero-knowledge is not in itself sufficient for use of the protocol in practice. For practical use, we would need at least a witness-hiding protocol, where it can be shown that whatever the verifier learns will not help him to compute the prover's secret.

This problem would be solved if we had a general method for transforming the honest verifier zero-knowledge protocol into a protocol with stronger security properties. From our results, a transformation follows that constructs witness-hiding protocols. Although witness-hiding is a weaker property than zero-knowledge, it can replace zero-knowledge in many protocol constructions, including identification schemes. Our transformation preserves the round complexity, increases communication complexity by a factor of two and will not need any computational assumptions. Our results can therefore be seen as giving a general method simplifying the design of witness-hiding protocols.

After surveying related work, we give in the following two sections more details on the protocols and the secret sharing schemes we consider. Section 4 then contains the main result and corollaries, Section 4.1 gives some concrete examples, and Section 5 contains an example of an application.

1.1. Related Work

Our techniques are to some extent related to those of De Santis et al. [10]. The models are quite different, however: [10] considers non-interactive zero-knowledge proofs of membership, while we consider interactive proofs of knowledge. Also, [10] considers variants of the quadratic residuosity problem, while we consider any problem that affords a protocol of the right form.

In some recent independent work, De Santis et al. [11] apply techniques similar to ours to proofs of membership in random self-reducible languages. This leads to perfect zero-knowledge proofs for monotone Boolean operations over such languages.

In [4], Feige and Shamir introduce the concepts of witness indistinguishable and witness hiding protocols and prove the existence of witness hiding protocols for a large class of problems, including the ones we consider (Corollary 4.4). This was done using general zero-knowledge techniques and the assumption that one-way functions exist. Compared to [4], our result shows that if we start from a proof of knowledge with properties as described above, witness hiding protocols can be constructed much more efficiently and without using computational assumptions.

In [3], a transformation from honest verifier zero-knowledge proof was given for protocols including the type we consider. That transformation produced zero-knowledge protocols, but on the other hand greatly increased the communication and round complexity so that, contrary to ours, the practical value of that transformation is quite limited. If the target is zero-knowledge, however, the increased round complexity seems to be unavoidable.

2. Proofs of Knowledge

Most of our formalism with respect to protocols follows Feige and Shamir [4], but some of the technicalities have been omitted in this extended abstract.

Our protocols take place between a prover P and a verifier V, both of which are interactive probabilistic polynomial time Turing machines. Both prover and verifier have private auxiliary input tapes. P's auxiliary input is denoted by w. There is a common input x of length k bits (k is sometimes called the security parameter). In the following, a probability is called negligible, if as a function of k, it converges to 0 faster than any polynomial fraction.

The proof system is designed with respect to a binary relation $R = \{(x, w)\}$, which can be tested in polynomial time. For any x, its *witness set* $w(x)$ is the set of w's, such that $(x, w) \in R$. The purpose of the protocol is for P to show that it has been given an element of $w(x)$ on its private input tape. We assume that completeness holds with probability 1, i.e. if indeed $w \in w(x)$, then the verifier always accepts.

As mentioned, we restrict ourselves to three round public coin protocols for simplicity (the three round restriction can be easily removed). Conversations in the protocol will be ordered triples of the form

$$m_1, c, m_2$$

The second message in the protocol is a random bit string c chosen by the verifier. We refer to this as a challenge, and to the prover's final message as the answer. The length of c is such that the number of possible c-values is super-polynomial in k.

We assume that the protocol satisfies knowledge soundness in the following sense: for any prover P^*, given two conversations between P^* and V: $(m_1, c, m_2), (m_1, c', m_2')$, where $c \neq c'$, an element of $w(x)$ can be computed in polynomial time. We call this the *special soundness property.* It is easily seen to imply the standard soundness definition, which calls for the existence of a knowledge extractor, which can extract a witness in polynomial time from any prover that is successful with non-negligible probability.

Although special soundness is less general than the standard definition, all known proofs of knowledge have this property, or at least a variant where computation of the witness follows from some small number of correct answers. Assuming special soundness is therefore not a serious restriction.

A protocol which is sound and complete in the above sense is called *a proof of knowledge for the relation R.*

Finally, we assume that the protocol is *honest verifier zero-knowledge*: there is a simulator S that on input x produces conversations that are indistinguishable from real conversations with input x between the honest prover and the honest verifier. For simplicity we assume perfect indistinguishability in the following; generalization to other flavors of indistinguishability is easy. Most known honest verifier zero-knowledge protocols in fact satisfy something stronger, viz. that there is a procedure that can take any c as input and produce a conversation indistinguishable from the space of all conversations between the honest prover and verifier in which c is the challenge. We call this *special honest verifier zero-knowledge.*

We will later need the concepts of *witness indistinguishable* (WI) and *witness hiding* (WH) protocols, which were introduced in [4]. Informally, a protocol is witness indistinguishable if conversations generated with the same x but different elements from $w(x)$ have indistinguishable distributions, i.e. even a cheating verifier cannot tell which witness the prover is using. If the problem instance x is generated with a certain probability distribution by a generator G which outputs pairs (x, w) with $w \in w(x)$, we can define the concept of *witness hiding.* A protocol is witness hiding over G, if it does not help even a cheating verifier to compute a witness for x with non-negligible probability when the x is generated by G. We refer to [4] for details.

