Efficient Protocols for Set Membership and Range Proofs

Jan Camenisch[1], Rafik Chaabouni[1,2], and abhi shelat[3]

[1] IBM Research
[2] EPFL
[3] U. of Virginia

Abstract. We consider the following problem: Given a commitment to a value σ, prove in zero-knowledge that σ belongs to some discrete set Φ. The set Φ can perhaps be a list of cities or clubs; often Φ can be a numerical range such as $[1, 2^{20}]$. This problem arises in e-cash systems, anonymous credential systems, and various other practical uses of zero-knowledge protocols.

When using commitment schemes relying on RSA-like assumptions, there are solutions to this problem which require only a constant number of RSA-group elements to be exchanged between the prover and verifier [5, 15, 16]. However, for many commitment schemes based on bilinear group assumptions, these techniques do not work, and the best known protocols require $O(k)$ group elements to be exchanged where k is a security parameter.

In this paper, we present two new approaches to building set-membership proofs. The first is based on bilinear group assumptions. When applied to the case where Φ is a range of integers, our protocols require $O(\frac{k}{\log k - \log\log k})$ group elements to be exchanged. Not only is this result asymptotically better, but the constants are small enough to provide significant improvements even for small ranges. Indeed, for a discrete logarithm based setting, our new protocol is an order of magnitude more efficient than previously known ones.

We also discuss alternative implementations of our membership proof based on the strong RSA assumption. Depending on the application, e.g., when Φ is a published set of values such a frequent flyer clubs, cities, or other ad hoc collections, these alternative also outperform prior solutions.

Keywords: Range proofs, set membership proofs, proofs of knowledge, bi-linear maps.

1 Introduction

In this paper we consider zero-knowledge protocols which allow a prover to convince a verifier that a digitally committed value is a member of a given public set. A special case of this problem is when to show that the committed value lies in a specified integer range.

The first problem, which we denote the *set membership* proof, occurs for instance in the context of anonymous credentials. Consider a user who is issued

J. Pieprzyk (Ed.): ASIACRYPT 2008, LNCS 5350, pp. 234–252, 2008.

a credential containing a number of attributes such as address. Further assume the user needs to prove that she lives in a European capital. Thus, we are given a list of all such cities and the user has to show that she possesses a credential containing one of those cities as address (without of course, leaking the city the user lives in). Or, consider a user who has a subscription to a journal (e.g., the news and the sports section). Further assume that some general sections are to all subscribers of a list of sections. Thus, using our protocol, the user can efficiently show that she is a subscriber to one of the required kinds.

The second problem, which we denote the *range proof*, also occurs often in anonymous credential and e-cash scenarios. For example, a user with passport credential might wish to prove that her age is within some range, e.g. greater than 18, or say between 13 and 18 in the case of a teen-community website. This problem is a special case of the set membership proof. Since the elements of the set occur in consecutive order, special techniques can be applied.

1.1 Our Results

Given a set $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$ and a commitment[1] C, a typical approach to the set membership problem is to use a zero-knowledge proof of the form

> "C is a commitment to the element ϕ_1 OR it is a commitment to ϕ_2 OR it is a commitment to $\phi_3 \cdots$ OR it is a commitment to ϕ_n."

Even though there exist efficient algebraic Σ (Sigma) protocols for handling a single such OR clause, such a proof still has length which is proportional to n. One might argue that such proofs necessarily have length proportional to n since the task of describing the set Φ itself requires space n.

However, in many practical situations, the set Φ is often specified in advance by the verifying party. In other words, Φ can be considered a common input to both Prover and Verifier, and thus we might ask whether it is possible to prove a commitment is a commitment to an element of Φ *without* having to explicitly list Φ in the proof.

To the best of our knowledge, we are the first to propose such a scheme for general, unstructured sets. Our approach is incredibly simple. We provide a way to "encode" the set Φ in a way that allows for $O(1)$-sized proofs that a committed element belongs to Φ. Specifically, we let the verifier specify Φ by providing "digital signatures" on the elements of Φ under a new verification key vk. Now if we consider this set of digital signatures as a common input, the proof becomes a statement of the form:

[1] One might wonder what it means to say "the element committed to in C" when the commitment scheme is not a perfectly-binding one. In such a case, technically, the proof is only *computationally sound*—often called an *argument* instead of a proof. In other words, we assume that a computationally-bounded prover knows only one way to open the commitment C and cannot deduce other ways. Indeed, such protocols are technically called *arguments* instead of proofs. Since prior work refers to the problem as a "proof," we continue to use that term.

> "The prover knows a signature under vk for the element committed to in C."

We provide two types of protocols that are instantiations of this idea. The first one is based on a bilinear-group signature scheme which enables an efficient way to make this proof. The second way is based on the Strong RSA assumption and uses the idea of cryptographic accumulators. In both cases, the actual proof of the statement requires $O(1)$ group elements to be exchanged between prover and verifier.

The special case of Range proofs. A popular special case of the set membership problem occurs when the set Φ consists of a range $[a, a+1, a+2, \ldots, b]$—which we denote $[a, b]$. This problem has been well-studied because it occurs so often in practice. Indeed, under the Strong RSA assumption, there are very efficient proofs for this problem as we discuss in the prior work section below. However, in cases when the range is small or the same range is used in many protocol instantiations, our protocol will be more efficient (by a factor of about 8-10, depending on the group employed).

If one is not willing to rely on the Strong RSA assumption, the folklore method to the problem of range proofs is to have the Prover commit to all k bits of his secret, prove that these commitments all encode either a 0 or a 1 and prove that the commitments indeed commit to all the bits of s. The verifier is then convinced that the secret lies in $[0, 2^{k+1} - 1]$ since there were only k commitments. The method can be generalized to any range. The size of such a proof is thus $O(k)$ group elements.

Using the simple idea of the set membership proof, we are able to reduce this size both asymptotically and in practice for many often-occurring ranges. Our simple idea is as follows: Instead of committing to the individual bits of the committed value, we write the secret value in base-u (for some optimally chosen u) and commit to these u-ary digits. If we only provide ℓ such commitments and prove that the secret can be written in u-ary notation, then we implicitly prove that the secret is in the range $[0, u^\ell]$. A generalization of this technique can be used to prove that the secret is in $[a, b]$ for arbitrary integers a and b. The key technique is to use the set-membership protocol in order to prove that each committed digit is indeed a digit in base-u. Writing the secret in base-u (instead of base 2) is indeed an obvious step. However, with prior methods, doing so does not reduce the proof size. With prior methods, proving that a committed digit is a u-ary digit requires a u-wise OR proof of size $O(u)$; since this u-wise OR proof must be done ℓ times independently, prior methods require communication $O(u \cdot \ell)$.

