How to achieve bidirectional zero-knowledge authentication?

Jin Li[1,2]*& Xingyu Li[1,3][0000−0001−7705−0709], Chang Chen[1,4], Guoyu Yang[1], Junyang Li[1], Qi Chen[1], and Hongyang Yan[1]

[1] Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, China
neucc1997@gmail.com, yangguoyu1020@163.com, jhouse.nr@qq.com, chenqi@gzhu.edu.cn, hyang_yan@gzhu.edu.cn

[2] School of Computer Science, Guangzhou University, China and State Key Laboratory of Integrated Service Networks (ISN), Xidian University, China., Guangzhou, China
lijin@gzhu.edu.cn

[3] Cyberspace Security College, Guangzhou University, Guangzhou, China
XYLhdu19@163.com

[4] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China and Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, China
neucc1997@gmail.com

Abstract. Due to the completeness, reliability and zero-knowledge nature, the zero-knowledge proof is widely used to design various protocols, including zero-knowledge authentication protocols. However, the existing zero-knowledge proof scheme cannot realize bidirectional authentication. In this paper, we design a series of bidirectional zero-knowledge protocols based on two new flavors of operations applicable to multiplicative cyclic group. The two notions are formally defined in this paper. We also provide some formal definitions and properties for the two notions. According to our definitions, any bounded polynomial function defined on multiplicative cyclic group has duality and mirror. Based on the two operations, we introduce and formally define dual commitment scheme and mirror commitment scheme. Besides, we provide two efficient constructions for dual commitment and mirror commitment respectively based on CDH assumption and RSA assumption, and named $\mathbf{DC_{CDH}}$, $\mathbf{DC_{RSA}}$, $\mathbf{MC_{CDH}}$ and $\mathbf{MC_{RSA}}$ respectively. We also provide the extended version supporting multiple messages in the appendix. Then, we design some efficient non-interactive as well as interactive zero-knowledge authentication protocols based on these commitments. The protocols allow two participants to achieve mutual zero-knowledge authentication only a communication initialization is needed. Different from other commitment schemes, our schemes can't be used to construct other schemes for cryptography, such as, verifiable secret sharing, zero-knowledge sets, credentials and content extraction signatures, but also can provide technical support for privacy protection of users in distributed scenarios.

Keywords: duality, mirror, dual commitment, mirror commitment, zero-knowledge authentication, non-interactive protocol

1 Introduction

A zero-knowledge proof, proposed by Goldwasser, Micali and Rackoff in 1985 [8], has become a fundamental protocol in cryptography. Due to the completeness, reliability and zero-knowledge nature, the zero-knowledge proof is favored by experts and scholars. Then it is widely used for the construction of public key encryption [9], signature [10], identity authentication [11,17–19], secret sharing [23] and other classical cryptography fields as well as blockchain [12, 14–16, 24], privacy computing [13], cloud computing [20], MPC [21] and other popular technology. However, the efficiency, scalability and other problems make zero-knowledge proof unable to run on resource-constrained equipment. A large number of scholars have carried out in-depth research on this issue and proposed a variety of new zero-knowledge proof implementation schemes [26–32]. These schemes can all make a prover be able to convince a verifier of the validity of some NP statement disclosing more than the fact that the prover knows a witness that satisfies the statement efficiently. Some of schemes even has good performance. But, under the condition of only one initialization, all schemes can only verify the verifier's statement to the prover, and cannot realize role exchange. In this paper, we designed a series of new zero-knowledge authentication protocols based on our newly defined cryptographic primitive: dual commitment and mirror commitment. These protocols can achieve mutual zero-knowledge authentication with only a communication initialization needed. Besides, our schemes also can be widely used for the construction of other schemes, such as verifiable secret sharing, zero-knowledge sets, credentials and content extraction signatures and so on. Our main contributions are as follows.

- We first provide two new notions applicable to multiplicative cyclic group, named duality and mirror.
- We first propose two new cryptographic commitment schemes based on duality and mirror, which we call dual commitment scheme and mirror commitment scheme. Besides, we also provide two efficient constructions for dual commitment and mirror commitment respectively based on CDH assumption and RSA assumption, and named $\mathbf{DC_{CDH}}$, $\mathbf{DC_{RSA}}$, $\mathbf{MC_{CDH}}$ and $\mathbf{MC_{RSA}}$ respectively. Moreover, we give the extended version of these constructions, which supports multiple messages.
- We first design two efficient non-interactive zero-knowledge authentication protocols for these commitments. The protocols allow two participants to submit commitments to each other so that they can achieve mutual zero-knowledge authentication only a communication is needed.

2 Preliminaries

2.1 Notation

We denote by $poly(\lambda)$ any polynomial function that is bounded by a polynomial in λ, where $\lambda \in \mathbb{N}$ is the security parameter. We denote any function that is *negligible* in the security parameter with $negl(\lambda)$ if it vanishes faster than the inverse of any polynomial. We say that an algorithm is *ppt* if and only if it is modeled as a probabilistic turing machine that runs in time polynomial in λ. Given a set S, we denote by $x \leftarrow S$ that x is uniformly sampled from S.

2.2 Commitments

Commitment turned out to be an extremely important primitive in cryptography and has been used as a building block to realize highly non-trivial protocols and primitives. Informally, a commitment scheme is a two-phase protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. In committing phase, the prover $\mathcal{P}$ commits to a statement m with a string c using some appropriate algorithm. In the decommitting stage, the prover reveals the opening information *op* and the message m to the verifier, who can check whether c was indeed a valid commitment on m. A commitment scheme is said to be non-interactive if each phase requires only one message from $\mathcal{P}$ to $\mathcal{V}$. All algorithms have access to a public random string r generated by a trusted setup party.

In their most basic form commitment schemes are expected to meet hiding and binding. A commitment scheme is hiding means with this that it should not reveal information about the committed message to a computationally bounded attacker.

Definition 1 (Hiding). A commitment scheme with commitment algorithm *Commit* is hiding if there exists a negligible function $negl(\lambda)$ such that for any *ppt* attacker $\mathcal{A}$, for a randomly sampled $r \leftarrow \mathit{Setup}(1^\lambda)$, and for all pairs of messages (m_0, m_1), we have that

$$Pr[\mathcal{A}(r, c) = b | b \leftarrow 0, 1; c \leftarrow \mathit{Commit}(r, m_b)] \leq \frac{1}{2} + negl(\lambda).$$

Definition 2 (Binding). A verification algorithm Verify is binding if there exists a negligible function $negl(\lambda)$ such that for any *ppt* attacker $\mathcal{A}$ and for a randomly sampled $r \leftarrow \mathit{Setup}(1^\lambda)$, we have that

$$\begin{aligned} Pr[Verify(r, c, op, m) = 1 \wedge Verify(r, c, op', m') = 1 \wedge m \neq m' |& \\ (c, op, m, op', m') \leftarrow \mathcal{A}(r)] \leq negl(\lambda).& \end{aligned}$$

2.3 Computational Assumptions

Here we formally describe the computational hardness assumptions that we need for the security of our construction.

