Efficient and Extractable Bit Commitment Scheme from Knowledge Assumptions

Wojciech Wodo, Lucjan Hanzlik

Wrocław University of Science Technology
wojciech.wodo@pwr.edu.pl, lucjan.hanzlik@pwr.edu.pl

Abstract. Commitment schemes are one of the main cryptographic primitives. Of particular interest are Pedersen commitments, which are based on the discrete logarithm problem and hide the committed value even against a powerful adversary.
Some applications require that the committing party proves that the committed value is from a specific range, e.g. $\{0, 1\}$. Moreover, in many application we would like to ensure that committing party "knows" the opening of the commitment. In case of Pedersen commitments this is usually ensured using non-interactive proofs of knowledge and range proofs, which are secure in the random oracle model.
In this paper we describe a different approach. We make use of the knowledge-of-exponent assumption and propose an extension to Pedersen commitments. Our idea not only makes the commitment scheme extractable (i.e. there exists an algorithm that given the same input as the committing party and the same random coins, outputs the committed message) but provides a proof that the committed message is a bit. What is more, our solution is efficient and requires only four time larger commitments than standard Pedersen commitments.

Key words: Pedersen Commitments, Knowledge Assumptions, Extraction, Bit Commitments

1 Introduction

Commitment schemes are one of the main cryptographic primitives and implement the real world idea of a lock box [1]. We can put a message inside the box and close it. By giving the box to the recipient, we "commit" to the message inside it. Obviously we cannot change what is inside the box, as the recipient holds the locked box. This property is called *binding*, as once committed we are bound to a particular message and cannot change it. On the other hand, the recipient cannot see what is the message, as the closed box hides the content. This property is called *hiding*. Finally, we can give the key to the recipient and he can open the box and check the message. The message and the key are usually called *opening information*.

One of the most popular schemes is the Pedersen commitment scheme [2]. It is designed for known order groups that are also used for DSA and ECDSA groups [3]. Given two generators of this group, g and h the commitment is of the form $Co = g^m \cdot h^r$, where m is the committed message and r is some randomness that hides the message. The main feature of this scheme is that it is unconditionally hiding. It means that even a powerful adversary cannot guess what is the committed message. The idea is that for a given commitment Co there exist opening information to different messages, thus without the correct opening information the recipient cannot guess the message. On the other hand, this commitment scheme is only computationally binding, i.e. if the committing party knows $x = \log_g(h)$, then it is able to change the committed message. Thus, if the discrete logarithm problem in the chosen group is hard, it is hard to change a once committed message. What is more important, the group can in particular be generated by the recipient. The knowledge of x does not help to break the hiding property but still ensures that the committing party cannot change the message inside the commitment.

For some applications it is required that the committing party proofs the knowledge of the opening information without revealing it or to prove that the committed message is a bit, i.e. is from the set $\{0, 1\}$. The common approach is to use non-interactive proofs of knowledge, which are usually based on Schnorr like protocols [4], and range proofs [5]. Common solutions are secure in the random oracle model [6] and thus only secure in theory. Recently, the non-interactive proof system presented by Groth and Sahai gained much attention [7]. This system can also be used in combination with Pedersen commitments as a replacement for the random oracle proof systems. However, it requires that a trusted third party computes a common reference string and thus is not applicable in cases where the committing party and the recipient communicate ad hoc.

Our Contribution

In this paper we propose two extensions of the Pedersen commitment scheme. Both schemes are extractable, i.e. in order to compute a commitment one must know a valid opening information. Moreover, one of them ensures that the committed message is a bit. Our constructions are solely based on the knowledge-of-exponent [8] and a variant of the computational Diffie-Hellman problem [12]. Both schemes are not only simple but also efficient. In particular, the resulting schemes only quadruple the size of Pedersen commitments.

2 Primitives

Before presenting our contribution we briefly review a few facts about bilinear maps, assumptions used in further sections.