With respect to the witness hiding property, we can already now note the following:

PROPOSITION 1 *Let $\mathcal{P}$ be a three round public coin proof of knowledge for relation R. If $\mathcal{P}$ is honest verifier zero-knowledge, then $\mathcal{P}$ is witness indistinguishable.*

Proof We trivially have WI for conversations with the honest verifier, since conversations generated with any witness will lead to the same distribution as produced by the simulator. But then conversations using different witnesses will still have the same distribution if we restrict to conversations with a fixed c occurring as the challenge. Since the only difference between the honest verifier and a general one lies in the distribution with which c is chosen, we get also WI against an arbitrary verifier. □

In many concrete cases, this proposition is not interesting because there is only one witness, in which case WI is trivial and cannot imply anything. Nevertheless, Proposition 1 will be needed in the following for technical reasons.

2.1. An Example

As a concrete example of a protocol with the properties we need, we present Schnorr's protocol from [12] for proving knowledge of a discrete log in a group G of prime order q. Let $g \neq 1$, and let $x = g^w$ be the common input. P is given w as private input. In the language of the above section, the protocol is a proof of knowledge for the relation that consists of pairs ((x, g, G), w) such that $x = g^w$ in G. Then the protocol works as follows:

1. The prover chooses z at random in $[0..q)$, and sends $a = g^z$ to V.
2. The verifier chooses c at random in $[0..q)$, and sends it to P.
3. P sends $r = (z + cw) \bmod q$ to V, and V checks that $g^r = a\, x^c$.

Completeness trivially holds with probability 1. Correct answers to two different c-values give two equations $r_1 = z + wc_1 \bmod q$ and $r_2 = z + wc_2 \bmod q$ so we find that $w = (r_1 - r_2)/(c_1 - c_2) \bmod q$. So special soundness holds also. Finally, note that by choosing c and r at random, we can make a simulated conversation $(g^r x^{-c}, c, r)$ between the honest verifier and prover. Since c can be chosen freely, we even get special honest verifier zero-knowledge.

3. SECRET SHARING

A secret sharing scheme is a method by which a secret s can be distributed among n participants, by giving a *share* to each participant. The shares are computed in such a way that some subsets of participants can, by pooling their shares, reconstruct s. These subsets are called *qualified* sets. Participants forming a non-qualified set should be able to obtain no information whatsoever about s. Such a secret sharing scheme is called *perfect*.

The collection of qualified sets is called the *access structure* for the secret sharing scheme. Clearly if participants in some set can reconstruct s, so can any superset, and therefore in order for the scheme to make sense, it must be the case that if A is a qualified set, then any set containing A is also qualified. An access structure with this property is called *monotone*.

A special case of monotone access structures is structures containing all subsets larger than some threshold value. Such structures are called *threshold structures*.

Any monotone access structure has a natural dual structure. This concept was first defined in [14].

Definition 1 *Let Γ be an access structure containing subsets of a set M. If $A \subset M$, then $\bar{A}$ denotes the complement of A in M. Now Γ^*,* the dual access structure *is defined as follows:*

$$A \in \Gamma^* \Leftrightarrow \bar{A} \notin \Gamma.$$

The next propositions follow directly from the definition.

Proposition 2 *The dual Γ^* of a monotone access structure is monotone as well, and satisfies*

$$(\Gamma^*)^* = \Gamma.$$

Furthermore, if Γ is a threshold structure, then so is Γ^*.

Proposition 3 *Let Γ be monotone. A set is qualified in Γ exactly when it has a non-empty intersection with every qualified set in Γ^*.*

In the next section, we will assume we are given a protocol of the form described in Section 2. For each input length k we will assume we are given a monotone access structure $\Gamma(k)$ on n participants, where $n = n(k)$ is polynomially bounded function of k. Thus we have a *family of access structures* $\{\Gamma(k)|\ k = 1, 2, \ldots\}$. We can then build a new protocol for proving statements on n problem instances provided we have a perfect secret sharing scheme $\mathcal{S}(k)$ for $\Gamma(k)^*$ satisfying certain requirements to be defined below.

Let $D(s)$ denote the joint probability distribution of all shares resulting from distributing the secret s. For any set A of participants, $D_A(s)$ denotes the restriction of $D(s)$ to shares in A. As $\mathcal{S}(k)$ is perfect, $D_A(s)$ is independent from s for any non-qualified set A. So we will write D_A instead of $D_A(s)$, whenever A is non-qualified. The requirements then are:

1. All shares generated in $\mathcal{S}(k)$ have length polynomially related to k.
2. Distribution and reconstruction of a secret can be done in time polynomial in k.
3. Given secret s and a full set of n shares, one can test in time polynomial in k that the shares are all consistent with s, i.e. that all qualified sets of shares determine s as the secret.
4. Given any secret s, a set of shares for participants in a non-qualified set A (distributed according to D_A) can always be completed to a full set of shares distributed according to $D(s)$ and consistent with s. This completion process can be done in time polynomial in k.