The key insight in our scheme is to design a scheme which can reuse part of one u-ary digit proof in all ℓ proof instances. Specifically, the verifier can send *one* list of u signatures representing u-ary digits, and the prover can use this *same* list to prove that all ℓ digits are indeed u-ary digits. Thus, the total communication complexity of our approach is $O(u + \ell)$. With well-selected values for u and ℓ, we show that this approach yields a proof of size $O(\frac{k}{\log k - \log \log k})$ which is both asymptotically and practically better than the only other known method.

Note that if the range is small or the same range is used for many protocols, then it is more efficient to employ the set membership protocol directly.

1.2 Prior and Related Work

Assume for concreteness the Pedersen commitment scheme over a prime order group. Let g, h be elements of a group G of prime order q. Let $C = g^s h^r$ be the commitment that the prover has sent to the verifier, where s is the secret of which the prover want to show that it lies in a specific range and r is a randomly chosen element from $\mathbb{Z}_q$.

There are a number of known ways that a prover can convince a verifier that the secret committed in C lies in a given range assuming the hardness of the Strong (or sometimes called flexible) RSA problem. Let us review them here.

The most frequent method used in practice is the following. First, the verifier picks a safe prime product $\mathfrak{n} = (2\mathfrak{p}+1)(2\mathfrak{q}+1)$ and two random quadratic residues $\mathfrak{g}, \mathfrak{h}$ modulo $\mathfrak{n}$, and proves to the prover that $\mathfrak{g} \in \langle \mathfrak{h} \rangle$ is true. Next, the verifier computes $\mathfrak{c} = \mathfrak{g}^s \mathfrak{h}^{r'} \bmod \mathfrak{n}$, sends this value to the prover and then runs the following protocol with him:

$$PK\{(s, r, r') : \mathfrak{c} = \mathfrak{g}^s \mathfrak{h}^{r'} \pmod{\mathfrak{n}} \ \wedge \ C = g^s h^r \ \wedge \ s \in [-A, A]\}$$

The protocol is basically a generalized Schnorr proof (in a group of unknown order), where the verifier in addition to accepting the basic proof also verifies whether the answer corresponding to the secret s lies in $[-A/2, A/2]$. If it does so, then the verifier can conclude that the secret must lie in the range $[-A, A]$ (this becomes apparent when one considers the knowledge extractor for the protocol). The drawback of this proof is that it in fact works only if the secret lies in the smaller range $[-A2^{-(k'+k'')}, A2^{-(k'+k'')}]$, with k' being the number of bits of the challenge sent by the verifier and k'' determining the statistical zero-knowledge property, i.e., the secret must be $k' + k''$ bits smaller. Therefore the protocol cannot be used for situations where one has to show that a secret lies exactly in a given range.

Boudot [5] provided an efficient proof that did not have this drawback. He used the observation that any positive number can be composed as the sum of four squares. Thus, to show that a secret s lies in $[A, B]$, one just needs to show that the values $s_1 = s - A$ and $s_2 = B - s$ are positive. So basically, what the prover has to do is to give commitments to s_1 and s_2 and to the numbers $s_{(1,1)}, \ldots, s_{(1,4)}$ and $s_{(2,1)}, \ldots, s_{(2,4)}$, the sum of whose squares are equal to s_1 and s_2 respectively. Of course, if these commitments were, e.g., Pedersen commitments in a group of prime order q, them all we could conclude is that s_1 and s_2 are the sum of four square modulo q, which is not very helpful. Luckily, Okamoto and Fujisaki [13] have shown that when the commitments and the proof is done in a group where the order is not known to the prover, then these relations hold over the integers and thus one can really assert that s_1 and s_2 are positive.

Thus, we get the following protocol: First the prover computes the following commitments $\mathfrak{c}_{(i,j)} = \mathfrak{g}^{s_{(i,j)}} \mathfrak{h}^{r_{(i,j)}} \bmod \mathfrak{n}$ for some randomly chosen $r_{(i,j)}$, sends these to the verifier and then engages in the following proof with him :

$$\begin{aligned}
PK\{(s, r, r', s_1^{(1)}, \ldots, s_1^{(4)}, s_2^{(1)}, \ldots, s_2^{(4)}, r'', r^*) :& \\
\mathfrak{c}_{(1,1)} = \mathfrak{g}^{s_{(1,1)}}\mathfrak{h}^{r_{(1,1)}} \wedge \ldots \wedge \mathfrak{c}_{(1,4)} = \mathfrak{g}^{s_{(1,4)}}\mathfrak{h}^{r_{(1,4)}} \wedge& \\
\mathfrak{c}_{(2,1)} = \mathfrak{g}^{s_{(2,1)}}\mathfrak{h}^{r_{(2,1)}} \wedge \ldots \wedge \mathfrak{c}_{(2,4)} = \mathfrak{g}^{s_{(2,4)}}\mathfrak{h}^{r_{(2,4)}} \wedge& \\
\mathfrak{c}/\mathfrak{g}^A = {\mathfrak{c}_{(1,1)}}^{s_{(1,1)}} \cdots {\mathfrak{c}_{(1,4)}}^{s_{(1,4)}}\mathfrak{h}^{r''} \wedge\ \mathfrak{g}^B/\mathfrak{c} = {\mathfrak{c}_{(2,1)}}^{s_{(2,1)}} \cdots {\mathfrak{c}_{(2,4)}}^{s_{(2,4)}}\mathfrak{h}^{r^*} \wedge& \\
\mathfrak{c} = \mathfrak{g}^s\mathfrak{h}^{r'} \pmod{\mathfrak{n}} \ \wedge\ C = g^s h^r\}&
\end{aligned}$$

We see that this protocol requires the prover to compute 22 modular exponentiations (including the computations of the commitments) and the verifier to compute 12 modular exponentiations. The communication complexity is in about 35 group elements. Groth [15] optimizes this protocol by exploiting the fact that special integers can be written as the sum of 3 squares instead of 4 squares. The major drawback of these approaches is that the Rabin and Shallit algorithm typically used to find the 4 (or 3) squares which sum to the secret takes time $O(k^4)$ where k is the size of the interval. Lipmaa [16] provides another algorithm to find this squares that improves somewhat on the Rabin-Shallit one. However, in practice, these algorithms running times quickly make this approach preventive.

Independently to our work, Teranishi and Sako [20] presented a k-Times Anonymous Authentication in which they present a range proof using Boneh-Boyen signature scheme [4], that can be obtained from our generalized set membership. However their range proof does not compete with ours as our verifier publishes significantly less signatures.

Schoenmakers [18, 19] studied and discussed several recursive relations which can be used to reduce the number of basic Schnorr proofs when committing to the individual bits of the secret. In particular, he writes the upper bound L of the positive range $[0, L)$ as either the product or the sum of two numbers. By doing this scheme recursively he decreased the amount of work needed. However the overall communication load in his protocols is still $O(k)$, where $2^{k-1} < L \leqslant 2^k$. We note that some of his techniques for reducing certain ranges to other more convenient ranges can be used with any range proof technique.