2.1 Notation and Bilinear Groups

By $y \leftarrow \mathcal{A}(x)$ we denote the execution of algorithm $\mathcal{A}$ outputting y, on input x. In addition, the superscript $\mathcal{O}$ in $\mathcal{A}^{\mathcal{O}}$ means that algorithm $\mathcal{A}$ has access to oracle $\mathcal{O}$. We say that $\mathcal{A}$ is probabilistic polynomial-time (PPT) if $\mathcal{A}$ uses internal random coins and the computation for any input $x \in \{0,1\}^*$ terminates in polynomial time. By $r \xleftarrow{\$} S$ we mean that r is chosen uniformly at random over the set S. Furthermore, we will use $1_{\mathbb{G}}$ to denote the identity element in group $\mathbb{G}$ and $[k]P$ to denote point multiplication, where:

$$[k]P = \underbrace{P + \ldots + P}_{k\text{- times}}$$

and point $P = (x, y)$ lies on some curve E.

Definition 1 (Negligible Function). *A function $\epsilon(\lambda) : \mathbb{N} \to \mathbb{R}$ is negligible, if for every positive polynomial $poly(.)$ there exists an integer $N > 0$ such that for all security parameters $\lambda > N$ we have:*

$$|\epsilon(\lambda)| < \frac{1}{poly(\lambda)}$$

Definition 2 (Bilinear map). *Let us consider cyclic groups $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$, $(\mathbb{G}_T, \cdot)$ of a prime order q. Let P_1, P_2 be generators of respectively $\mathbb{G}_1$ and $\mathbb{G}_2$. We call $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ a* bilinear map *(pairing) if it is efficiently computable and the following holds:*

Bilinearity: $\forall (S, T) \in \mathbb{G}_1 \times \mathbb{G}_2$, $\forall a, b \in \mathbb{Z}_q$, *we have* $e([a]S, [b]T) = e(S, T)^{a \cdot b}$,
Non-degeneracy: $e(P_1, P_2) \neq 1$ *is a generator of group $\mathbb{G}_T$,*

Depending on the choice of groups we say that map e is of:

Type 1: if $\mathbb{G}_1 = \mathbb{G}_2$,
Type 2: if $\mathbb{G}_1$ and $\mathbb{G}_2$ are distinct groups and there exists an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$,
Type 3: if $\mathbb{G}_1$ and $\mathbb{G}_2$ are distinct groups and no efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is known.

Bilinear map groups are known to be instantiable with ordinary elliptic curves such as MNT curves [10] or curves introduced by Barreto and Naehrig [11] (in short BN-curves).

Definition 3 (Bilinear-group generator). *A bilinear-group generator is a polynomial-time algorithm* BGGen *that on input of a security parameter* λ *returns a bilinear group* $\mathsf{BG} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ *such that* $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$ *and* $\mathbb{G}_T$ *are groups of order* q *and* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *is a bilinear map.*

Remark 1. Although, the bilinear map defined in [11] uses elliptic curves, in this paper we will use the multiplicative notation for all groups.

2.2 Assumptions

In this subsection we define standard computational assumptions and non-standard knowledge assumptions.

Computational Assumptions We start by introducing the discrete logarithm problem in $\mathbb{G}_i$.

Definition 4 (Discrete Logarithm Problem (DLP)**).** *Given two elements* $g_i, g_i^x \in \mathbb{G}_i$*, output* x*. We say that an algorithm* $\mathcal{A}$ *has advantage* ϵ *in solving* DLP *in* $\mathbb{G}_i$ *of prime order* q *if:*

$$Pr[x \leftarrow \mathcal{A}(g_i, g_i^x)] \geq \epsilon,$$

where the probability is taken over the random choice of the generator $g_i \in \mathbb{G}_i$*, the random choice of* $x \in \mathbb{Z}_q$*, and the random bits of the algorithm* $\mathcal{A}$*.*

Definition 5 (Co-Discrete Logarithm Problem (Co-DLP)**).** *Given elements* $g_1, g_1^x \in \mathbb{G}_1$ *and* $g_2, g_2^x \in \mathbb{G}_2$*, output* x*. We say that an algorithm* $\mathcal{A}$ *has advantage* ϵ *in solving* Co-DLP *in* $\mathbb{G}_i$ *of prime order* q *if:*

$$Pr[x \leftarrow \mathcal{A}(g_1, g_1^x, g_2, g_2^x)] \geq \epsilon,$$

where the probability is taken over the random choice of the generators $g_1 \in \mathbb{G}_1$ *and* $g_2 \in \mathbb{G}_2$*, the random choice of* $x \in \mathbb{Z}_q$*, and the random bits of the algorithm* $\mathcal{A}$*.*