5. For any non-qualified set A, the probability distribution D_A is such that shares for the participants in A are independent and uniformly chosen.

Definition 2 *A perfect secret sharing scheme satisfying requirements 1–4 is called* semi-smooth. *If, in addition, requirement 5 is satisfied it is called* smooth.

It is natural to ask if for any family of monotone access structures there is a family of smooth secret sharing schemes. This question is easy to answer in case of threshold structures. In that case it is clear that Shamir's secret sharing scheme [13] can be used. This scheme is even *ideal*, i.e. the shares are of the same length as the secret. In Shamir's scheme, the secret is an element in a finite field $GF(q)$. A secret is shared by choosing a random polynomial $f(X) = \alpha_{d-1}X^{d-1} + \cdots + \alpha_1 X + s$, where s is the secret, n is fixed and different points $p_1, \ldots, p_n$ in $GF(q)$ are chosen, and the i-th share is $f(p_i)$. Given d or more shares, f and therefore s can be found by Lagrange interpolation. With $d-1$ or fewer shares, s is completely unknown.

As an alternative to Shamir's scheme we have the following secret sharing scheme, which is also ideal. Again $s \in GF(q)$ is the secret, but the i-th share now is a number $c_i \in GF(q)$, $1 \le i \le n$, such that $B\boldsymbol{c} = s\boldsymbol{e}_1$. Here, B is a $n-d+1$ by n matrix over $GF(q)$, $\boldsymbol{c} = (c_1, \ldots, c_n)$, and $\boldsymbol{e}_1 = (1, 0, \ldots, 0)$ is a vector of length $n-d+1$. Matrix B should be such that any $n-d+1$ columns are linearly independent (which implies that the rank of B is equal to $n-d+1$). An appropriate choice for B is therefore the first $n-d+1$ rows of a Vandermonde matrix over $GF(q)$, say:

$$B = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{n-d} & \cdots & n^{n-d} \end{pmatrix}.$$

The secret s can be recovered from any d shares as follows. Since $B\boldsymbol{c} = s\boldsymbol{e}_1$, it follows that $s = \sum_{i=1}^{n} c_i$. Furthermore, when d entries of $\boldsymbol{c}$ are known, the remaining $n-d$ entries follow uniquely from the equation $B'\boldsymbol{c} = \mathbf{o}$, where B' is the matrix B with the first row removed and $\mathbf{o}$ denotes a vector of $n-d$ zeros. This is true because B' is a $n-d$ by n matrix for which any $n-d$ columns are linearly independent. In case less than d shares are known, the remaining shares can be chosen such that any secret is matched.

For more general families of access structures, the answer to this question depends on whether the parameter n is a constant, or is allowed to increase polynomially as a function of k.

In case n is a constant, there exists a smooth secret sharing scheme for any monotone access structure. For any minimal qualified set A, we do the following: choose $s_1, \ldots, s_{|A|}$ at random under the condition that $s_1 \oplus \cdots \oplus s_{|A|} = s$, and give one s_i to each participant in A. This scheme was first proposed in [9].

Any qualified set can reconstruct the secret since it must contain a minimal qualified set. By monotonicity, no non-qualified set contains a qualified one, so the secret cannot be reconstructed by a non-qualified set. It is easy to check that all properties above are satisfied by this scheme: the size of shares and the work needed in this scheme is linear in k, but the constant involved depends of course on n and on the access structure. However, the number of possible subsets is exponential in n, so for non-constant n this scheme will not necessarily be smooth.

For non-constant n, it is an open question whether there are secret sharing schemes of the kind we need for any sequence of access structures. Benaloh and Leichter [1] have proposed secret sharing schemes for more general access structures defined by monotone formulae, i.e. Boolean formulae containing only AND and OR operators.

Consider a monotone formula F with n variables. Any subset A of n participants corresponds in a natural way to a set of values of the n variables by assigning a variable to each participant and let each variable be 1 if the corresponding participant is in A and 0 otherwise. We let $F(A)$ be the bit resulting from evaluating F on inputs corresponding to A. Then we can define an access structure Γ_F by

$$A \in \Gamma_F \Leftrightarrow F(A) = 1$$

We let F^* denote the *dual formula*, which results from replacing in F all AND operators by OR's and vice versa. It is not hard to show the following proposition.

PROPOSITION 4 *If F is monotone then Γ_F is also monotone. Conversely, for any monotone access structure Γ, there is a monotone formula F, such that $\Gamma = \Gamma_F$. We have that $(\Gamma_F)^* = \Gamma_{F^*}$.*

In [1], a generic method is given that, based on any monotone formula F, builds a perfect secret sharing scheme for the access structure Γ_F. The formula F may contain general threshold operators, in addition to simple AND and OR operations. For a polynomial size formula, it can be shown that the secret sharing scheme from [1] satisfies all of the above requirements except possibly requirement 5. This leads to:

PROPOSITION 5 *Let $\{\Gamma(k)\}$ be a family of access structures such that $\Gamma(k) = \Gamma_{F_k}$ for a family of polynomial size monotone formula $\{F_k\}$. Then there exists a family of semi-smooth secret sharing schemes for $\{\Gamma(k)\}$.*

A final comment before we go on to the main result is that we will need to distribute secrets of length $t = t(k)$ bits, where t is polynomially bounded in k. This does not impose any restrictions on $\mathcal{S}(k)$ because any secret sharing scheme can distribute secrets of any length by running an appropriate number of copies of the scheme in parallel. We therefore assume that $\mathcal{S}(k)$ always distributes secrets of length t. Note that, if n is constant as a function of k, only one access structure and secret sharing scheme are involved.