Definition 3 (Discrete Logarithm Assumption). DLA Let $\mathcal{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathcal{G}$. We say that the discrete logarithm problem is hard if, for a random integer $x \in \mathbb{Z}_p$ and for all *ppt* attackers $\mathcal{A}$, there exists a negligible function $negl(\lambda)$ such that

$$Pr[\mathcal{A}(\mathcal{G}, g, g^x) = x] \leq negl(\lambda).$$

Definition 4 (Computational Diffie-Hellman Assumption, CDH). Let $\mathcal{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathcal{G}$. We say that the computational Diffie-Hellman problem is hard if, for two random integers $x, y \in \mathbb{Z}_p$ and for all *ppt* attackers $\mathcal{A}$, there exists a negligible function $negl(\lambda)$ such that

$$Pr[\mathcal{A}(\mathcal{G}, g, g^x, g^y) = g^{xy}] \leq negl(\lambda).$$

Definition 5 (RSA Assumption, RSA). Let $\lambda \in \mathbb{N}$ be the security parameter, N is a random RSA modulus of length, z be a random element in $\mathbb{Z}_N$ and e be an $(\ell+1)$-bit prime (for a parameter ℓ). Then we say that the RSA assumption holds if for any *ppt* attackers $\mathcal{A}$, the probability

$$Pr[\mathcal{A}(N, y, y^e) = z] \leq negl(\lambda).$$

Definition 6 (Square Computational Diffie-Hellman Assumption, CDH) Let $\mathcal{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathcal{G}$ and $a \xleftarrow{\$} \mathbb{Z}_p$. We say that the Square Computational Diffie-Hellman Assumption holds in $\mathbb{G}$ if for every *ppt* attackers $\mathcal{A}$, the probability

$$Pr[A(g, g^a) = g^{a^2}] \leq negl(\lambda)$$

In [2,3] is shown that the Square-CDH assumption is equivalent to the classical Computational Diffie-Hellman (CDH) assumption.

2.4 Duality and Mirror Function on multiplicative cyclic group

Here we extend the notion of dual and mirror in logical algebra and provide a formal definition of duality and mirror applicable to multiplicative cyclic group.
Definition 6 (Dual on multiplicative cyclic group). Let $\mathcal{F}$ be a polynomial function defined on multiplicative cyclic group $\mathbb{G}$, where g is the generator of $\mathbb{G}$. Another polynomial function $\mathcal{F}^*$ defined on $\mathbb{G}$ is said to be the duality of function $\mathcal{F}$ if it may be obtained from $\mathcal{F}$ by replacing the corresponding operation symbols with the following replacement rules and has the same operation order as $\mathcal{F}$, recording as $\mathcal{F} \triangleright \mathcal{F}^*$.

- Replace $+, \times$ with $\times, +$.
- Replace $-, /$ with $/, -$, where $/$ represents the inverse operation defined on multiplicative cyclic group.
- Replace $1, 0$ with $0, 1$.

To facilitate readers to better understand the definition, we give three extended definitions and three examples to explain these definitions.
Definition 7 (unidirectional Dual) If $\mathcal{F}^*$ is the duality of $\mathcal{F}$ while $\mathcal{A}^*$ is not the duality of $\mathcal{F}$. We say $\mathcal{F}$ and $\mathcal{F}^*$ are unidirectional dual, recording as $\mathcal{F} \triangleright \mathcal{F}^*$.
Definition 8 (Bidirectional Dual) If $\mathcal{F}$ is the duality of $\mathcal{F}^*$ while $\mathcal{F}^*$ is also the duality of $\mathcal{F}$. We say $\mathcal{F}$ and $\mathcal{F}^*$ are bidirectional dual, recording as $\mathcal{F} \triangleleft\triangleright \mathcal{F}^*$.

Definition 9 (Self Dual) If $\mathcal{F}$ is the duality of $\mathcal{F}$. We say $\mathcal{F}$ is self-dual, recording as $\overset{\triangledown}{\mathcal{F}}$.
Example 1 $\mathcal{F}^* = x * g - y * h$ is the duality of $\mathcal{F} = g^x/h^y$, where g,h,x,$y \in \mathbb{G}$. However, $\mathcal{F}^*$ is not the duality of $\mathcal{F}$. Then, $\mathcal{F} \triangleright \mathcal{F}^*$.
Example 2 $\mathcal{F}^* = (x+g) * (y-h) \triangleleft\triangleright \mathcal{F} = xg + y/h$ are Bidirectional dual.
Example 3 $\mathcal{F}^* = z$ is self dual, where $z \in \mathbb{G}$.
Definition 10 (Mirror on multiplicative cyclic group). Let $\mathcal{F} = \sum_{i=0}^{n} a_i x_i^{b_i}$ be a polynomial function defined on multiplicative cyclic group $\mathbb{G}$, where g is the generator of $\mathbb{G}$ and $\forall i \in \mathcal{Z}_p, a_i, x_i, b_i \in \mathbb{G}$. Another polynomial function $\mathcal{F}^\star$ defined on $\mathbb{G}$ is said to be the mirror of function $\mathcal{F}$ if it equals to $\sum_{i=0}^{n} a_{n-i} x_i^{b_{n-i}}$ and has the same operation order as $\mathcal{F}$, recording as $\mathcal{F} \Leftrightarrow \mathcal{F}^\star$. To facilitate readers to better understand the definition, we give a extended definition, a example and a theorem based on definition 10.
Proposition 1. If $\mathcal{F}^\star$ is the mirror of $\mathcal{F}$, then $\mathcal{F}$ must also be the mirror of $\mathcal{F}^\star$. It can be easily proved according to definition 10.
Definition 11 (Self Mirror) If $\mathcal{F}$ is the mirror of $\mathcal{F}$. We say $\mathcal{F}$ is self-mirror, recording as $\overset{\star}{\mathcal{F}}$.
Example 4 If $\mathcal{F} = \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} a_i x_i^{b_i} + a_i x_{n-i}^{b_i}$, then, $\mathcal{F}$ must be self-mirror.
Proposition 2. If $\mathcal{F}$ is a $poly(\lambda)$ defined on multiplicative cyclic group $\mathbb{G}$, where g is the generator of $\mathbb{G}$, $\mathcal{F}^*$ is the duality of $\mathcal{F}$ and $(F^*)^\star$ is the mirror of $\mathcal{F}^*$, $\mathcal{F}^\star$ is the mirror of $\mathcal{F}$ and $(\mathcal{F}^\star)^*$ is the duality of $\mathcal{F}^\star$ then $(\mathcal{F}^\star)^* = (\mathcal{F}^*)^\star$, recording as $\mathcal{F}^{*\star}$. We show the diagram for $\mathcal{F}$, $\mathcal{F}^*$, $\mathcal{F}^\star$ and $\mathcal{F}^{*\star}$ in Fig.1.
Example 5 If $\mathcal{F} = \sum_{i=0}^{n} a_i x_i^{b_i}$, then, we can get that $\mathcal{F}^\star = \sum_{i=0}^{n} a_{n-i} x_i^{b_{n-i}}$, $\mathcal{F}^* = \sum_{i=0}^{n} (a_i + b_i x_i)$. Then we can compute that $(\mathcal{F}^\star)^* = \sum_{i=0}^{n} (a_{n-i} + b_{n-i} x_i)$ and $(\mathcal{F}^*)^\star = \sum_{i=0}^{n} (a_{n-i} + b_{n-i} x_i)$. Obviously, $(\mathcal{F}^*)^\star = (\mathcal{F}^\star)^* = \mathcal{F}^{*\star}$.