Experiment $\mathsf{Exp}^{kea1}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g)$**:**
$b \xleftarrow{s} \mathbb{Z}_q$; $B \leftarrow g^b$
$(C,Y) \leftarrow \mathbf{A}_n(q,g,B)$; $c \leftarrow \bar{\mathbf{A}}_n(q,g,B)$
If ($Y = C^b$ AND $g^c \neq C$) then return 1 else return 0

Fig. 1. Experiment $\mathsf{Exp}^{kea1}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g)$

Definition 6 (Computational Diffie-Hellman Problem in $\mathbb{G}_i$). *Given elements $g_i, g_i^a, g_i^b \in \mathbb{G}_i$, output $g_i^{ab} \in \mathbb{G}_i$. We say that an algorithm $\mathcal{A}$ has advantage ϵ in solving the* CDH *in $\mathbb{G}_i$ (of prime order q) if:*

$$\Pr[g_i^{ab} \leftarrow \mathcal{A}(g_i, g_i^a, g_i^b)] \geq \epsilon,$$

where the probability is taken over the random choice of the generator $g_i \in \mathbb{G}_i$, the random choice of $a, b \in \mathbb{Z}_q$, and the random bits of $\mathcal{A}$.

The computational Diffie-Hellman problem can be considered separately for each group. In the asymmetric pairing setting we can define the following variant of the CDH problem called co-Diffie-Hellman problem [12].

Definition 7 (co-Diffie-Hellman Problem). *Given elements $g_1, g_1^a, g_1^b \in \mathbb{G}_1$, $g_2, g_2^a \in \mathbb{G}_2$, output $g_1^{ab} \in \mathbb{G}_1$. We say that an algorithm $\mathcal{A}$ has advantage ϵ in solving the* co-DHP* *in $\mathbb{G}_1$ and $\mathbb{G}_2$ (of prime order q) if:*

$$\Pr[g_1^{ab} \leftarrow \mathcal{A}(g_1, g_1^a, g_1^b, g_2, g_2^a)] \geq \epsilon,$$

where the probability is taken over the random choice of the generators $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$, the random choice of $a, b \in \mathbb{Z}_q$, and the random bits of $\mathcal{A}$.

Knowledge of Exponent Assumptions We now recall two non-uniform definitions for knowledge of exponent assumptions. [8]

$GL = \{(q,g) : q, 2q+1 are primes and g is a generator of G_q\}$ $GL_n = \{(q,g) \in GL : |2q+1| = n\}$

Definition 8. *KEA1 Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ and $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ be families of circuits, and $v : \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$, any $(q,g) \in GL_n$, and any $A \in G_q$ the following experiment:*

We let

$$\mathbf{Adv}^{kea1}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) = \Pr[\mathsf{Exp}^{kea1}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) = 1]$$

Experiment $\mathsf{Exp}^{kea3}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A)$**:**
$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow g^b$; $X \leftarrow A^b$
$(C,Y) \leftarrow \mathbf{A}_n(q,g,A,B,X)$; $c_1,c_2 \leftarrow \bar{\mathbf{A}}_n(q,g,A,B,X)$
If ($Y = C^b$ AND $g^{c_1}A^{c_2} \neq C$) then return 1 else return 0

Fig. 2. Experiment $\mathsf{Exp}^{kea3}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A)$

denote the advantage of $\mathbf{A}$ *relative to* $\bar{\mathbf{A}}$ *on inputs* n,q,g*. We say that* $\bar{\mathbf{A}}$ *is a kea1-extractor for* $\mathbf{A}$ *with error bound* v *if*

$$\forall n \in \mathbb{N}\, \forall (q,g) \in GL_n : \mathbf{Adv}^{kea1}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) \leq v(n).$$