Micali, Kilian, and Rabin [17] considered a more general problem in which a polynomial-time prover wants to commit to a finite set Φ of strings so that, later on, he can, for any string x, reveal with a proof whether $x \in \Phi$ or $x \notin \Phi$ without leaking any knowledge beyond the membership assertions. In particular, the proofs do not even reveal the size of Φ—much less the actual elements. Thus, these protocols are not directly comparable to ours.

1.3 Organization

In section 2, we recall zero-knowledge proofs, Σ-protocols and define proofs of set membership and range proofs. In section 3, we describe our new signature-based set membership together with its corresponding proof. In section 4, we explain how to apply our new signature-based set membership for efficient range proof. We also emphasis on the communication complexity and show how our new range proof is asymptotically better. To have a better insight of our state of the art, we provide a concrete example together with some comparison of previous work. In section 5, we recall cryptographic accumulators together with their proofs, and we describe our new accumulator-based set membership.

2 Definitions

Zero-knowledge proofs and Σ-protocols. We use definitions from [2, 11]. A pair of interacting algorithms (P, V) is a proof of knowledge (PK) for a relation $R = \{(\alpha, \beta)\} \subseteq \{0,1\}^* \times \{0,1\}^*$ with knowledge error $\kappa \in [0,1]$ if (1) for all $(\alpha, \beta) \in R$, $\mathsf{V}(\alpha)$ accepts a conversation with $\mathsf{P}(\beta)$ with probability 1; and (2) there exists an expected polynomial-time algorithm E, called the *knowledge extractor*, such that if a cheating prover P^* has probability ϵ of convincing V to accept α, then E, when given rewindable black-box access to P^*, outputs a witness β for α with probability $\epsilon - \kappa$.

A proof system (P, V) is *honest-verifier zero-knowledge* if there exists a p.p.t. algorithm Sim, called the *simulator*, such that for any $(\alpha, \beta) \in R$, the outputs of $V(\alpha)$ after interacting with $\mathsf{P}(\beta)$ and that of $\mathsf{Sim}(\alpha)$ are computationally indistinguishable.

Note that standard techniques can be used to transform an honest-verifier zero-knowledge proof system into a general zero-knowledge one [11]. This is especially true of special Σ-protocols that will be presented later in the paper. Thus, for the remainder of the paper, our proofs will be honest-verifier zero-knowledge. (This also allows us to make more accurate comparisons with the other proof techniques since they are usually also presented as honest-verifier protocols).

A Σ-protocol is a proof system (P, V) where the conversation is of the form (a, c, z), where a and z are computed by P, and c is a challenge chosen at random by V. The verifier accepts if $\phi(\alpha, a, c, z) = 1$ for some efficiently computable predicate ϕ. Given two accepting conversations (a, c, z) and (a, c', z') for $c \neq c'$, one can efficiently compute a witness β. Moreover, there exists a polynomial-time simulator Sim that on input α and a random string c outputs an accepting conversation (a, c, z) for α that is perfectly indistinguishable from a real conversation between $\mathsf{P}(\beta)$ and $\mathsf{V}(\alpha)$.

We use notation introduced by Camenisch and Stadler [9] for the various zero-knowledge proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. For instance,

$$PK\{(\alpha, \beta, \gamma) : y = g^{\alpha} h^{\beta} \ \wedge \ \mathfrak{y} = \mathfrak{g}^{\alpha} \mathfrak{h}^{\gamma} \ \wedge \ (u \leq \alpha \leq v)\}$$

denotes a "*zero-knowledge* P*roof of* K*nowledge of integers α, β, and γ such that $y = g^{\alpha} h^{\beta}$ and $\mathfrak{y} = \mathfrak{g}^{\alpha} \mathfrak{h}^{\gamma}$ holds, where $v \leq \alpha \leq u$*," where $y, g, h, \mathfrak{y}, \mathfrak{g}$, and $\mathfrak{h}$ are elements of some groups $G = \langle g \rangle = \langle h \rangle$ and $\mathfrak{G} = \langle \mathfrak{g} \rangle = \langle \mathfrak{h} \rangle$. The convention is Greek letters denote quantities the knowledge of which is being proved, while all other parameters are known to the verifier. Using this notation, a proof-protocol can be described by just pointing out its aim while hiding all details. We note that all of the protocols we present in this notation can be easily instantiated as Σ-protocols.

Definition 1 (Proof of Set Membership). *Let $C = (\text{Gen}, \text{Com}, \text{Open})$ be the generation, the commit and the open algorithm of a string commitment scheme. For an instance c, a proof of set membership with respect to commitment scheme C and set Φ is a proof of knowledge for the following statement:*

$$PK\{(\sigma, \rho) \,:\, c \leftarrow \text{Com}(\sigma;\ \rho) \wedge \sigma \in \Phi\}$$

Remark: The proof system is defined with respect to *any* commitment scheme. Thus, in particular, if Com is a perfectly-hiding scheme, then the language Γ_S consists of all commitments (assuming that S is non-empty). Thus for soundness, it is important that the protocol is a proof of knowledge.

Definition 2 (Range Proof). *A range proof with respect to a commitment scheme C is a special case of a proof of set membership in which the set Φ is a continuous sequence of integers $\Phi = [a, b]$ for $a, b \in \mathbb{N}$.*

3 Signature-Based Set Membership

Here we present a new set membership protocol that is inspired by the oblivious transfer protocol presented by Camenisch, Neven, and shelat [8]. The basic idea is that the verifier first sends the prover a signature of every element in the set Φ. Thus, the prover receives a signature on the particular element σ to which C is a commitment. The prover then "blinds" this received signature and performs a proof of knowledge that she possesses a signature on the committed element. Notice that the communication complexity of this proof depends on the cardinality of Φ—in particular because the verifier's first message contains a signature of every element in Φ. The rest of the protocol, however, requires only a constant number of group elements to be sent. The novelty of this approach is that the first verifier message can be re-used in other proofs of membership; indeed, we use this property to achieve our results for range proofs.

Computational Assumptions. Our protocols in this section require bilinear groups and associated hardness assumptions. Let PG be a pairing group generator that on input 1^k outputs descriptions of multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_\mathrm{T}$ of prime order p where $|p| = k$. Let $\mathbb{G}_1^* = \mathbb{G}_1 \setminus \{1\}$ and let $g \in \mathbb{G}_1^*$. The generated groups are such that there exists an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_\mathrm{T}$, meaning that (1) for all $a, b \in \mathbb{Z}_p$ it holds that $e(g^a, g^b) = e(g, g)^{ab}$; (2) $e(g, g) \neq 1$; and (3) the bilinear map is efficiently computable.