$$\begin{tikzcd}
\mathcal{F} \arrow[r, "\triangleright"] \arrow[d, leftrightarrow, "\Updownarrow"'] & \mathcal{F}^* \arrow[d, leftrightarrow, "\Updownarrow"] \\
\mathcal{F}^\star \arrow[r, "\triangleright"] & \mathcal{F}^{*\star}
\end{tikzcd}$$

(a) the dual is unidirectional dual

$$\begin{tikzcd}
\mathcal{F} \arrow[r, "\triangleleft\triangleright"] \arrow[d, leftrightarrow, "\Updownarrow"'] & \mathcal{F}^* \arrow[d, leftrightarrow, "\Updownarrow"] \\
\mathcal{F}^\star \arrow[r, "\triangleleft\triangleright"] & \mathcal{F}^{*\star}
\end{tikzcd}$$

(b) the dual is Bidirectional dual

Fig. 1: Relation for $\mathcal{F}$, $\mathcal{F}^*$, $\mathcal{F}^\star$ and $\mathcal{F}^{*\star}$

3 Dual Commitment

In this section, on the basis of the definition of duality in section 2.4, we provide a formal definition of a dual commitment scheme, followed by two constructions.

In the first construction, the commitment, designed based on CDH Assumption, is a unidirectional dual commitment. While in the second construction, the commitment, designed based on RSA Assumption, is a bidirectional dual commitment. We also prove the security properties and discuss some useful features of our constructions.

3.1 Definition

A dual commitment consists of seven *ppt* algorithms: **Setup**, **Commit**, **Open**, $\mathbf{Verify}^{\mathbf{part}}$, $\mathbf{Verify}^{\mathbf{full}}$, $\mathbf{Update}^{\mathbf{message}}$ and $\mathbf{Update}^{\mathbf{proof}}$.

- $(c, pp) \leftarrow \mathbf{Setup}(1^\lambda)$ Given the security parameter λ, the setup algorithm *Setup* outputs a public random string c and some public parameters pp (which implicitly define the message space $\mathcal{M}_{pp}$, randomizer space R_{pp} and commitment space C_{pp}.)
- $(c, c^*, aux) \leftarrow \mathbf{Commit}(r, m, pp)$ Given the public random string r, a message m and public parameters pp, the commitment algorithm $Commit$ outputs a commitment c, a dual commitment c^* and corresponding auxiliary information aux.
- $op \leftarrow \mathbf{Open}(m, aux, pp)$ This algorithm is run by the committer to produce a proof op that m is the committed message and pp is the public parameters. In particular, notice that in the case when some updates have occurred the auxiliary information aux can include the update information produced by these updates.
- $b \leftarrow \mathbf{Verify}^{\mathbf{part}}(c, m, c|c^*, pp, op)$ Given the public random string c, a message m, a commitment c and opening information op, the partial verification algorithm $Verify^{part}$ outputs 1 if op is a valid opening for commitment c or dual commitment c^* on message m.
- $(b, t) \leftarrow \mathbf{Verify}^{\mathbf{full}}(c, m, c, c^*, pp, op)$ Given the public random string c, a message m, a commitment c, a commitment c^*, opening information op, the full verification algorithm $Verify^{full}$ outputs b=1 if op is a valid opening for commitment c and dual commitment c^* on message m. $Verify^{full}$ outputs t=2 if b=1 and $c \lhd\rhd c^*$ are Bidirectional dual, outputs t=1 if b=1 and $c \rhd c^*$, outputs t=- 1 if other conditions occur.
- $(c', c^{*'}, U) \leftarrow \mathbf{Update}^{\mathbf{message}}(c, c^*, m, m')$ This algorithm is run by the committer to update the dual commitment by changing the message m to m'. The algorithm takes as input the old message m, the new message m', the commitment c and the dual commitment c^* of message m. It outputs a new commitment c' and a new dual commitment $c^{*'}$ together with an update information U.
- $(op') \leftarrow \mathbf{Update}^{\mathbf{proof}}(c, c^*, U, op)$ This algorithm can be run by any user who holds a proof op for message m, and it allows the user to compute an updated proof op' (and the updated commitment c' and $c^{*'}$) such that op' will be valid. Basically, the value U contains the updated information.

For correctness, we require that $\forall \lambda \in \mathbb{N}$, for all honestly generated parameters pp, an honest committer should be able to correctly generate a commitment, a dual commitment and a proof op for all message $m \in \mathcal{M}$. Then, a honest verifier can correctly verify the correctness of a proof, a commitment and a dual commitment and the relevance of the commitment and the dual commitment for all message $m \in \mathcal{M}$.

For security, we require that a malicious committer should not be able to convincingly present two different messages m and m' with respect to c and c^*. we formally define the security and correctness of a dual commitment scheme.

Definition 12. We say (**Setup**, **Commit**, **Open**, $\mathbf{Verify^{part}}$, $\mathbf{Verify^{full}}$, $\mathbf{Update^{message}}$ and $\mathbf{Update^{proof}}$) is a secure dual commitment scheme if it satisfies the following properties.

Correctness. Let $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$ and $(c, c^*\ , aux) \leftarrow \mathbf{Commit}(r, m, pp)$. For a commitment c and a dual commitment c^* output by $\mathbf{Commit}(r, m, pp)$, and all $m \in \mathcal{M}$, the output of $\mathbf{Open}(m, aux, pp)$ can be successfully verified by $\mathbf{Verify^{part}}(r, m, c|c^*, pp, op)$ and $\mathbf{Verify^{full}}(r, m, c, c^*, pp, op)$.