We say that KEA1 holds if for every poly-size family of circuits $\mathbf{A}$ *there exists a poly-size family of circuits* $\bar{\mathbf{A}}$ *and a negligible function* v *such that* $\bar{\mathbf{A}}$ *is a kea1-extractor for* $\mathbf{A}$ *with error bound* v*.*

Definition 9. *KEA3 according to [8] Let* $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ *and* $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ *be families of circuits, and* $v : \mathbb{N} \to [0,1]$ *a function. We associate to any* $n \in \mathbb{N}$*, any* $(q,g) \in GL_n$*, and any* $A \in G_q$ *the following experiment:*
We let

$$\mathbf{Adv}^{kea3}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) = \Pr[\mathsf{Exp}^{kea3}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A) = 1]$$

denote the advantage of $\mathbf{A}$ *relative to* $\bar{\mathbf{A}}$ *on inputs* n,q,g,A*. We say that* $\bar{\mathbf{A}}$ *is a kea1-extractor for* $\mathbf{A}$ *with error bound* v *if*

$$\forall n \in \mathbb{N}\, \forall (q,g) \in GL_n : \mathbf{Adv}^{kea3}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A) \leq v(n).$$

We say that KEA3 holds if for every poly-size family of circuits $\mathbf{A}$ *there exists a poly-size family of circuits* $\bar{\mathbf{A}}$ *and a negligible function* v *such that* $\bar{\mathbf{A}}$ *is a kea3-extractor for* $\mathbf{A}$ *with error bound* v*.*

2.3 Pedersen Commitments for Bilinear Groups

Definition 10 (Pedersen Commitment in $\mathbb{G}_i$**).** *Pedersen commitments consist of the following algorithms:*

$\mathsf{Setup}_{\mathsf{P}}(\lambda)$**:**
Compute a bilinear group $\mathsf{BG} = (q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T,e,g_1,g_2) \leftarrow \mathsf{BGGen}(\lambda)$*, choose* $x \xleftarrow{\$} \mathbb{Z}_q$*, compute* $X = g_i^x$ *and output the commitment key* $\mathsf{cpp} = (\mathsf{BG}, X)$ *(which is an implicit parameter to the rest algorithms).*

$\mathsf{Commit}_{\mathsf{P}}(m, r)$**:**
On input a message $m \in \mathbb{Z}_q$ *and a randomness* $r \in \mathbb{Z}_q$*, output the commitment* $Co = g_i^m \cdot X^r$ *and opening* $O = (m, r)$*.*

$\mathsf{Open}_{\mathsf{P}}(Co, O)$**:**
On input a commitment $Co \in \mathbb{G}_i$ *and an opening* O*, if* $Co = g_i^m \cdot X^r$ *output* m*, else output* $\perp$*.*

The above commitments are perfectly hiding and computationally binding under the DLP assumption in $\mathbb{G}_i$. Which is a classical result. However, To be more formal, we recall the game based definitions for this properties in Figures 3 and 4.

Experiment $\mathsf{Binding}_{\mathcal{A}}(\lambda)$**:**
$\mathsf{cpp} \leftarrow \mathsf{Setup}_{\mathsf{P}}(\lambda)$
$(Co, O_0, O_1) \leftarrow \mathcal{A}(\mathsf{cpp})$
If $\mathsf{Open}_{\mathsf{P}}(Co, O_0) = \mathsf{Open}_{\mathsf{P}}(Co, O_1) \neq \perp$
and $O_0 \neq O_1$ then return 1,
else return 0.

Fig. 3. Experiment $\mathsf{Binding}_{\mathcal{A}}(\lambda)$

Experiment $\mathsf{Hiding}_{\mathcal{A}}(\lambda)$**:**
$b \xleftarrow{\$} \{0, 1\}$
$\mathsf{BG} \leftarrow \mathsf{BGGen}(\lambda)$
$r_0, r_1 \xleftarrow{\$} \mathbb{Z}_q$
$(m_0, m_1, \mathsf{cpp}, \mathsf{St}) \leftarrow \mathcal{A}(\lambda)$
$Co_0 \leftarrow \mathsf{Commit}_{\mathsf{P}}(m_b, r_0)$
$Co_1 \leftarrow \mathsf{Commit}_{\mathsf{P}}(m_{1-b}, r_1)$
$\hat{b} \leftarrow \mathcal{A}(Co_0, Co_1, \mathsf{St})$
If $b = \hat{b}$ then return 1, else return 0.