Definition 3 (Strong Diffie-Hellman Assumption [4]). *We say that the q-SDH assumption associated to a pairing generator PG holds if for all p.p.t. adversaries A, the probability that $A(g, g^x, \ldots, g^{x^q})$ where $(\mathbb{G}_1, \mathbb{G}_\mathrm{T}) \leftarrow \mathsf{PG}(1^k)$, $g \leftarrow \mathbb{G}_1^*$ and $x \leftarrow \mathbb{Z}_p$, outputs a pair $(c, g^{1/(x+c)})$ where $c \in \mathbb{Z}_p$ is negligible in k.*

A recent work by Cheon [10] shows a "weakness" in the q-SDH assumption. However, this "weakness" is not so relevant when q is a very small number like 50 as it is in our paper.

Boneh-Boyen Signatures. Our scheme relies on the elegant Boneh-Boyen short signature scheme [4] which we briefly summarize. The signer's secret key is $x \leftarrow \mathbb{Z}_p$, the corresponding public key is $y = g^x$. The signature on a message m is $\sigma \leftarrow g^{1/(x+m)}$; verification is done by checking that $e(\sigma, y \cdot g^m) = e(g, g)$. This scheme is similar to the Dodis and Yampolskiy verifiable random function [12].

Security under weak chosen-message attack is defined through the following game. The adversary begins by outputting ℓ messages $m_1, \ldots, m_\ell$. The challenger generates a fresh key pair and gives the public key to the adversary, together with signatures $\sigma_1, \ldots, \sigma_\ell$ on $m_1, \ldots, m_\ell$. The adversary wins if it succeeds in outputting a valid signature σ on a message $m \notin \{m_1, \ldots, m_\ell\}$. The scheme is said to be unforgeable under a chosen-message attack if no p.p.t. adversary A has non-negligible probability of winning this game. Our scheme relies on the following property of the Boneh-Boyen short signature [4] which we paraphrase below:

Lemma 1 ([4](Lemma 3.2)). *Suppose the q-Strong Diffie Hellman assumption holds in $(\mathbb{G}_1, \mathbb{G}_\mathrm{T})$. Then the basic Boneh-Boyen signature scheme is q-secure against an existential forgery under a weak chosen message attack.*

A Note on Protocol Clarity. In order to make our protocols more readable in this version, we do not specifically mention standard checks such as verifying that a received number is a prime, verifying that an element is a proper generator and in the correct group, and, specifically related to our protocols, whether all of the received verifier values are signatures, etc. Again, many of these checks only apply when compiling from honest-verifier zero-knowledge to full zero-knowledge; as we mentioned above, we only consider the honest case.

Common Input: g, h, a commitment C, and a set Φ
Prover Input: σ, r such that $C = g^\sigma h^r$ and $\sigma \in \Phi$.

$P \xleftarrow{y,\{A_i\}} V$ Verifier picks $x \in_R \mathbb{Z}_p$ and
sends $y \leftarrow g^x$ and $A_i \leftarrow g^{\frac{1}{x+i}}$ for every $i \in \Phi$.

$P \xrightarrow{V} V$ Prover picks $v \in_R \mathbb{Z}_p$ and sends $V \leftarrow A_\sigma^v$.

Prover and Verifier run $\mathrm{PK}\{(\sigma, r, v) : C = g^\sigma h^r \ \wedge \ V = g^{\frac{v}{x+\sigma}}\}$

$P \xrightarrow{a,D} V$ Prover picks $s, t, m \in_R \mathbb{Z}_p$ and
sends $a \leftarrow e(V, g)^{-s} e(g, g)^t$ and $D \leftarrow g^s h^m$.

$P \xleftarrow{c} V$ Verifier sends a random challenge $c \in_R \mathbb{Z}_p$.

$P \xrightarrow{z_\sigma, z_v, z_r} V$ Prover sends $z_\sigma \leftarrow s - \sigma c$, $z_v \leftarrow t - vc$, and $z_r \leftarrow m - rc$.
Verifier checks that $D \overset{?}{=} C^c h^{z_r} g^{z_\sigma}$ and
that $a \overset{?}{=} e(V, y)^c \cdot e(V, g)^{-z_\sigma} \cdot e(g, g)^{z_v}$

Fig. 1. Set membership protocol for set Φ

Theorem 1. *If the $|\Phi|$-Strong Diffie-Hellman assumption associated with a pairing generator* PG *holds, then protocol in Fig. 1 is a zero-knowledge argument of set membership for a set Φ.*

Proof. The completeness of the protocol follows by inspection. The soundness follows from the extraction property of the proof of knowledge and the unforgeability of the random function. In particular, the extraction property implies that for any prover P^* that convinces V with probability ϵ, there exists an extractor which interacts with P^* and outputs a witness (σ, r, v) with probability $poly(\epsilon)$. Moreover, if we assume that the extractor input consists of two transcripts, i.e.,

$$\{y, \{A_i\}, V, a, D, c, c', z_\sigma, z'_\sigma, z_v, z'_v, z_r, z'_r\},$$

the witness can be obtained by computing:

$$\sigma = \frac{z_\sigma - z'_\sigma}{c' - c}; \quad r = \frac{z_r - z'_r}{c' - c}; \quad v = \frac{z_v - z'_v}{c' - c}$$

The extractor succeeds when $(c' - c)$ is invertible in $\mathbb{Z}_p$. If $\sigma \notin \Phi$, then P^* can be (almost) directly be used to mount a weak chosen-message attack against the Boneh-Boyen signature scheme with probability $poly(\epsilon)$ of succeeding. Thus, ϵ must be negligible.

Finally, to prove honest-verifier zero-knowledge, we construct a simulator Sim that will simulate all interactions with any honest verifier V^*, see Fig. 2.

1. Sim retrieves $y, \{A_i\}$ from V^*.
2. Sim chooses $\sigma \in_R \Phi$, $v \in_R \mathbb{Z}_p$ and sends $V \leftarrow A_\sigma^v$ to V^*.
3. Sim chooses $s, t, m \in_R \mathbb{Z}_p$ and sends $a \leftarrow e(V, g)^{-s} e(g, g)^t$ and $D \leftarrow g^s h^m$ to V^*.
4. Sim receives c from V^*
5. Finally Sim computes and sends $z_\sigma \leftarrow s - \sigma c, z_v \leftarrow t - vc$, and $z_r \leftarrow m - rc$ to V^*.

Fig. 2. Simulator for the set membership protocol

Since $\mathbb{G}_1$ is a prime-order group, then the blinding is perfect in the first two steps; thus the zero-knowledge property follows from the zero-knowledge property of the Σ-protocol (Steps 3 to 5).

4 Range Proofs

We now turn our attention to the range proofs.

First note that the protocol for set membership can be directly applied to the problem of range proofs. This will not be efficient for ranges spanning more than a few hundred elements. However, if the particular range is fixed over many protocols as it might often be (as is for instance the case when one needs to prove that one is between 13 and 18 years old), then the verifier can publish the signatures once and for all. Thus, the proofs become just the second phase which amounts to one pairing and two exponentiation for the prover and the verifier. This will be about a factor of 8-10 times more efficient than employing Boudot's method.