Binding. For all adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, where $\mathcal{A}_0$ is *ppt* (and $\mathcal{A}_1$ is not computationally bounded), and for a randomly sampled $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$, we have that:

$$
\begin{aligned}
Pr[Verify^{part}(r, m, c|c^*, pp, op) = 1 \wedge Verify^{part} \\
(r, m', c|c^*, pp, op') = 1 \wedge m \neq m' | (c|c^*, op, m) \leftarrow \\
\mathcal{A}_0(r); (m', op') \leftarrow \mathcal{A}_1(r, state)] \leq negl(\lambda).
\end{aligned}
$$

Besides,

$$
\begin{aligned}
Pr[Verify^{full}(r, m, c, c^*, pp, op) = 1 \wedge Verify^{full} \\
(r, m', c, c^*, pp, op') = 1 \wedge m \neq m' | (c, c^*, op, m) \leftarrow \\
\mathcal{A}_0(r); (m', op') \leftarrow \mathcal{A}_1(r, state)] \leq negl(\lambda).
\end{aligned}
$$

Hiding.for any *ppt* attacker $\mathcal{A}$, for a randomly sampled $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$, and for all pairs of messages (m_0, m_1), we have that

$$
Pr[\mathcal{A}(r, c, c^*) = b | b \leftarrow 0, 1; (c, c^*, aux) \leftarrow Commit(r, m, pp)] \leq \frac{1}{2} + negl(\lambda).
$$

3.2 A unidirectional Dual Commitment based on CDH: $\mathbf{DC_{CDH}}$

Here we propose an implementation of concise unidirectional dual commitment $\mathbf{DC_{CDH}}$ for single message based on the CDH assumption of multiplicative cyclic group of order p proportional to the security parameter λ, where g is the generator . Precisely , the security of the scheme reduces to the Square Computational Diffie-Hellman assumption (see Definition 6 in Section 2.1), which has been shown equivalent of the standard CDH assumption [2, 3](see Definition 4 in Section 2.1).

$\mathbf{Setup}(1^\lambda)$ Let $\mathbb{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathbb{G}$. Randomly choose $z_c, z_1, z_2 \leftarrow \mathbb{Z}_p$. Set $r = g^{z_c}$, $h_1 = g^{z_1}$, $h_2 = g^{z_2}$. Set $pp = (g, h_1, h_2)$. The message space is $\mathcal{M} = \mathbb{Z}_p$.

Commit(r, m, pp) Compute

$$c = h_1^m h_2^r,\ c^* = m * h_1 + r * h_2$$

and output $C = (c, c^*, aux)$ and the auxiliary information $aux = none$.
Open(m, r, pp) Compute

$$op_c = h_1^m,\ op_{c^*} = m * h_1$$

and output $op = (op_c, op_{c^*})$.
Verify$^{\textbf{part}}(r, m, c|c^*, pp, op_c|op_{c^*})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } b = 1 \text{ and } c = op_c * h_2^r \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c^* = op_{c^*} + r * h_2 \\ 0 & \text{otherwise} \end{cases}$$

and output $b_1 \vee b_2$.
Verify$^{\textbf{full}}(r, m, c, c^*, pp, op_c, op_{c^*})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } b = 1 \text{ and } c = op_c * h_2^r \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c^* = op_{c^*} + r * h_2 \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2,\quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \lhd\rhd c^* \\ 0, & \text{if } b = 1 \text{ and } c \rhd c^* \\ -1 & \text{otherwise} \end{cases}$$

and output (b,t).
Update$^{\textbf{message}}(c, c^*, m, m')$ Compute the updated commitment $c' = c * h_1^{m'-m}$ and dual commitment $c^{*'} = c^* + h_2(m' - m)$. Finally output $C' = (c', c^{*'})$ and $U = (m, m')$.
Update$^{\textbf{proof}}(c, c^*, U, op)$ A client who owns a proof op, that is valid to c and c^* for the message m, can produce a new proof $op' = (op_c * h_1^{m'-m}, op_{c^*} * h_1(m' - m)$.

The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem.
Theorem 1. If the CDH assumption holds, then the scheme defined above is a concise dual commitment.

proof 1 We prove the theorem by showing that the scheme satisfies the binding property. For the sake of contradiction assume that there exists an efficient attackers $\mathcal{A}$ who produces two valid openings to two different messages, then we show how to build an efficient algorithm $\mathcal{B}$ to break the CDH assumption. First, $\mathcal{B}$ chooses $z_1, z_2, z_3 \leftarrow \mathbb{Z}_p$,it computes: $h_1 = g^{z_1}$, $h_2 = g^{z_2}, r = g^{z_3}$. $\mathcal{B}$ sets $pp = (g, h_1, h_2)$ and runs $\mathcal{A}(pp)$. Notice that the public parameters are perfectly distributed as the real ones. The adversary is supposed to output a tuple $(c, c^*, m, m', op_c, op_{c^*}, op'_c, op'_{c^*})$ such that $m \neq m'$ and both op_c, op_{c^*} and op'_c, op'_{c^*} correctly verify. Then $\mathcal{B}$ computes

$$h_1 = (op_c/op'_c)^{(m-m')^{-1}} = (op_{c^*} - op'_{c^*}) * (m - m')^{-1}$$

To see that the output is correct, observe that since the two openings verify correctly, then it holds:

$$op_c * h_1^{m'} = op'_c * h_1^m$$

$$op_{c^*} + h_1^{m'} = op'_{c^*} + h_1^m$$

which means that

$$h_1^{(m-m')} = op'_c / op_c = op'_{c^*} - op_{c^*}$$

One can easily see that this justifies the correctness of $\mathcal{B}$'s output. Notice that if $\mathcal{B}$ has probability ϵ of breaking the Square CDH assumption.

3.3 A Double Dual Commitment based on RSA: $\mathbf{DC_{RSA}}$

Here we propose a realization of double dual commitment $\mathbf{DC_{RSA}}$ for a single message from the RSA assumption (whose definition is given in section 2.1). Appendix A shows the double dual commitment scheme supporting multiple messages.

Setup$(1^\lambda, \ell)$ Randomly choose two $\ell/2$-bit primes p_1, p_2, set $N = p_1 p_2$, and then choose $2(\ell+1)$-bit primes e_1, e_2, a, r that do not divide $\varphi(N)$. Compute,

$$S_1 = a^{e_2}, S_2 = a^{e_1}$$

The public parameters pp are $(N, a, r, S_1, S_2, e_1, e_2)$. The message space is $M = \{0,1\}^\ell$.