Fig. 4. Experiment $\mathsf{Hiding}_{\mathcal{A}}(\lambda)$

Let

$$\mathbf{Adv}^{\mathsf{Binding}}_{P,\mathcal{A}}(\lambda) = \Pr[\mathsf{Exp}\ \mathsf{Binding}_{\mathcal{A}}(\lambda) = 1]$$

and

$$\mathbf{Adv}^{\mathsf{Hiding}}_{P,\mathcal{A}}(\lambda) = |2 \cdot \Pr[\mathsf{Exp}\ \mathsf{Hiding}_{\mathcal{A}}(\lambda) = 1] - 1|$$

denote the adversary's advantage for the binding and hiding experiments (Fig. 4).It is a known result that for Pedersen commitment we have:

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hiding}}(\lambda) = 0.$$

and

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Binding}}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathrm{DLP}_{\mathbb{G}_i}}(\lambda).$$

Remark 2. Since Pedersen commitments are perfectly hiding, we used a stronger hiding definition where the adversary can compute the commitment key cpp. Note that the knowledge of the exponent z does not help to break the hiding property.

3 A Bit Commitment Scheme from KEA-3

In this section we introduce a commitment scheme based on Pedersen commitments. The resulting scheme is unconditionally hiding and computationally binding. Moreover, the scheme is extractable under the KEA-3 assumption. We also extend this scheme and present a simple NIZK proof system that, under the KEA-3 and CDH assumptions, can be used to prove that the commited values are bits.

3.1 Construction

Definition 11 (Extractable Commitment Scheme).

$\mathsf{Setup}_{\mathsf{P}}(\lambda)$**:**
Compute a bilinear group $\mathsf{BG} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathsf{BGGen}(\lambda)$*, choose* $x, y \xleftarrow{\$} \mathbb{Z}_q$*, compute* $X = g_1^x$*,* $Y = g_1^y$ *and* $Z = Y^x$ *and output the commitment key* $\mathsf{cpp} = (\mathsf{BG}, X, Y, Z)$ *(which is an implicit parameter to the rest algorithms).*

$\mathsf{Commit}_{\mathsf{P}}(m, r)$**:**
On input a message $m \in \mathbb{Z}_q$ *and a randomness* $r \in \mathbb{Z}_q$*, output the commitment* $Co = (g_1^r \cdot X^m, Y^r \cdot Z^m)$ *and opening* $O = (m, r)$*.*

$\mathsf{Open}_{\mathsf{P}}(Co, O)$**:**
On input a commitment $Co \in \mathbb{G}_i$ *and an opening* O*, if* $Co = (g_1^r \cdot X^m, Y^r \cdot Z^m)$ *output* m*, else output* $\perp$*.*

Lemma 1. *The above scheme is unconditionally hiding, i.e.* $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hiding}}(\lambda) = 0$.

Proof (Sketch). Given a honestly generated commitment key $\mathsf{cpp} = (\mathsf{BG}, X, Y, Z)$, a commitment $Co = (g_1^r \cdot X^m, Y^r \cdot Z^m)$ and opening $O = (m, r)$, there always exists a opening $O' = (m', r')$, such that $Co = (g_1^{r'} \cdot X^{m'}, Y^{r'} \cdot Z^{m'})$. Note that this follows from the fact that Co can be rewritten as $Co = (w, w^y)$ and w is a Pedersen commitment to r, which is unconditionally hiding.

Lemma 2. *For the above scheme we have* $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Binding}}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathrm{DLP}_{\mathbb{G}_1}}(\lambda)$.

Proof (Sketch). The same reasoning as above can be applied, i.e. given a honestly generated commitment key $\mathsf{cpp} = (\mathsf{BG}, X, Y, Z)$, any commitment can be rewritten as $Co = (w, w^y)$, where w is a Pedersen commitment to r. Thus, if an adversary $\mathcal{A}$ outputs a different opening $O' = (m', r')$, then it can be used to solve the DLP in $\mathbb{G}_1$, i.e. to compute $\log_{g_1}(X)$.