For the remainder assume, however, that the range is large or that the cost of publishing/sending the signatures on the set elements cannot be amortized.

Instead, our approach is to write the secret σ in u-ary notation, i.e., $\sigma = \sum_j^\ell \sigma_j \cdot u^j$. We may now easily prove that $\sigma \in [0, u^\ell)$ by simply providing (and proving) commitments to the u-ary digits of σ. This problem, however, can be solved by repeating the basic set-membership protocol from above on the set $[0, u-1]$. Moreover, the first verifier message, which requires the most communication, can be re-used for each of the ℓ digits. Assuming that $\sigma \in [0, B)$, the goal is thus to minimize the communication load under the constraint $u^\ell \geqslant B$.

4.1 Range Proofs From Our Signature-Based Set-Membership Protocol

We first present how to prove that our secret σ lies in $[0, u^\ell)$ (see Figure 3). Write σ in the base u to obtain ℓ elements as such: $\sigma = \sum_j \left(\sigma_j u^j\right)$.

Common Input: g, h, u, ℓ, and a commitment C

Prover Input: σ, r such that $C = g^\sigma h^r$ and $\sigma \in [0, u^\ell)$.

$P \xleftarrow{y,\{A_i\}} V$ Verifier picks $x \in_R \mathbb{Z}_p$ and sends $y \leftarrow g^x$ and $A_i \leftarrow g^{\frac{1}{x+i}}$ for every $i \in \mathbb{Z}_u$.

$P \xrightarrow{\{V_j\}} V$ Prover picks $v_j \in_R \mathbb{Z}_p$ and sends $V_j \leftarrow A_{\sigma_j}^{v_j}$ for every $j \in \mathbb{Z}_l$, s.t. $\sigma = \sum_j \left(\sigma_j u^j\right)$

Prover and Verifier run $\mathrm{PK}\{(\sigma_j, r, v_j) : C = h^r \prod_j (g^{u^j})^{\sigma_j} \;\wedge\; V_j = g^{\frac{v_j}{x+\sigma_j}}\}$

$P \xrightarrow{\{a_j\},D} V$ Prover picks $s_j, t_j, m_j \in_R \mathbb{Z}_p$ for every $j \in \mathbb{Z}_l$ and sends $a_j \leftarrow e(V_j, g)^{-s_j} e(g,g)^{t_j}$ and $D \leftarrow \prod_j \left(g^{u^j s_j}\right) h^{m_j}$.

$P \xleftarrow{c} V$ Verifier sends a random challenge $c \in_R \mathbb{Z}_p$.

$P \xrightarrow{\{z_{\sigma_j}\},\{z_{v_j}\},z_r} V$ Prover sends $z_{\sigma_j} \leftarrow s_j - \sigma_j c, z_{v_j} \leftarrow t_j - v_j c$ for every $j \in \mathbb{Z}_\ell$, and $z_r = m - rc$.

Verifier checks that $D \overset{?}{=} C^c h^{z_r} \prod_j \left(g^{u^j z_{\sigma_j}}\right)$ and that $a_j \overset{?}{=} e(V_j, y)^c \cdot e(V_j, g)^{-z_{\sigma_j}} \cdot e(g,g)^{z_{v_j}}$ for every $j \in \mathbb{Z}_l$

Fig. 3. Range proof protocol for range $[0, u^\ell)$

Lemma 2. *If the* $(\log k)$*-Strong Diffie Hellman assumption associated to a pairing generator* $\mathsf{PG}(1^k)$ *holds, there exists a zero-knowledge range argument for the range* $[0, u^\ell)$ *where* $u^\ell < \{0,1\}^{k-1}$.

Proof. (Sketch)

Completeness follows from inspection. As before, the soundness follows from the unforgeability of the Boneh-Boyen signature and the extraction property of the proof of knowledge protocol. The honest-verifier zero-knowledge property follows from the perfect blinding of the signatures in the first phase, and the corresponding honest-verifier zero-knowledge property of the Σ-protocol.

Remark: The prover will have to compute 5ℓ exponentiations.

4.2 Communication Complexity

The first message consisting of u signatures and a verification key sent by the verifier to the prover, is not counted as part of the protocol $((u+1)\cdot|\mathbb{G}_1|)$. The prover then sends ℓ blinded values back. Thus, the first phase requires $Init_l(u,\ell) = \ell \cdot |\mathbb{G}_1|$ communication. The second phase of the protocol involves a proof of knowledge. The prover sends $\ell+1$ first-messages of a Σ-protocol. The verifier sends a single challenge, and the prover responds with $2\ell+1$ elements. Thus the overall communication load according to the parameters u and ℓ is:

$$Com(u,\ell) = \ell \cdot (|\mathbb{G}_1| + |\mathbb{G}_T| + 2\cdot|\mathbb{Z}_p|) + (|\mathbb{G}_1| + 2\cdot|\mathbb{Z}_p|) \tag{1}$$

Finding the optimal u and ℓ thus involves solving

$$\min\ c_1 u + c_2 \ell + c_3 \ \text{ s.t. } u^\ell \geqslant B$$

Notice that the bit-committing protocol corresponds to a setting where $u=2$ and $\ell = k$ which leads to a total communication complexity $O(k)$. Since our protocol allows us to choose more suitable u, we first show that the asymptotic complexity of our approach is smaller than the prior protocols.

Asymptotic Analysis. For the asymptotic analysis, we may ignore the constants c_1, c_2 and c_3. Moreover, we can take $B \approx p/2$ as this is sufficient for showing that a committed value is "positive," i.e., in the range $[0,(p-1/2)]$. Since $p/2 \approx 2^k$, the constraint becomes $u^\ell \geqslant 2^{k-1}$.

By taking logs and dividing, we have that $\ell \approx \frac{k}{\log u}$. Setting $u = \frac{k}{\log k}$ then we get that

$$u = O\left(\frac{k}{\log k}\right), \quad \ell = O\left(\frac{k}{\log k - \log\log k}\right)$$

resulting in a total communication complexity of

$$Com(u,\ell) = O\left(\frac{k}{\log k - \log\log k}\right)$$

which is asymptotically smaller than $O(k)$.

Concrete Optimization. Not only is our solution asymptotically better, but it also performs well for realistic concrete parameters. In order to perform the optimization for concrete parameters we substitute the constraint that $u^\ell \approx B$ into the equation $u + \ell$ above. To minimize, we set the derivative with respect to u to 0 and attempt to solve the equation:

$$c_1 - \frac{c_2 \log B}{u \log^2 u} = 0$$

which simplifies to

$$u \log^2 u = \frac{c_2 \log B}{c_1}. \tag{2}$$

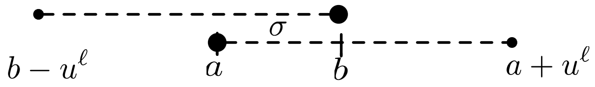
where $\frac{c_2}{c_1} \approx 10$ when standard bilinear groups are used [14]. This equation cannot be solved analytically. However, given B, c_1 and c_2, we can use numerical methods to find a good u as described in [3].