Commit(r, m, pp) Compute

$$c = S_1^m S_2^r = a^{e_2 m + e_1 r},\ c^* = a^{(e_2+m)(e_1+r)}$$

and output $C = (c, c^*, aux)$ and the auxiliary information $aux = none$.

Open(m, r, pp) Compute

$$op_c = S_1^{\frac{m}{e_2}},\ op_{c^*} = S_1^{\frac{r}{e_1}} S_2^{\frac{m}{e_2}}$$

and output $op = (op_c, op_{c^*})$. Notice that knowledge of pp allows to compute op_c efficiently without the factorization of N.

$\mathbf{Verify^{part}}(r, m, c|c^*, pp, op_c|op_{c^*})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } S_1 S_2 op_{c^*}^{e_1} a^{m*r}\ mod\ N = op_{c^*} \\ 0 & \text{otherwise} \end{cases}, \quad b_2 = \begin{cases} 1, & \text{if } S_2^r op_c^{e_2}\ mod\ N = op_c \\ 0 & \text{otherwise} \end{cases}$$

and output $b = b_1 \vee b_2$.

$\mathbf{Verify^{full}}(r, m, c, c^*, pp, op_c, op_{c^*})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } S_1 S_2 op_{c^*}^{e_1} a^{m*r}\ mod\ N = c^* \\ 0 & \text{otherwise} \end{cases}, \quad b_2 = \begin{cases} 1, & \text{if } (S_2^r op_c^{e_2})\ mod\ N = c \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2, \quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \triangleleft\triangleright c^* \\ 0, & \text{if } b = 1 \text{ and } c \triangleright c^* \\ -1 & \text{otherwise} \end{cases}$$

and output (b,t).

Update$^{\textbf{message}}$ (c, c^*, m, m') Compute the updated commitment $c' = c * S_1^{m'-m}$ and dual commitment $c^{*'} = c^* * a^{(e_1+r)(m'-m)}$. Finally output $C' = (c', c^{*'})$ and $U = (m, m')$.

Update$^{\textbf{proof}}$ (c, c^*, U, op) A client who owns a proof op, that is valid to c and c^* for the message m, can produce a new proof $op' = (op_c * S_1^{\frac{m-m'}{e_2}}, op_{c^*} * S_2^{\frac{m-m'}{e_1}})$.

In order for the verification process to be correct, notice that one should also check that the S_1, S_2 are correctly generated with respect to a and the exponents e_1, e_2. The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem.

Theorem 2. If the RSA assumption holds, then the scheme defined above is a concise dual commitment.

proof 2 We prove the theorem by showing that the scheme satisfies the binding property. More precisely, assume for the sake of contradiction that there exists an efficient adversary that produces two valid openings to two different messages, then we show how a *ppt* attacker $\mathcal{A}$ builds an algorithm $\mathcal{B}$ that breaks the RSA assumption. First, $\mathcal{B}$ is run on input (N, z, e_1, e_2), where e is an $(\ell+1)$-bit prime, then, it is used to compute a value y such that $z_1 = y^{e_1} mod N, z_2 = y^{e_2} mod N$. The proceeds are as follows. First, it sets $a_1 = z_1, a_2 = z_2$. $\mathcal{B}$ runs *Setup* and gets back $(S_1, S_2, m, m', op_c, op_{c^*}, op', op'_{c^*})$ where $m \neq m'$ and both $op_c, op_{c'}$ and op_{c^*}, op'_{c^*} are correctly verified. From the equations $S_1^m op_c^{e_2} = S_1^{m'} op_c'^{e_2}$, $S_2^m op_{c^*}^{e_1} = S_2^{m'} op_{c^*}'^{e_1}$ we get

$$S_1^{m-m'} = op_c / op_c'^{\,e_2}, \; S_2^{m-m'} = op_{c^*} / op_{c^*}'^{\,e_1}$$

if $op_c/op'_c = 1$ *or* $op_{c^*}/op'_{c^*} = 1$ then we can factor with non-negligible probability. Thus, assuming $op_c/op'_c \neq 1$ *and* $op_{c^*}/op'_{c^*} \neq 1$ we can apply Shamir's trick [4] to get an $e_1 - th$ root of a_1, a_2. In particular, since $gcd(me_1, e_2) = 1$, by the extended Euclidean Algorithm, we can compute two integers λ, μ such that $m\lambda e_1 + \mu e_2 = 1$. This leads to the equation

$$a_1 = (op_c/op'_c)^{\lambda e_2} a^{\mu e_1}, \; a_2 = (op_{c^*}/op'_{c^*})^{\lambda e_2} a^{\mu e_1}$$

thus $(op_c/op'_c)^{\lambda e_2} a^{\mu e_1}$ *and* $(op_{c^*}/op'_{c^*})^{\lambda e_2} a^{\mu e_1}$ is the required corresponding root.

4 Mirror Commitment

In this section, On the basis of the definition of mirror in section 2.4, we provide a formal definition of a mirror commitment scheme, followed by two constructions. In the first construction, the commitment was designed based on CDH

Assumption. While in the second construction, the commitment was designed based on RSA Assumption. We also prove the security properties and discuss some useful features of our constructions.

4.1 Definition

A mirror commitment consists of seven *ppt* algorithms: **Setup**, **Commit**, **Open**, $\mathbf{Verify^{part}}$, $\mathbf{Verify^{full}}$, $\mathbf{Update^{message}}$ and $\mathbf{Update^{proof}}$.

- $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$ Given the security parameter λ, the setup algorithm *Setup* outputs a public random string r and some public parameters pp (which implicitly define the message space $\mathcal{M}_{pp}$, randomizer space R_{pp} and commitment space C_{pp}.)
- $(c, c^\star, aux) \leftarrow \mathbf{Commit}(r, m, pp)$ Given the public random string r, a message m and public parameters pp, the commitment algorithm $Commit$ outputs a commitment c, a dual commitment $c^\star$ and corresponding auxiliary information aux.
- $op \leftarrow \mathbf{Open}(m, aux, pp)$ This algorithm is run by the committer to produce a proof op that m is the committed message and pp is the public parameters. In particular, notice that in the case when some updates have occurred the auxiliary information aux can include the update information produced by these updates.
- $b \leftarrow \mathbf{Verify^{part}}(r, m, c|c^\star, pp, op)$ Given the public random string r, a message m, a commitment c and opening information op, the partial verification algorithm $Verify^{part}$ outputs 1 if op is a valid opening for commitment c or dual commitment $c^\star$ on message m.
- $(b, t) \leftarrow \mathbf{Verify^{full}}(r, m, c, c^\star, pp, op)$ Given the public random string r, a message m, a commitment c, a commitment $c^\star$, opening information op, the full verification algorithm $Verify^{full}$ outputs b=1 if op is a valid opening for commitment c and dual commitment $c^\star$ on message m. $Verify^{full}$ outputs t=1 if b=1 and $c \Leftrightarrow c^\star$, and outputs t=0 if other conditions occur.
- $(c', c^{\star'}, U) \leftarrow \mathbf{Update^{message}}(c, c^\star, m, m')$ This algorithm is run by the committer to update the dual commitment by changing the message m to m'. The algorithm takes as input the old message m, the new message m', the commitment c and the dual commitment $c^\star$ of message m. It outputs a new commitment c' and a new dual commitment $c^{\star'}$ together with an updated information U.
- $(op') \leftarrow \mathbf{Update^{proof}}(c, c^\star, U, op)$ This algorithm can be run by any user who holds a proof op for message m, and it allows the user to compute an updated proof op' (and the updated commitment c' and $c^{\star'}$) such that op' will be valid. Basically, the value U contains the update information.