The main difference between our commitment scheme and Pedersen commitment scheme is the second element $Y^r \cdot Z^m$. We use this second element to prove the following lemma.

Lemma 3. *Let $\mathcal{A}$ be an algorithm that on input receives the commitment key* cpp *and outputs a commitment $Co = (g_1^r \cdot X^m, Y^r \cdot Z^m)$. Then, there exists an algorithm $\mathcal{A}'$ that given the commitment key* cpp *and the same random coins as $\mathcal{A}$ returns (m, r).*

Proof (Sketch). Under the KEA-3 assumption, $\mathcal{A}$ can only compute Co if it "knows" m and r. Thus, there must exists an extraction algorithm $\mathcal{A}'$ that given the same input and random coins outputs (m, r).

3.2 Committing to Bits

Our commitments scheme can be used to commit to an arbitrarily element of $\mathbb{Z}_q$. For some application it is required to limit the message space to bits $\{0, 1\}$. Usually, this is ensured using non-interactive proof system. However, they either require random oracles or are not very efficient. Here we present a simple way to prove committing to a bit. Our method is based on simple algebra and the co-CDH and KEA-3 assumptions.

Let us first extend the commitment key by an additional element $X_2 = g_2^x$. In addition to the $Co = (g_1^r \cdot X^m, Y^r \cdot Z^m)$, the committing party also computes a Pedersen commitment in $\mathbb{G}_2$, i.e. $g_2^r \cdot X_2^m$. Note that since $\log_{g_1} X = \log_{g_2} X_2$, this element does not break the hiding property. Given those values, a verifier can compute the value

$$e(X \cdot g_1^{-r} \cdot X^{-m}, g_2^r \cdot X_2^m) = e(g_1, g_2)^{(-r+(1-m)\cdot x)\cdot(r+m\cdot x)}.$$

Let us take a look at the exponent $(-r + (1 - m) \cdot x) \cdot (r + m \cdot x)$. Simplifying this value we receive $-r^2 + r \cdot (1 - 2 \cdot m) \cdot x + (1 - m) \cdot m \cdot x^2$. The main observation is that the term x^2 only exists if $(1 - m) \cdot m \neq 0$, thus implying that $m \notin \{0, 1\}$. Therefore, if there exists an algorithm $\mathcal{A}$ that commits to a element $\notin \{0, 1\}$, then we can use it to solve the co-computational Diffie-Hellman assumption. However, in order to do so, we require the element $g_1^{x^2}$ but we only can compute $e(g_1, g_2)^{x^2}$ (we know m and r because of the extractor). Thus, we require that the committing party also computes a value T, such that $e(T, g_2) = e(X \cdot g_1^{-r} \cdot X^{-m}, g_2^r \cdot X_2^m)$. Then, $\log_{g_1} T = -r^2 + r \cdot (1 - 2 \cdot m) \cdot x + (1 - m) \cdot m \cdot x^2$ and knowing r and m, we can compute $g_1^{x^2}$.

It remains to argue how this helps us to solve the co-computational Diffie-Hellman problem. It is a known fact [9] that the square Diffie-Hellman problem, i.e. given g, g^x compute g^{x^2} is equivalent to the computational Diffie-Hellman problem, as we can compute $a \cdot b = 2^{-1} \cdot (a^2 + b^2 - (a - b)^2)$. The full scheme is given below.

Definition 12 (Extractable Bit Commitment Scheme).

$\mathsf{Setup}_\mathsf{P}(\lambda)$**:**

Compute a bilinear group $\mathsf{BG} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathsf{BGGen}(\lambda)$*, choose* $x, y \xleftarrow{\$} \mathbb{Z}_q$*, compute* $X_1 = g_1^x$*,* $Y_1 = g_1^y$*,* $Z = Y_1^x$*,* $X_2 = g_2^x$*,* $Y_2 = g_2^y$ *and output the commitment key* $\mathsf{cpp} = (\mathsf{BG}, X_1, X_2, Y_1, Y_2, Z)$ *(which is an implicit parameter to the rest algorithms).*