4.3 Handling Arbitrary Ranges $[a, b]$

The above protocol works for the range $[0, u^\ell)$. In order to handle an arbitrary range $[a, b]$, we use an improvement of a folklore reduction described by Schoenmakers in [18] and [19]. Suppose that $u^{\ell-1} < b < u^\ell$. To show the $\sigma \in [a, b]$, it suffices to show that

$$\sigma \in [a, a + u^\ell] \text{ AND } \sigma \in [b - u^\ell, b]$$

Proving that our secret lies in both subsets can be derived from our general



proof that $\sigma \in [0, u^\ell)$ as illustrated in the figure:

$$\sigma \in [b - u^\ell, b) \iff \sigma - b + u^\ell \in [0, u^\ell)$$
$$\sigma \in [a, a + u^\ell) \iff \sigma - a \in [0, u^\ell).$$

Note that the u signatures and the verification key need to be sent only once for both subsets. Since both a, b are public, the only modification necessary is the verifier's check, which should now be:

$$D \stackrel{?}{=} C^c g^{-B+u^\ell} h^{z_r} \prod_j \left(g^{z_{\sigma_j}}\right), \quad D \stackrel{?}{=} C^c g^{-A} h^{z_r} \prod_j \left(g^{z_{\sigma_j}}\right).$$

Thus, essentially 3ℓ extra elements are sent in the protocol, and the prover will have to compute in overall 7ℓ exponentiations.

This scheme can be further optimized when $A+u^{\ell-1} < B$ with an OR-composition. Indeed, the decomposition becomes:

$$[A, B) = [B - u^{\ell-1}, B) \cup [A, A + u^{\ell-1}).$$

The needed modifications are similar to the previous case; the efficiency arises from the fact that we are now working with $\mathbb{Z}_{\ell-1}$. The length of the range set can also be optimized. Indeed if $B - A = u^{\ell}$ then the proof reduces to proving that $\sigma - A \in [0, u^{\ell})$.

Combining this analysis with Lemma 2 yields the following theorem.

Theorem 2. *If the* $\log k$*-Strong Diffie Hellman assumption associated to a pairing generator* $\mathsf{PG}(1^k)$ *holds, there exists a zero-knowledge range argument for the range* $[a, b]$ *where* $0 < a < b < \{0, 1\}^{k-1}$ *whose communication complexity is* $O(\frac{k}{\log k - \log \log k})$.

4.4 Concrete Example and Discussion

Let us discuss our protocol and compare it with other available solutions. The bottom line is the performance of the different methods depend on the application at hand as well as for the assumptions one is willing to make. Assume for a while, all assumptions are fine. Then, for very small intervals (a couple of bits), the standard bit-by-bit method and Schoenmaker's method will probably be the most efficient one. For very large intervals, the method by Boudot will probably be the one of choice as it is mostly independent of the size of the interval. More precisely, it is independent for the verifier but not for the prover as the prover needs to run the Rabin-Shallit algorithm to represent numbers as the sum of four squares and this algorithm has complexity $\mathcal{O}(n^4)$ where n is the bit-length of the number to be decomposed.

Having said that, our methods will typically be the most efficient one when the signatures can be made part of the system parameters, which is probably the case in many scenarios. Of course, at some point it will no longer possible to publish signature of all elements in the range and thats where one will have to restrict these signatures and employ the protocol in this section. When this becomes necessary, one will in practice to make a choice whether it is more efficient to use our algorithm or Boudot's one, the other two will definitely be less efficient.

If one is not restricted by the assumptions one is willing to make, the case is not so clear cut. Let us give a concrete example to shed some light on this. If we pick $B = 599644800$ (which will represent people born before 1989, with their birth date encoded using the Unix Epoch system), we can find the optimal values of u and ℓ by either computing them numerically or by following [3]. Both methods will lead us to $u = 57$ and $\ell = 5$, which minimize the overall communication load:

$$Com_l(57, 5) = 6 \cdot |\mathbb{G}_1| + 5 \cdot |\mathbb{G}_T| + 12 \cdot |\mathbb{Z}_p| \quad (3)$$

Let us illustrate this optimization case with a concrete example. We will assume that an airline company wants to provide special offers to its young clients from a third party. However the exact age of clients should not be divulged to the third party. This offer targets those who are born between 1981 and 1989 (not included). Following the previous example, the birth date will be a secret number between $[347184000, 599644800)$. Here the best option will be to use the OR-composition as $A + u^{\ell-1} < B$ (we know from the previous example that $u = 57$ and $\ell = 5$). Using parameters from Galbraith, Paterson, and Smart [14], we estimate that the size of $\mathbb{G}_1$ is 256 bits, the size of $\mathbb{G}_T$ is 3072 bits and the size of $\mathbb{Z}_p$ is upper-bounded by 256 bits. This leads to an overall communication load of:

$$Com_l(u = 57, \ell = 5) = \ell \cdot |\mathbb{G}_1| + (2\ell - 2) \cdot |\mathbb{G}_T| + 4\ell \cdot |\mathbb{Z}_p| = 30976 \; bits \quad (4)$$

In order to have a better appreciation of this result, let us compare it to previous protocols:

Scheme	*Communication Complexity*
Our new range proof	30976 bits
Boudot's method	48946 bits
Standard bit-by-bit method	96768 bits
Schoenmaker's method	50176 bits

Fig. 4. Communication load comparison for range proof $[347184000, 599644800)$

Let us also discuss the computational complexities. For the verifier, the figure are about similar to the communication complexities as basically the verifier needs to do some computation with the elements received. For the prover it is about the same with the exception that for Boudot's method where the prover needs to run the Rabin-Shallit algorithms. Experiments show that the later algorithm dominates by far the other operations the prover needs to do.

Now, when one does not want to resort to the (strong) RSA assumption, our methods is the only one that provides an efficient proof except when the interval is only a couple of bits.

5 Alternative Set Membership Proofs

The protocol in the previous section employed a set-membership proof as a building block. The set-membership proof protocol we presented in Section 3 has the verifier to produce signatures on the set elements, send them to the prover and then has the prover to show that he knows a signature (by the verifier) and the element he holds. Concretely, we employed the weak signature scheme by Boneh and Boyen in that section. We now discuss alternative solutions to the set membership protocol, i.e., essentially so that the whole protocol could be based on different assumptions. Due to space restriction we do not give all the details here but only in the full version of this paper. However, the solution presented previously is the most efficient one, the alternatives discussed in this section are of similar efficiency.