For correctness, we require that $\forall \lambda \in \mathbb{N}$, for all honestly generated parameters pp, an honest committer should be able to correctly generate a commitment, a mirror commitment and a proof op for all message $m \in \mathcal{M}$. Then, an honest verifier can correctly verify the correctness of a proof, a commitment and a mirror commitment and the relevance of the commitment and the mirror commitment for all messages $m \in \mathcal{M}$.

For security, we require that a malicious committer should not be able to convincingly present two different messages m and m' with respect to c and $c^\star$. we formally define the security and correctness of a mirror commitment scheme.

Definition 13. We say (**Setup**, **Commit**, **Open**, $\mathbf{Verify}^{\mathbf{part}}$, $\mathbf{Verify}^{\mathbf{full}}$, $\mathbf{Update}^{\mathbf{message}}$ and $\mathbf{Update}^{\mathbf{proof}}$) is a secure dual commitment scheme if it satisfies the following properties.

Correctness. Let $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$ and $(c, c^\star\ , aux) \leftarrow \mathbf{Commit}(r, m, pp)$. For a commitment c and a mirror commitment c^* output by $\mathbf{Commit}(r, m, pp)$, and all $m \in \mathcal{M}$, the output of $\mathbf{Open}(m, aux, pp)$ can be successfully verified by $\mathbf{Verify}^{\mathbf{part}}(r, m, c|c^\star, pp, op)$ and $\mathbf{Verify}^{\mathbf{full}}(r, m, c, c^\star, pp, op)$.

Binding. For all adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, where $\mathcal{A}_0$ is *ppt* (and $\mathcal{A}_1$ is not computationally bounded), and for a randomly sampled $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$, we have that:

$$\begin{aligned} Pr[Verify^{part}(r, m, c|c^\star, pp, op) = 1 \wedge Verify^{part} \\ (r, m', c|c^\star, pp, op') = 1 \wedge m \neq m' | (c|c^\star, op, m) \leftarrow \\ \mathcal{A}_0(r); (m', op') \leftarrow \mathcal{A}_1(r, state)] \leq negl(\lambda). \end{aligned}$$

Besides,

$$\begin{aligned} Pr[Verify^{full}(r, m, c, c^\star, pp, op) = 1 \wedge Verify^{full} \\ (r, m', c, c^\star, pp, op') = 1 \wedge m \neq m' | (c, c^\star, op, m) \leftarrow \\ \mathcal{A}_0(r); (m', op') \leftarrow \mathcal{A}_1(r, state)] \leq negl(\lambda). \end{aligned}$$

Hiding. For all *ppt* adversaries $\mathcal{A}$, for a randomly sampled $(r, pp) \leftarrow \mathbf{Setup}(1^\lambda)$, and for all pairs of messages (m_0, m_1), we have that

$$Pr[\mathcal{A}(r, c, c^\star) = b | b \leftarrow 0, 1; (c, c^\star, aux) \leftarrow Commit(r, m, pp)] \leq \frac{1}{2} + negl(\lambda).$$

4.2 A Mirror Commitment based on CDH: $\mathbf{MC_{CDH}}$

Here we propose an implementation of concise mirror commitment $\mathbf{MC_{CDH}}$ for a single message based on the CDH assumption in multiplicative cyclic group G of order p proportional to the security parameter λ, where g is the generator. Precisely, the security of the scheme reduces to the Square Computational Diffie-Hellman assumption(see Definition 6 in Section 2.1), which has been shown equivalent to the standard CDH assumption $[2, 3]$(see Definition 4 in Section 2.1). Appendix B shows the mirror commitment scheme supporting multiple messages.

$\mathbf{Setup}(1^\lambda)$ Let $\mathbb{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathbb{G}$. Randomly choose $z_c, z_1, z_2 \leftarrow \mathbb{Z}_p$. Set $r = g^{z_c}$, $h_1 = g^{z_1}$, $h_2 = g^{z_2}$. Set $pp = (g, h_1, h_2)$. The message space is $\mathcal{M} = \mathbb{Z}_p$.

Commit(r, m, pp) Compute

$$c = h_1^m h_2^r,\ c^\star = h_1^r h_2^m$$

and output $C = (c, c^\star, aux)$ and the auxiliary information $aux = none$.
Open(m, r, pp) Compute

$$op_c = h_1^m,\ op_{c^\star} = h_2^m$$

and output $op = (op_c, op_{c^*})$.
$\textbf{Verify}^{\textbf{part}}(r, m, c|c^\star, pp, op_c|op_{c^\star})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } c = op_c * h_2^r \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c = op_{c^\star} * h_1^r \\ 0 & \text{otherwise} \end{cases}$$

and output $b = b_1 \vee b_2$.
$\textbf{Verify}^{\textbf{full}}(r, m, c, c^\star, pp, op_c, op_{c^\star})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } c = op_c * h_2^r \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c = op_{c^\star} * h_1^r \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2,\quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \Leftrightarrow c^\star \\ 0 & \text{otherwise} \end{cases}$$

and output $(b{,}t)$.
$\textbf{Update}^{\textbf{message}}(c, c^\star, m, m')$ Compute the updated commitment $c' = c * h_1^{m'-m}$ and dual commitment $c^{\star'} = c^\star * h_2^{m'-m}$. Finally output $C' = (c', c^{\star'})$ and $U = (m, m')$.
$\textbf{Update}^{\textbf{proof}}(c, c^\star, U, op)$ A client who owns a proof op, that is valid to c and $c^\star$ for the message m, can produce a new proof $op' = (op_c * h_1^{m'-m}, op_{c^\star} * h_2^{m'-m})$.

The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem.