$\mathsf{Commit}_\mathsf{P}(m, r)$**:**

On input a message $m \in \{0, 1\}$ *and a randomness* $r \in \mathbb{Z}_q$*, output the commitment*

$$Co = (Co_1, Co_2, Co_3, Co_4) = (g_1^r \cdot X_1^m, Y_1^r \cdot Z^m, g_2^r \cdot X_2^m, g_1^{-r^2 + r\cdot(1-2\cdot m)\cdot x})$$

and opening $O = (m, r)$*.*

$\mathsf{Open}_\mathsf{P}(Co, O)$**:**

On input a commitment $Co \in \mathbb{G}_i$ *and an opening* O*, if* $Co = (g_1^r \cdot X_1^m, Y_1^r \cdot Z^m, \cdot, \cdot)$ *output* m*, else output* $\perp$*.*

In order to verify that a commitment $Co = (Co_1, Co_2, Co_3, Co_4)$ was computed in an honest way, i.e. using message space $\{0, 1\}$ one has to

verify that $e(Co_1, g_2) = e(g_1, Co_3)$, $e(X_1 \cdot Co_1^{-1}, Co_3) = e(Co_4, g_2)$ and that $e(Co_1, Y_2) = e(Co_2, g_2)$.

Theorem 1. *The above scheme is unconditionally hiding, i.e.* $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hiding}}(\lambda) = 0$.

Proof. Let $\mathsf{cpp} = (\mathsf{BG}, X_1, X_2, Y_1, Y_2, Z)$ be a honestly generated commitment key and r the randomness used to compute a commitment to the bit 0, i.e. $Co = (Co_1, Co_2, Co_3, Co_4) = (g_1^r, Y_1^r, g_2^r, g_1^{-r^2+r\cdot x})$. Note that there exists a value $r' = r - x$ such that $Co_1 = g_1^{r'} \cdot X_1$, $Co_2 = Y_1^{r'} \cdot Z$, $Co_3 = g_2^{r'} \cdot X_2$ and

$$Co_4 = g_1^{-(r'+x)^2+r'\cdot x+x^2} c = g_1^{-((r')^2+2\cdot r'\cdot x+x^2)+r'\cdot x+x^2} = g_1^{-(r')^2-r'\cdot x}.$$

Thus, both $(0, r)$ and $(1, r')$ are valid openings for the commitment Co. It follows that the commitment scheme is unconditionally hiding.

Theorem 2. *For the above scheme we have* $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Binding}}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\text{Co-DLP}}(\lambda)$.

Proof. Let (g_1, g_1^x, g_2, g_2^x) be an instance of the co-discrete logarithm problem for parameters BG. We construct the commitment key cpp as follows. First we compute $y, z \xleftarrow{\$} \mathbb{Z}_q$ and set $\mathsf{cpp} = (\mathsf{BG}, g_1^x, g_2^x, g_1^y, g_2^y, (g_1^x)^z)$. Now if there exists an algorithm $\mathcal{A}$ that can output commitment $Co = (Co_1, Co_2, Co_3, Co_4)$ and openings $(0, r_0)$ and $(1, r_1)$, then we can solve the co-discrete logarithm problem. Note that by definition $Co_1 = g_1^{r_0} = g_1^{r_1} \cdot X_1$. It follows that $X_1 = g_1^{r_0 - r_1}$ and $\log_{g_1} X_1 = r_0 - r_1$. Thus, by returning $r_0 - r_1$ we can solve the co-discrete logarithm problem

Theorem 3. *An adversary $\mathcal{A}$ that can commit to a message $\notin \{0, 1\}$, can be used to solve the co-computational Diffie-Hellman problem.*