5.1 Using Alternative Signature Schemes

The protocol that we presented in Section 3 required the prover to be able to prove the knowledge of a signature on a value that he has committed to, where we used Pedersen commitment scheme. Apart for the weak Boneh-Boyen signature scheme, there are other signature schemes that could be employed. In terms of assumptions, one notable alternative would be the one by Camenisch and Lysyanskaya [7] that is based on the strong RSA assumption. It is not hard to adapt the protocol given in Section 3 to that signature scheme, in particular, as Camenisch and Lysyanskaya give protocols to prove knowledge of a committed value in their paper [7].

5.2 Alternative Protocol Using Cryptographic Accumulators

The reasons why we employed a signature scheme in our set-membership protocol is that the prover needed to show that he committed to a value for which he knows an authenticator without revealing that value or authenticator. Now it turns out that one can achieve exactly the same goal with cryptographic accumulators with similar complexities.

Recall cryptographic accumulators. A cryptographic accumulator is an algorithm that allows one to compress a list of elements into a single accumulator value. For each element there exists a witness attesting to the fact that the element is indeed contained in the accumulator value. For some cryptographic accumulator, there exists efficient proof protocols that allow a prover holding the element and the witness to prove to a verifier in zero knowledge that he indeed is privy to an element that is contained in the accumulator. Camenisch and Lysyanskaya have given an implementation of such an accumulator and a protocol that a committed value is indeed contained in the accumulator based on the strong RSA assumption[6].

Now the idea to build an efficient set-membership proof with dynamic accumulator is very similar to the signature based one: The verifier add each element in the set into the accumulator and sends the accumulator value to the prover together with the witness for each element. The prover then proves to the verifier that the value he has committed to is indeed contained in the accumulator produced by the verifier using the witness obtained for the verifier. This protocol is depicted in Appendix A for the SRSA-based accumulator.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 216483.

References

1. Bangerter, E., Camenisch, J., Maurer, U.M.: Efficient proofs of knowledge of discrete logarithms and representations in groups with hidden order. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 154–171. Springer, Heidelberg (2005)
2. Bellare, M., Goldreich, O.: On defining proofs of knowledge. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 390–420. Springer, Heidelberg (1993)
3. Black, K.: Classroom note: Putting constraints in optimization for first-year calculus students. SIAM Rev. 39(2), 310–312 (1997)
4. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
5. Boudot, F.: Efficient proofs that a committed number lies in an interval. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 431–444. Springer, Heidelberg (2000)
6. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
7. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (2003)
8. Camenisch, J., Neven, G., shelat, a.: Simulatable adaptive oblivious transfer. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 573–590. Springer, Heidelberg (2007)
9. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
10. Cheon, J.H.: Security analysis of the strong diffie-hellman problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 1–11. Springer, Heidelberg (2006)
11. Cramer, R., Damgård, I., MacKenzie, P.D.: Efficient zero-knowledge proofs of knowledge without intractability assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
12. Dodis, Y., Yampolskiy, A.: A verifiable random function with short proofs and keys. In: Public Key Cryptography, pp. 416–431 (2005)
13. Fujisaki, E., Okamoto, T.: Statistical zero knowledge protocols to prove modular polynomial relations. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 16–30. Springer, Heidelberg (1997)
14. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Cryptology ePrint Archive, Report 2006/165 (2006)
15. Groth, J.: Non-interactive zero-knowledge arguments for voting. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 467–482. Springer, Heidelberg (2005)
16. Lipmaa, H.: On diophantine complexity and statistical zero-knowledge arguments. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 398–415. Springer, Heidelberg (2003)
17. Micali, S., Rabin, M., Kilian, J.: Zero-knowledge sets. In: FOCS 2003: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, Washington, DC, USA. IEEE Computer Society Press, Los Alamitos (2003)

18. Schoenmakers, B.: Some efficient zeroknowledge proof techniques. In: International Workshop on Cryptographic Protocols, Monte Verita, Switzerland (March 2001)
19. Schoenmakers, B.: Interval proofs revisited. In: International Workshop on Frontiers in Electronic Elections, Milan, Italy (September 2005)
20. Teranishi, I., Sako, K.: K-times anonymous authentication with a constant proving cost. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 525–542. Springer, Heidelberg (2006)

A Accumulator Based Membership Proof

A.1 Cryptographic Accumulators and Proofs for Them

Definition 4. *[6] A secure accumulator for a family of inputs $\{\mathcal{X}_k\}$ is a family of families of functions $\mathcal{G} = \{\mathcal{F}_k\}$ with the following properties:*

Efficient generation: *There is an efficient probabilistic algorithm G that on input 1^k produces a random element f of $\mathcal{F}_k$. Moreover, along with f, G also outputs some auxiliary information about f, denoted t_f.*

Efficient evaluation: *$f \in \mathcal{F}_k$ is a polynomial-size circuit that, on input $(u, x) \in \mathcal{U}_f \times \mathcal{X}_k$, outputs a value $v \in \mathcal{U}_f$, where $\mathcal{U}_f$ is an efficiently-samplable input domain for the function f; and $\mathcal{X}_k$ is the intended input domain whose elements are to be accumulated.*

Quasi-commutative: *For all k, for all $f \in \mathcal{F}_k$, for all $u \in \mathcal{U}_f$, for all $x_1, x_2 \in \mathcal{X}_k$, $f(f(u, x_1), x_2) = f(f(u, x_2), x_1)$. If $X = \{x_1, \ldots, x_m\} \subset \mathcal{X}_k$, then by $f(u, X)$ we denote $f(f(\ldots(u, x_1), \ldots), x_m)$.*

Witnesses: *Let $v \in \mathcal{U}_f$ and $x \in \mathcal{X}_k$. A value $w \in U_f$ is called a* witness *for x in v under f if $v = f(w, x)$.*

Security: *Let $\mathcal{U}'_f \times \mathcal{X}'_k$ denote the domains for which the computational procedure for function $f \in \mathcal{F}_k$ is defined (thus $U_f \subseteq \mathcal{U}'_f$, $\mathcal{X}_k \subseteq \mathcal{X}'_k$). For all probabilistic polynomial-time adversaries $\mathcal{A}_k$,*

$$\begin{aligned}\Pr[f \leftarrow G(1^k); u \leftarrow \mathcal{U}_f; (x, w, X) \leftarrow \mathcal{A}_k(f, U_f, u) :\\ X \subset \mathcal{X}_k; w \in \mathcal{U}'_f; x \in \mathcal{X}'_k; x \notin X; f(w, x) = f(u, X)] = \mathrm{neg}(k)\end{aligned}$$

Note that only the legitimate accumulated values, $(x_1, \ldots, x_m)$, must belong to $\mathcal{X}_k$; the forged value x can belong to a possibly larger set $\mathcal{X}'_k$.

Implementation Based on the Strong RSA Assumption. Here we recall the Camenisch-Lysyanskaya accumulator [6].