4. Main Result

The next theorem describes the construction of a proof of knowledge from a basic proof of knowledge $\mathcal{P}$ for a relation R and a family of secret sharing schemes. In the constructed proof of knowledge both prover and verifier are probabilistic polynomial time machines, using the prover and verifier of $\mathcal{P}$, respectively, as subroutines.

For the statement of the result we need some notation. Let Γ be an access structure on n participants. Then R_Γ is a relation defined by:

$$((x_1, ..., x_m), (w_1, ..., w_m)) \in R_\Gamma$$

iff all x_i's are of the same length, say, k bits, $m = n(k)$, and the set of indices i for which $(x_i, w_i) \in R$ corresponds to a qualified set in $\Gamma(k)$. In a proof of knowledge for relation R_Γ the prover thus proves to know witnesses to a set of the x_i's corresponding to a qualified set in Γ.

Theorem 1 *Let $\mathcal{P}$ be a three round public coin, honest verifier zero-knowledge proof of knowledge for relation R, and assume that $\mathcal{P}$ has the special soundness property. Let $\{\Gamma(k)\}$ be a family of monotone access structures and let $\{\mathcal{S}(k)\}$ be a family of smooth secret sharing schemes such that the access structure of $\mathcal{S}(k)$ is $\Gamma(k)^*$. Then there exists a three round public coin, witness indistinghuisable proof of knowledge for relation R_Γ.*

Proof To improve readability we drop in the following the dependency on k from the notation, and write $\mathcal{S} = \mathcal{S}(k)$, $\Gamma = \Gamma(k)$ and $n = n(k)$. We will distribute secrets of length t in $\mathcal{S}$. If the length of any share resulting from this is larger than t, we will replace $\mathcal{P}$ by a number of parallel executions of $\mathcal{P}$ to make sure that a challenge is at least as long as any share.[2] Note that this does not violate the honest verifier zero-knowledge nor the special soundness property. A basic idea in the following will be to interpret a challenge as a share. If challenges are longer than shares, we will simply take the first appropriate number of bits of the challenge to be the corresponding share. If c is a challenge, $share(c)$ will denote the corresponding share.

The following now describes the new protocol:

1. Let $A \in \Gamma$ be the set of indices i such that P knows a witness for x_i. For each $i \in \overline{A}$, P runs simulator S on input x_i. Let (m_1^i, c_i, m_2^i) be the resulting conversation produced by S. P sends to V m_1^i, $i = 1, \ldots, n$, where m_1^i is the value just produced by S if $i \in \overline{A}$, and otherwise m_1^i is what the prover in $\mathcal{P}$ would send as m_1 given a witness for input x_i.
2. V chooses a t-bit string s at random and sends it to P.
3. Consider the set of shares $\{share(c_i) | i \in \overline{A}\}$ that correspond to the c_i from the simulation in Step 1. As $\overline{A}$ is non-qualified in Γ^*, requirement 4 guarantees that P can complete these shares to a full set of shares consistent

[2] For some secret sharing schemes, there is a lower bound on the length of shares in terms of n. For Shamir's scheme, the length of shares is at least $\log_2(n+1)$. If t is smaller than this bound, we can again replace $\mathcal{P}$ by a number of parallel executions.

with s. P then forms challenges c_i for indices $i \in A$, such that $share(c_i)$ equals the share produced in the completion process. This is done by simply copying the bits of the shares and padding with random bits if necessary. In Step 1, S has produced a final message m_2^i in $\mathcal{P}$ for $i \in \overline{A}$. For $i \in A$, P knows a witness for x_i, and can therefore find a valid m_2^i for m_1^i and c_i by running the prover's algorithm from $\mathcal{P}$. Finally, P sends the set of messages c_i, m_2^i, $i = 1, \ldots, n$ to V.

4. V checks that all conversations (m_1^i, c_i, m_2^i) now produced would lead to acceptance by the verifier in $\mathcal{P}$, and that the shares $share(c_i)$ are consistent with secret s. He accepts if and only if these checks are satisfied.

It is clear from the assumptions on $\mathcal{S}$ that P and V need only poly-time and access to the prover and verifier of $\mathcal{P}$. It therefore remains to be seen that the protocol is a proof of knowledge and that it is witness indistinguishable.