Proof. We will show how to use $\mathcal{A}$ to compute squares, which as shown in [9] is equivalent to the computational Diffie-Hellman problem. Let $(g_1, g_1^a, g_1^b, g_2, g_2^a, g_2^b)$ be an instance of the co-Diffie-Hellman problem for parameters BG. We construct the commitment key cpp as follows. First we compute $y, z \xleftarrow{\$} \mathbb{Z}_q$ and set $\mathsf{cpp} = (\mathsf{BG}, g_1^a, g_2^a, g_1^y, g_2^y, (g_1^a)^z)$. This parameter is given to the adversary $\mathcal{A}$, which outputs a valid commitment Co. We use the extraction algorithm to extract $m \notin \{0, 1\}$ and randomness r. Since this is a valid commitment we know that: $Co_1 = g_1^r \cdot X_1^m$, $Co_3 = g_2^r \cdot X_2^m$ and $e(X_1 \cdot Co_1^{-1}, Co_3) = e(Co_4, g_2)$. It follows that:

$$\log_{g_1}(Co_4) = (a - \log_{g_1}(Co_1)) \cdot \log_{g_2}(Co_3) = (a - (r + a \cdot m)) \cdot (r + a \cdot m)$$

and

$$\log_{g_1}(Co_4) = (-r+a\cdot(1-m))\cdot(r+a\cdot m) = -r^2+r\cdot a\cdot(1-2\cdot m)+a^2\cdot(m-1)\cdot m.$$

However, since $m \notin \{0,1\}$ we have that:

$$(Co_4 \cdot g_1^{r^2} \cdot (g_1^a)^{-r\cdot(1-2\cdot m)})^{((m-1)\cdot m)^{-1}} = g_1^{a^2}.$$

The same can be done for g_1^b and g_1^{a+b}. Then, using the formula

$$a \cdot b = (a^2 + b^2 - (a-b)^2) \cdot 2^{-1}$$

we can compute the solution for the co-computational Diffie-Hellman problem.

4 Conclusions

We have proposed a practical extractable bit commitment scheme. Moreover, we proposed an extension that ensures that the message space is $\{0,1\}$, which is required by many application. We achieve this without random oracles or Groth-Sahai proofs. Moreover, our commitment scheme is efficient in terms of computational and space complexity. For a future work we plan to apply our commitment scheme to blind signatures.

Acknowledgements

This research was supported by the National Science Centre (Poland) based on decision no 2014/15/N/ST6/04577.

References

1. Brassard, G., Chaum, D., Crépeau, C.: Minimum Disclosure Proofs of Knowledge. J. Comput. Syst. Sci. **37**(2) (October 1988) 156–189
2. Pedersen, T.P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology. CRYPTO '91, London, UK, UK, Springer-Verlag (1992) 129–140
3. Childs, L.N., ed. In: Cyclic Groups and Cryptography. Springer New York, New York, NY (2009) 387–412
4. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. In Brassard, G., ed.: CRYPTO. Volume 435 of Lecture Notes in Computer Science., Springer (1989) 239–252

5. Camenisch, J., Chaabouni, R., Shelat, A.: Efficient Protocols for Set Membership and Range Proofs. In: Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology. ASIACRYPT '08, Berlin, Heidelberg, Springer-Verlag (2008) 234–252
6. Canetti, R., Goldreich, O., Halevi, S.: The Random Oracle Methodology, Revisited. J. ACM **51**(4) (July 2004) 557–594
7. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In Smart, N., ed.: Advances in Cryptology EUROCRYPT 2008. Volume 4965 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2008) 415–432
8. Bellare, M., Palacio, A.: The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In: Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. Volume 3152 of Lecture Notes in Computer Science., Springer (2004) 273–289
9. Bao, F., Deng, R.H., Zhu, H.: Variations of Diffie-Hellman Problem. In: In ICICS 03, volume 2836 of LNCS, Springer (2003) 301–312
10. Miyaji, A., Nakabayashi, M., Takano, S.: New Explicit Conditions of Elliptic Curve Traces for FR-Reduction. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **84**(5) (2001) 1234–1243
11. Barreto, P.S.L.M., Naehrig, M.: Pairing-Friendly Elliptic Curves of Prime Order. In Preneel, B., Tavares, S.E., eds.: Selected Areas in Cryptography. Volume 3897 of Lecture Notes in Computer Science., Springer (2005) 319–331
12. Chatterjee, S., Menezes, A.: On Cryptographic Protocols Employing Asymmetric Pairings – The Role of Ψ Revisited. Cryptology ePrint Archive, Report 2009/480 (2009) http://eprint.iacr.org/.