Theorem 3. If the CDH assumption holds, then the scheme defined above is a concise dual commitment.

proof 3 We prove the theorem by showing that the scheme satisfies the binding property. For the sake of contradiction assume that there exists an efficient attacker $\mathcal{A}$ who produces two valid openings to two different messages, then we show how to build an efficient algorithm $\mathcal{B}$ to break the CDH assumption. First, $\mathcal{A}$ chooses $z_1, z_2, z_3 \leftarrow \mathbb{Z}_p$,it computes: $h_1 = g^{z_1}$, $h_2 = g^{z_2}, r = g^{z_3}$. $\mathcal{B}$ sets $pp = (g, h_1, h_2)$ and runs *Setup*. Notice that the public parameters are perfectly distributed as the real ones. The adversary is supposed to output a tuple $(c, c^\star, m, m', op_c, op_{c^\star}, op_c', op_{c^\star}^{'})$ such that $m \neq m'$ and both $op_c, op_{c^\star}$ and $op_c', op_{c^\star}^{'}$ correctly verify. Then $\mathcal{A}$ computes

$$h_1 = (op_c/op_c')^{m-m'^{-1}},\ h_2 = (op_{c^\star}/op_{c^\star}^{'})^{m-m'^{-1}}$$

To see that the output is correct, observe that since the two openings verify correctly, then it holds:

$$op_c * h_1^{m'} = op' * h_1^m,\ op_{c^\star} * h_2^{m'} = op'_{c^\star} * h_2^m$$

which means that

$$h_1^{m-m'} = op'_c/op_c,\ h_2^{m-m'} = op'_{c^\star}/op_{c^\star}$$

One can easily see that this justifies the correctness of $\mathcal{B}$'s output. Notice that if $\mathcal{A}$ succeeds with probability ϵ breaking the Square CDH assumption.

4.3 A Mirror Commitment based on RSA: $\mathbf{MC_{RSA}}$

Here we propose an implication of mirror commitment $\mathbf{MC_{RSA}}$ for a single message from the RSA assumption (whose definition is given in section 2.1). Appendix C shows the dual commitment scheme supporting multiple messages.
Setup$(1^\lambda, \ell)$ Randomly choose two $\ell/2$-bit primes p_1, p_2, set $N = p_1 p_2$, and then choose $2(\ell + 1)$-bit primes e_1, e_2, a, r that do not divide $\varphi(N)$. Compute,

$$S_1 = a^{e_2},\ S_2 = a^{e_1}$$

The public parameters pp are $(N, a, r, S_1, S_2, e_1, e_2)$. The message space is $M = \{0, 1\}^\ell$.
Commit(r, m, pp) Compute

$$c = S_1^m S_2^r = a^{e_2 m + e_1 r},\ c^\star = S_1^r S_2^m = a^{e_1 m + e_2 r}$$

and output $C = (c, c^\star, aux)$ and the auxiliary information $aux = none$.
Open(m, r, pp) Compute

$$op_c = S_1^{\frac{m}{e_2}}\ mod\ N,\ op_{c^\star} = S_2^{\frac{m}{e_1}}\ mod\ N$$

and output $op = (op_c, op_{c^\star})$. Notice that knowledge of pp allows to compute op_c efficiently without the factorization of N.
Verify$^{\mathbf{part}}(r, m, c|c^\star, pp, op_c|op_{c^\star})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } S_2^r op_c^{e_2}\ mod\ N = c \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } S_1^r op_{c^\star}^{e_1}\ mod\ N = c^\star \\ 0, & \text{otherwise} \end{cases}$$

and output $b = b_1 \vee b_2$.
Verify$^{\mathbf{full}}(r, m, c, c^*, pp, op_c, op_{c^*})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } S_2^r op_c^{e_2}\ mod\ N = c \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } S_1^r op_{c^\star}^{e_1}\ mod\ N = c^\star \\ 0, & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2,\quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \Leftrightarrow c^* \\ 0, & \text{otherwise} \end{cases}$$

and output (b,t).
Update$^{\textbf{message}}(c, c^\star, m, m')$ Compute the updated commitment $c' = c * S_1^{m'-m}$ and dual commitment $c^{\star'} = c^\star * S_2^{m'-m}$. Finally output $C' = (c', c^{\star'})$ and $U = (m, m')$.
Update$^{\textbf{proof}}(c, c^\star, U, op)$ A client who owns a proof op, that is valid to c and $c^\star$ for the message m, can produce a new proof $op' = (op_\star * S_1^{\frac{m-m'}{e_2}}, op_{c^\star} * S_2^{\frac{m-m'}{e_1}})$.

In order for the verification process to be correct, notice that one should also check that the S_1, S_2 are correctly generated with respect to a and the exponents e_1, e_2. The correctness of the scheme can be easily verified by inspection. We prove its security via the following theorem.

Theorem 4. If the RSA assumption holds, then the scheme defined above is a concise mirror commitment.

> **proof 4** We prove the theorem by showing that the scheme satisfies the binding property. More precisely, assume for the sake of contradiction that there exists an efficient adversary that produces two valid openings to two different messages at the same position, then we show how a *ppt* attacker $\mathcal{A}$ buids an algorithm $\mathcal{B}$ that breaks the RSA assumption. Firstly, $\mathcal{B}$ is run on input (N, z, e_1, e_2), where e is an $(\ell + 1)$-bit prime, and it is used to compute a value y such that $z_1 = y^{e_1} mod N, z_2 = y^{e_2} mod N$. The proceeds are as follows. First, it sets $a_1 = z_1, a_2 = z_2$. Then, it runs *Setup* and gets back $(S_1, S_2, m, m', op_c, op_{c^\star}, op', op'_{c^\star})$ where $m \neq m'$ and both $op_c, op_{c'}$ and $op_{c^\star}, op'_{c^\star}$ are correctly verified. From the equations $S_1^m op_c^{e_2} = S_1^{m'} {op'_c}^{e_2},\ S_2^m op_{c^\star}^{e_1} = S_2^{m'} {op'_{c^\star}}^{e_1}$ we get
>
> $$S_1^{m-m'} = op_c / {op'_c}^{e_2},\ S_2^{m-m'} = op_{c^\star} / {op'_{c^\star}}^{e_1}$$
>
> if $op_c/op'_c = 1$ *or* $op_{c^\star}/op'_{c^\star} = 1$ then we can factor with non-negligible probability. Thus, assuming $op_c/op'_c \neq 1$ *and* $op_{c^\star}/op'_{c^\star} \neq 1$ we can apply Shamir's trick [4] to get an $e_1 - th$ root of a_1, a_2. In particular, since $gcd(me_1, e_2) = 1$, by the extended Euclidean Algorithm, we can compute two integers λ, μ such that $m\lambda e_1 + \mu e_2 = 1$. This leads to the equation
>
> $$a_1 = (op_c/op'_c)^{\lambda e_2} a^{\mu e_1},\ a_2 = (op_{c^\star}/op'_{c^\star})^{\lambda e_2} a^{\mu e_1}$$
>
> thus $(op_c/op'_c)^{\lambda e_2} a^{\mu e_1}$ *and* $(op_{c^\star}/op'_{c^\star})^{\lambda e_2} a^{\mu e_1}$ is the required corresponding root.