Completeness is trivially seen to hold by inspection of the protocol. For *soundness*, assume that some prover P^* for a given first message $\{m_1^i |\ i = 1, \ldots, n\}$ can answer correctly a non-negligible fraction of the possible choices of s. This means that by rewinding P^*, we can efficiently get correct answers to two different values, say s and s'.[3] Let the shares of s and s' sent in the protocol be $share(c_i)$ and $share(c_i')$, $i = 1, \ldots, n$, respectively. Then for every qualified set $B \in \Gamma^*$, there must be an $i \in B$, such that $share(c_i) \neq share(c_i')$ since otherwise it would follow that $s = s'$. But then we also have that $c_i \neq c_i'$ and so by assumption on $\mathcal{P}$, we can compute a witness for x_i. So P^* knows a witness in every qualified set of Γ^*. On account of Proposition 3 the set of witnesses we thus extract is a qualified set in the access structure Γ.

As for *witness indistinguishability*, we have to show that the distribution of the conversation is independent of which qualified set $A \in \Gamma$ the prover uses. First observe that the distribution of each m_1^i depends only on x_i and equals the distribution of the prover's first message in an execution of $\mathcal{P}$ with x_i as input. This follows from Proposition 1, using that $\mathcal{P}$ is honest verifier zero-knowledge. In particular, the joint distribution of the m_1^i's, and hence the verifier's choice of s, is independent of A.

Since the set $\{share(c_i)\}$ is constructed by completing a set of uniformly distributed shares in a non-qualified set of $\mathcal{S}$, the joint distribution of the $share(c_i)$'s is simply $D(s)$. Since the c_i's are constructed from the shares by possibly padding with random bits, the joint distribution of the c_i's is independent of A. Finally, Proposition 1 implies that the distribution of each m_2^i depends only on x_i, m_1^i and c_i, and is therefore also independent of A. □

If the secret sharing schemes are ideal, the communication complexity of the protocol in Theorem 1 is at most t bits plus n times that of $\mathcal{P}$. Note that instead of taking several instances of the same proof of knowledge, it is also possible to combine different proofs of knowledge. In this way, one may for instance prove knowledge of either a discrete log or an RSA root without revealing which.

[3] There are 2^t possible s-values which is super-polynomial in k, whence any polynomial fraction of these contain at least 2 values for all large enough k.

THEOREM 2 *As Theorem 1, but with $\mathcal{P}$* special *honest verifier zero-knowledge and $\mathcal{S}(k)$* semi-*smooth.*

Proof In this case the protocol from Theorem 1 is changed as follows. In Step 1, the prover uses $\mathcal{S}$ to distribute an arbitrary secret, and discards all shares in A. The remaining shares are distributed according to $D_{\overline{A}}$. He then runs the special simulator on the corresponding challenges. Note that the completion process can still be performed on account of requirement 4, and as before, the honest prover can counter any challenge s by the verifier. Soundness is proven in the same way as before. Therefore, the modified scheme still constitutes a proof of knowledge for relation R_Γ.

As for witness indistinguishability, we only have to note that the distribution of any m_1^i generated by the (special) simulator is the same for any particular challenge value c_i used, because m_1^i in a real execution of $\mathcal{P}$ is independent of the challenge. Therefore the joint distribution of the m_1^i's is the same as in the case of Theorem 1. The rest of the proof is therefore the same as for Theorem 1. □

The witness indistinguishable property of the protocol from Theorem 1 leads us to a generalization of Theorem 4.3 of [4]. To state the result, we need to introduce the concept of an *invulnerable generator G* for a relation R. Such generators were first introduced in [6] and later used in slightly modified form in [4]. Such a generator is a probabilistic polynomial time algorithm which outputs a pair $(x, w) \in R$. The generator is invulnerable if no probabilistic polynomial time enemy given only x can compute an element in $w(x)$ with non-negligible probability, taken over the coin flips of both G and the enemy.

Thus, asserting the existence of an invulnerable generator for a relation is a way of stating that it is feasible to generate hard, solved instances of the underlying computational problem.

For any generator G, we let G^n denote the generator that produces an n-tuple of pairs in R by running G independently n times in parallel. We will also need some notation for access structures: for a monotone access structure Γ, we let the sets in Γ correspond to subsets of the index set $N = \{1, ..., n\}$. Now let the set $I_\Gamma \subset N$ be defined by: $i \in I_\Gamma$ iff i is contained in every qualified set in Γ. It is easy to see by monotonicity of Γ that $i \in I_\Gamma$ precisely if $N \setminus \{i\}$ is not qualified.

THEOREM 3 *Let $\mathcal{P}$ be a witness indistinguishable proof of knowledge for the relation R_Γ, where $\Gamma = \{\Gamma(k)\}$ is a family of monotone access structures on $n(k)$ participants, and R is a binary relation. If for all k, $\Gamma(k)$ contains at least two different minimal qualified sets, and there is an invulnerable generator G for R, then $\mathcal{P}$ is witness hiding over $G^{n(k)}$.*

Proof We follow the line of reasoning from Thm. 4.3 of [4]. Suppose we are given an probabilistic polynomial time enemy $\mathcal{A}$ that has non-negligible probability of computing a witness, using the honest prover in the scheme from Theorem 1 as a subroutine. We show that $\mathcal{A}$ can be compiled into an algorithm that solves with non-negligible probability random instances x generated by G, thus contradicting the invulnerability of the generator (see [4]).