- $\mathcal{F}_k$ is the family of functions that correspond to exponentiating modulo safe-prime products drawn from the integers of length k. Choosing $f \in \mathcal{F}_k$ amounts to choosing a random modulus $n = pq$ of length k, where $p = 2p' + 1$, $q = 2q' + 1$, and p,p',q,q' are all prime. We will denote f corresponding to modulus n and domain $\mathcal{X}_{A,B}$ by $f_{n,A,B}$. We denote $f_{n,A,B}$ by f_n or by f when it does not cause confusion.
- $\mathcal{X}_{A,B}$ is the $\{e \in \texttt{primes} : e \neq p', q' \ \wedge \ A \leq e \leq B\}$, where A and B can be chosen with arbitrary polynomial dependence on the security parameter k, as long as $2 < A$ and $B < A^2$. $\mathcal{X}'_{A,B}$ is (any subset of) of the set of integer from $[2, A^2 - 1]$ such that $\mathcal{X}_{A,B} \subseteq \mathcal{X}'_{A,B}$.

Common Input: g, h, a commitment C, and a set $\S$

Prover Input: s_j, r such that $C = g^{s_j} h^r$ and $s_j \in \S$.

$P \xleftarrow{\mathfrak{n}, \S_{ew}} V$ Verifier picks a safe prime product $\mathfrak{n} = (2\mathfrak{p}+1)(2\mathfrak{q}+1)$ and
a random quadratic residues $\mathfrak{u}, \mathfrak{g}, \mathfrak{h}$ modulo $\mathfrak{n}$,
picks random $u_i \in \{0,1\}^{k'}$ such that $e_i = s_i 2^k + u_i$ are prime.
computes $\mathfrak{v} \leftarrow \mathfrak{u}^{2\prod e_i} \bmod \mathfrak{n}; \mathfrak{w}_i \leftarrow \mathfrak{v}^{1/e_i} \bmod \mathfrak{n}$,
sends $\mathfrak{n}$, $\mathfrak{u}$, $\mathfrak{v}$, $\mathfrak{g}$, $\mathfrak{h}$, and $\S_{ew} \leftarrow \{(s_1, e_1, \mathfrak{w}_1).....(s_n, e_n, \mathfrak{w}_n)\}$
convinces the prover that $\mathfrak{g} \in \langle \mathfrak{h} \rangle$
(we will discuss the details separately below).

$P \xrightarrow{\mathfrak{W}, \mathfrak{R}, \mathfrak{C}} V$ Prover picks $r_1, r_2, r_3 \in \{0, \ldots, n2^\ell\}$,
where ℓ is a security parameter and
sends $\mathfrak{W} \leftarrow \mathfrak{w}_j \mathfrak{u}^{r_1} \bmod n$, $\mathfrak{R} \leftarrow \mathfrak{g}^{r_1} \mathfrak{h}^{r_2} \bmod n$
and $\mathfrak{C} \leftarrow \mathfrak{g}^{e_j} \mathfrak{h}^{r_3} \bmod n$

Prover and Verifier run

$$\begin{aligned}\mathrm{PK}\{(\alpha,\beta,\gamma,\delta,\epsilon,\rho,\rho_1,\rho_2,\rho_3,\phi,\xi,\nu):\quad & C = g^\sigma h^\rho \quad \wedge \\ & \mathfrak{C} = (\mathfrak{g}^{2^k})^\sigma \mathfrak{g}^\mu \mathfrak{h}^{\rho_3} \pmod n \quad \wedge \quad \mathfrak{R} = \mathfrak{g}^{\rho_1}\mathfrak{h}^{\rho_2} \pmod n \quad \wedge \\ & \mathfrak{v} = \mathfrak{W}^\epsilon (\tfrac{1}{\mathfrak{u}})^\delta \pmod n \quad \wedge \quad 1 = \mathfrak{R}^\epsilon \mathfrak{g}^\delta \mathfrak{h}^\phi \pmod n \\ & \wedge \quad \mu \in [-2^{k-1}, 2^{k-1}]\}\end{aligned}$$

Fig. 5. Set membership protocol for set §

- For $f = f_n$, the auxiliary information t_f is the factorization of n.
- For $f = f_n$, $\mathcal{U}_f = \{u \in QR_n : u \neq 1\}$ and $\mathcal{U}'_f = \mathbb{Z}_n^*$.
- For $f = f_n$, $f(u, x) = u^x \bmod n$.
  Note that $f(f(u, x_1), x_2) = f(u, \{x_1, x_2\}) = u^{x_1 x_2} \bmod n$

A.2 Membership Proof with Cryptographic Accumulators

We are now ready to employ the accumulator for the membership proof which can be used as an alternative building block for our range proof presented in Section 4.

One complication that we have to deal with here is that the accumulator allows one to accumulator prime number only whereas our set is arbitrary bits strings. We thus need to encode a mapping. This can be done as follows. Let $\{s_1, \ldots, s_n\}$ be our set, where we assume that the s_i are integers. We let $e_i = s_i 2^k + u_i$ where $u_i < 2^{k'} < 2^k$ is selected so that e_i is prime as k and k' are security parameters (we discuss them below). With this encoding, the verifier can accumulate all the e_i's and send the accumulator value, the e_i, and the corresponding witnesses to the prover. Now the prover has to prove that e_i that corresponds to the s_i in his commitment is contained in the accumulators. The resulting protocol is given in Figure A.1, where we adapt the accumulator proof given by Camenisch and Lysyanskaya [6] to our setting. That is, we have to additionally prove that the correspondence between the e_i and the committed

s_i holds. For this to work, the prover need that show he knows some u_i such that $e_i = s_i 2^k + u_i$ holds. Here it is of course important that this u_i be at most 2^{k-1} bits. This can be enforced efficiently provided that in reality u_i is a couple of bits smaller, i.e., k' bits, where in practice the difference should be about 300 bits for this to work. More precisely, we employ the first range proof discussed in Section 1.2.

Remarks: 1) We need to discuss how the verifier can convince the prover that $\mathfrak{g} \in \langle \mathfrak{h} \rangle$ holds. One way to achieve this, is that the prover runs with the verifier the protocol $PK\{(\alpha) : \mathfrak{g} = \mathfrak{h}^{\alpha} \pmod{\mathfrak{n}}\}$ using binary challenges. Another, more efficient, way is described by Bangerter et al.[1].

2) We note also, that for many applications, the parameters $\mathfrak{n}$, $\mathfrak{u}$, $\mathfrak{v}$, $\mathfrak{g}$, $\mathfrak{h}$, and $\S_{ew} \leftarrow \{(s_1, e_1, \mathfrak{w}_1).....(s_n, e_n, \mathfrak{w}_n)\}$ can be computed and published once and for all (possibly a trusted third party). In this case the computational complexity of our protocols becomes independent of the number of members in the set.