From the assumption on $\Gamma(k) = \Gamma$ (at least two minimal qualified sets) it follows that $N \setminus I_\Gamma$ must contain at least two elements, and that I_Γ is not qualified.

Our compilation now works as follows:

1. Determine the set I_Γ. This can be done by recalling that soundness of $\mathcal{P}$ allows the prover to convince the verifier with only negligible probability if the prover only knows witnesses in a non-qualified set. So for each i, we can use G to generate a set of problem instances and emulate the protocol with $N \setminus \{i\}$ corresponding to the set of known witnesses. If $i \in I_\Gamma$, then this fails almost always, otherwise it fails with negligible probability.

   For simplicity, we argue in the following as if this procedure determines the correct I_Γ with probability 1. Taking into account the small probability of making a mistake introduces only a negligible change in the success probability of our algorithm.
2. Recall that our input is a problem instance x generated by G. We now form an n tuple of instances $(x_1, ..., x_n)$ as follows: choose at random $j \in N \setminus I_\Gamma$ (which is non-empty), and let $x_j = x$. For all other indices i, run G to produce a solved instance x_i and save the witness w_i.
3. Give $x_1, ..., x_n$ as input to $\mathcal{A}$. When $\mathcal{A}$ needs to interact with the prover, we simply simulate the prover's algorithm from Theorem 1. This can be done because we know witnesses of all instances except x_j, and the fact that $j \notin I_\Gamma$ guarantees that $N \setminus \{j\}$ is qualified.
4. If $\mathcal{A}$ is successful, it outputs a witness for the relation R_Γ which by definition is a set of witnesses $\{w_i\}$ corresponding to a qualified set A in Γ. If $j \in A$, we have success and can output w_j. Else output something random.

We now show that this compilation finds a witness for x with non-negligible probability. First note that the joint distribution of the x_i's we give to $\mathcal{A}$ is the same as in an ordinary interaction with the prover. Therefore $\mathcal{A}$ is successful with non-negligible probability. We therefore only have to bound the probability that j is in A, the set of witnesses we get from $\mathcal{A}$. Since I_Γ is not qualified, A must contain at least one index not in I_Γ. By witness indistinguishability, $\mathcal{A}$ has no information about which j in $N \setminus I_\Gamma$ we have chosen, and so the probability that $j \in A$ is at least $1/|N \setminus I_\Gamma|$. Hence if $\mathcal{A}$ has success probability ϵ, we have success probability at least ϵ/n, which is non-negligible. □

Note that an access structure has at least two minimal qualified sets exactly when the corresponding minimal CNF-formula contains at least one OR-operator.

Note also that this result only shows that an enemy cannot compute a complete qualified set of witnesses. It does not rule out that the protocol could help him to compute a small, non-qualified set. Ideally, we would like to prove that the enemy cannot compute even a single witness. With a stronger assumption on the access structure, this can be done:

COROLLARY 1 *Let $\mathcal{P}$ be a witness indistinguishable proof of knowledge for the relation R_Γ, where $\Gamma = \{\Gamma(k)\}$ is a family of monotone access structures on $n(k)$ participants, and R is a binary relation. Suppose that for all k the set $I_{\Gamma(k)}$ is empty. Suppose finally that there is an invulnerable generator G for R, and that inputs for $\mathcal{P}$ are generated by $G^{n(k)}$. Then no probabilistic polynomial time enemy interacting with the honest prover can with non-negligible probability compute a witness for any of the x_i in the input to the protocol.*

Proof Since $I_{\Gamma(k)}$ is non qualified, there are at least two minimal sets, and therefore the proof is the same as for Theorem 3, except that it follows from the assumption that the index j is always chosen among all indices. Hence if the enemy outputs at least one correct witness, there is a non-negligible probability of at least $1/n$ that this is the witness we are looking for. □

A certain special case of Theorem 1 is interesting in its own right:

COROLLARY 2 *Assume we have a proof of knowledge $\mathcal{P}$ for relation R as described in Section 2. Then for any n, d there is a protocol with the same round complexity as $\mathcal{P}$ in which the prover shows that he knows d out of n witnesses without revealing which d witnesses are known.*

Proof Use Theorem 1 with, for example, Shamir's secret sharing scheme for $\mathcal{S}$ and a threshold value of $n - d + 1$. □

COROLLARY 3 *Consider the protocol guaranteed by Corollary 2, let $n = 2$ and $d = 1$, i.e. the prover proves that he knows at least 1 out of 2 solutions. For any generator G generating pairs in R, this protocol is witness hiding over G^2.*

Proof Since protocols constructed from Theorem 1 are always witness indistinguishable, we can use Theorem 4.2 of Feige and Shamir[4]. □

Note that for this corollary, we do not need the assumption that G is invulnerable, as in Theorem 3.

To build the protocol of Corollary 3, we need a 2 out of 2 threshold scheme. Such a scheme can be implemented by choosing random shares c_1, c_2 such that $c_1 \oplus c_2$ equals the secret. Therefore, in the simple case of Corollary 3, the protocol constructed by Theorem 1 simply becomes a game where the verifier chooses a random s, and the prover shows that he can answer correctly a pair of challenges c_1, c_2, such that $s = c_1 \oplus c_2$. In the prover's final message, he only has to send c_1 because the verifier can then compute c_2 himself. Hence the communication complexity of the new protocol is exactly twice that of $\mathcal{P}$, whence the new protocol is just as practical as $\mathcal{P}$. See also the examples below.

COROLLARY 4 *Let $\{\Gamma(k) = \Gamma_{F_k}\}$ be a family of monotone access structure on $n(k)$ participants defined by a polynomial size family of formulas $\{F_k\}$, and*

$$\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier}\\
(x_i = g^{w_i},\ 1\le i\le d) & & \\
z_1,\dots,z_n \in_R \mathbb{Z}_q & & \\
c_{d+1},\dots,c_n \in_R \mathbb{Z}_q & & \\
a_i := \begin{cases} g^{z_i} & ,1\le i\le d \\ g^{z_i}x_i^{-c_i} & ,d{+}1\le i\le n\end{cases} & \xrightarrow{a_1,\dots,a_n} & \\
 & & s\in_R \mathbb{Z}_q\\
 & \xleftarrow{s} & \\
\text{Solve } \alpha_1,\dots,\alpha_{n-d} \text{ s.t. } f(0)=s & & \\
\text{and } f(i)=c_i \text{ for } d{+}1\le i\le n & \xrightarrow{\alpha_1,\dots,\alpha_{n-d}} & \\
r_i := \begin{cases} z_i + f(i)w_i & ,1\le i\le d\\ z_i & ,d{+}1\le i\le n\end{cases} & \xrightarrow{r_1,\dots,r_n} & \\
 & & g^{r_i} \overset{?}{=} a_i x_i^{f(i)}\ ,\\
 & & 1\le i\le n
\end{array}$$

FIGURE 1. Proof of knowledge of d out of n secrets using a polynomial.

let $\mathcal{P}$ be a proof of knowledge with properties as described in Section 2. Suppose $\mathcal{P}$ is special honest verifier zero-knowledge. Then there exists a witness-indistinguishable proof of knowledge in which the prover proves that he knows a subset of solutions to $n(k)$ problem instances that is qualified in $\Gamma(k)$. Let $M(k)$ be the maximal number of occurrences of a variable in $F(k)$. Then the communication complexity of the new protocol is at most $nM(k)$ times that of $\mathcal{P}$ plus t bits.

Proof By Proposition 2, $\Gamma(k)^* = \Gamma_{F_k^*}$, and since the size of F_k^* is the same as that of F_k, we can use the secret sharing scheme guaranteed by Proposition 5 when we do the construction of Theorem 1. The statement on the communication complexity follows from the fact that the shares of the secret sharing scheme constructed in [1] from $F(k)$ have maximal size $tM(k)$ bits, so that we have to use $M(k)$ parallel executions of $\mathcal{P}$ in the construction of Theorem 1. □

4.1. Examples

We present two instances of the general case for threshold structures (cf. Corollary 2), using Schnorr's protocol as the basic proof of knowledge (cf. Section 2.1). As secret sharing schemes we use either Shamir's scheme or the alternative scheme as described in Section 3. The protocols use threshold values of $n-d+1$ to obtain proofs of knowledge for d out of n secrets. Polynomial f in Figure 4.1 is therefore of degree $n-d$:

$$f(X) = s + \alpha_1 X + \cdots + \alpha_{n-d}X^{n-d}.$$

$$
\begin{array}{ccc}
\text{Prover} & & \text{Verifier}\\
(x_i = g^{w_i},\ 1 \le i \le d) & & \\
z_1,\ldots,z_n \in_R \mathbb{Z}_q & & \\
c_{d+1},\ldots,c_n \in_R \mathbb{Z}_q & & \\
a_i := \begin{cases} g^{z_i} & ,1\le i\le d \\ g^{z_i}x_i^{-c_i} & ,d{+}1 \le i \le n\end{cases} & \xrightarrow{a_1,\ldots,a_n} & \\
 & & s \in_R \mathbb{Z}_q \\
 & \xleftarrow{s} & \\
\text{Solve } c_1,\ldots,c_d \text{ s.t. } Bc = se_1 & \xrightarrow{c_1,\ldots,c_{n-d}} & \\
r_i := \begin{cases} z_i + c_i w_i & ,1\le i \le d\\ z_i & ,d{+}1\le i\le n\end{cases} & \xrightarrow{r_1,\ldots,r_n} & \begin{array}{c}\text{Solve } c_{n-d+1},\ldots,c_n\\ \text{s.t. } Bc = se_1\\ g^{r_i} \overset{?}{=} a_i x_i^{c_i},\\ 1\le i\le n\end{array}
\end{array}
$$

FIGURE 2. Proof of knowledge of d out of n secrets using a matrix.

And matrix B in Figure 4.1 is of size d by n:

$$
B = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{d-1} & \cdots & n^{d-1} \end{pmatrix}.
$$

For convenience, we assume that the prover knows the first d witnesses called $w_1, \ldots, w_d$.

Compared to the general description in the proof of Theorem 1, the protocols have been optimized to reduce the communication complexity. That is, in Figure 4.1 the coefficients of f are sent to the verifier rather than $f(i)$, for $i = 1, \ldots, n$. Similarly, in Figure 4.1, only the first $n-d$ entries of $\boldsymbol{c}$ are sent to the verifier rather than all entries of $\boldsymbol{c}$.

It is interesting to compare the number of multiplications required for the computations involving f and B. In Figure 4.1, finding f requires about $(n-d)^2$ multiplications. Furthermore, there are $n+d$ applications of f, requiring $n-d$ multiplications for each application. The grand total is therefore $2n(n-d)$ multiplications. In Figure 4.1, both the prover and verifier compute a vector of length d requiring nd multiplications each. The grand total for this scheme is therefore $2nd$ multiplications.

From this we conclude that Shamir's scheme should be used for $d > n/2$ and that the alternative scheme should be used otherwise.

5. Application to Identification and Signatures

Suppose we have n users, for example employees of a company, such that the i-th user has a public key x_i and secret key $w_i \in w(x_i)$. Suppose also that certain subsets of users are qualified in the sense that they are allowed to initiate certain actions, sign letters on behalf of the company, etc. This defines an access structure on the set of users. Theorem 1 now gives a way in which a subset of users can collaborate to identify themselves as a qualified subset, without revealing anything else about their identities. This makes good sense, if they are to assume responsibility on behalf of the company, rather than personally.

This also extends to digital signatures, since by using a hash function, any three round proof of knowledge as the one produced by Theorem 1 can be turned into a signature scheme by computing the challenge as a hash value of the message to be signed and the prover's first message (this technique was introduced in [5]). By this method, a signature can be computed which will show that a qualified subset was present, without revealing which subset was involved. This may be seen as a generalization of the group signature concept, introduced by Chaum and Van Heyst [2]. One aspect of group signatures which is missing here, however, is that it is not possible later to "open" signatures to discover the identities of users involved.

6. Open Problems

Two obvious open problems remain. First, can Theorem 1 be proved assuming ordinary soundness of $\mathcal{P}$, and not special soundness? We remark that this can be done at the expense of assuming existence of a bit commitment scheme; it is also interesting to note that if one aims at proofs of membership and not proofs of knowledge, special soundness is not needed. A second question is whether Theorem 1 can be generalized to other types of protocols than public coin protocols.

Acknowledgement

We thank Douglas Stinson for helping us with information about results on secret sharing schemes, and Matthew Franklin for useful discussions and comments on the presentation.

References

1. J. Benaloh & J. Leichter (1988). Generalized Secret Sharing and Monotone Functions, *Proc. of Crypto 88*, Springer Verlag LNCS series, 25–35.
2. D. Chaum & E. van Heyst (1991). Group Signatures, *Proc. of EuroCrypt 91*, Springer Verlag LNCS series.
3. I. Damgård (1993). Interactive Hashing can Simplify Zero-Knowledge Protocol Design Without Complexity Assumptions, *Proc. of Crypto 93*, Springer Verlag LNCS series.

4. U. Feige & A. Shamir (1990). Witness Indistinguishable and Witness Hiding Protocols, *Proc. of STOC 90.*
5. U. Feige, A. Fiat, & A. Shamir (1988). Zero-Knowledge Proofs of Identity, Journal of Cryptology 1 (1988) 77–94.
6. M. Abadi, E. Allender, A. Broder, J. Feigenbaum, & L. Hemachandra (1988). On Generating Solved Instances of Computational Problems, *Proc. of Crypto 88*, Springer Verlag LNCS series.
7. S. Goldwasser, S. Micali, & C. Rackoff (1989). The Knowledge Complexity of Interactive Proof Systems, *SIAM Journal on Computing* **18**, 186–208.
8. L. Guillou & J.-J. Quisquater (1988). A Practical Zero-Knowledge Protocol fitted to Security Microprocessor Minimizing both Transmission and Memory, *Proc. of EuroCrypt 88,* Springer Verlag LNCS series.
9. M. Ito, A. Saito, & T. Nishizeki (1987). Secret Sharing Scheme Realizing any Access Structure, *Proc. Glob.Com.*
10. A. De Santis, G. Di Crescenzo, & G. Persiano (1993). Secret Sharing and Perfect Zero-Knowledge, *Proc. of Crypto 93*, Springer Verlag LNCS series.
11. A. De Santis, G. Persiano, M. Yung. Formulae over Random Self-Reducible Languages: The Extended Power of Perfect Zero-Knowledge, manuscript.
12. C.P. Schnorr (1991). Efficient Signature Generation by Smart Cards, *Journal of Cryptology* **4**, 161–174.
13. A. Shamir (1979). How to Share a Secret, *Communications of the ACM* **22**, 612–613.
14. G.J. Simmons, W.A. Jackson, & K. Martin (1991). The Geometry of Shared Secret Schemes, *Bulletin of the Institute of Combinatorics and its Applications* **1**, 71–88.