5 Features on Mirror Commitment and Dual Commitment

We next discuss some important features of Mirror Commitment and Dual Commitment.

Theorem 5. Homomorphism. DC$_{\textbf{CDH}}$ and **MC$_{\textbf{CDH}}$** are both (additive) homomorphic in nature.

proof 5 Observe that given $\mathbf{DC_{CDH}}$ commitments $C_1 = (c_1, c_1^*)$ and $C_2 = (c_2, c_2^*)$ associated with message pairs $\langle m_1, m_2\rangle$ respectively, one can compute the commitment $C = (c, c^*)$ for $m = m_1 + m_2$ as $C = (c_1 * c_2, c_1^* + c_2^*)$.
proof 6 Observe that given $\mathbf{MC_{CDH}}$ commitments $C_1 = (c_1, c_1^*)$ and $C_2 = (c_2, c_2^*)$ associated with message pairs $\langle m_1, m_2\rangle$ respectively, one can compute the commitment $C = (c, c^*)$ for $m = m_1 + m_2$ as $C = (c_1 * c_2, c_1^* * c_2^*)$.

Theorem 6. Standard Security Properties $\mathbf{DC_{RSA}}$ and $\mathbf{MC_{RSA}}$ are computationally hiding under the RSA assumption. $\mathbf{DC_{CDH}}$ and $\mathbf{MC_{CDH}}$ are statistically binding.

proof 7 The construction is computationally hiding under the RSA assumption, because the commitment algorithm is identical to the one for ElGamal commitments. For binding, pedersen commitments are computationally binding under the CDH assumption. We refer the reader to ElGamal [6] and Pedersen [7] for detailed discussions.

Theorem 7. Trapdoor Commitment. $\mathbf{DC_{CDH}}, \mathbf{MC_{CDH}}, \mathbf{DC_{RSA}}, \mathbf{MC_{RSA}}$ are also trapdoor commitment schemes, where $r = g^{z_c}$ is the trapdoor.

proof 8 For $\mathbf{DC_{CDH}}$, given r, a simulator can create witnesses for arbitrary values with respect to $C = h_1^m h_2^r$ for an unknown r. To "prove" m (where m is the message supposedly committed to by C), output op. It can easily be checked that $\mathbf{Verify^{part}}(C, m, op, pp) = 1$ and $\mathbf{Verify^{full}}(C, m, op, pp) = 1$ The same also applies to $\mathbf{MC_{CDH}}$, $\mathbf{DC_{RSA}}$, $\mathbf{MC_{RSA}}$.

6 Two-way Zero-knowledge Authentication Protocols

In this section, we describe the applications of our commitment schemes to construct bidirectional non-interactive zero-knowledge authentication protocols. A Prover($\mathcal{P}$) can convince a Verifier ($\mathcal{V}$) that it is a legal user by proving c^* is the duality of c or $c^\star$ is the mirror of c without revealing any private information and they can still continue to authenticate without another initialization even after changing roles. Here, we give the instance of constructing non-interactive and interactive bidirectional zero-knowledge authentication protocols through $\mathbf{DC_{RSA}}$ and $\mathbf{MC_{RSA}}$. However, $\mathbf{DC_{CDH}}$ is an one-way dual commitment so it can't be used to build a bidirectional authentication protocol, but, it can be used to build a tone-way authentication protocol. The rest 2 instances of constructing zero-knowledge authentication protocol through $\mathbf{DC_{CDH}}$, $\mathbf{MC_{CDH}}$ are similar to $\mathbf{DC_{RSA}}$ and $\mathbf{MC_{RSA}}$. They are shown in Appendix D.
zero-knowledge authentication for $\mathbf{DC_{RSA}}$
Let p_1, p_2 are two $\ell/2$-bit primes , set $N = p_1 p_2$. Let e_1, e_2, a, r are three $2(\ell+1)$-bit primes that do not divide $\varphi(N)$. Let $S_1 = a^{e_2}, S_2 = a^{e_1}, m \in \{0,1\}^\ell$. The protocol presented in algorithm 1 as a sigma protocol for the relation $\mathbb{R}_1$.

proof 9 Completeness follows trivially by inspection. We next show that the protocol is 2-special sound by a standard rewinding argument, where

Algorithm 1 Interactive protocol for $\mathbf{DC_{RSA}}$
$\mathcal{R}_1 = c, c^* \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = a^{e_2 m + e_1 r}, c^* = a^{(e_2+m)(e_1+r)}$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$ *and computes* :
$R_0 = a^{r_0(e_1+e_2)}, R_1 = a^{r_1(e_1-e_2)}$
$\mathbb{P} \to \mathbb{V}$: R_0, R_1
$\mathbb{V} \leftarrow \mathbb{P}$: $t \xleftarrow{\$} \mathbb{Z}_l$
$\mathbb{P}$: computes:
$y_0 = a^{r_0+t(m+r)}, y_1 = a^{r_1+t(r-m)}$
$\mathbb{P} \to \mathbb{V}$: y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$y_0^{(e_1+e_2)} * y_1^{(e_1-e_2)} / c^{2t} \stackrel{?}{=} R_0 * R_1$ (1)

we define an extractor that produces valid witness elements on accepting transcripts using distinct verifier challenges. Fix an initial transcript $(c, c^*, \mathcal{R}_0, \mathcal{R}_1)$, and let $c \neq c'$ be distinct verifier challenges for this transcript, with corresponding responses (y_0, y_1) and (y_0', y_1'). We apply Equations (1) to these transcripts to obtain

$$(\frac{y_0}{y_0'})^{(e_1+e_2)}(\frac{y_1}{y_1'})^{(e_1-e_2)} = c^{2(c-c')}(2)$$

and hence

$$(\frac{y_0}{y_0'})^{\frac{(e_1+e_2)}{(c-c')}}(\frac{y_1}{y_1'})^{\frac{(e_1-e_2)}{(c-c')}} = c^2(3)$$

Define $\alpha_0 = (\frac{y_0}{y_0'})^{\frac{1}{(c-c')}}$ and $\alpha_1 = (\frac{y_1}{y_1'})^{\frac{1}{(c-c')}}$, and note that both are well-defined since $c \neq c'$. According to Equation (3) and definition, we obtain the following expressions for c and c^*

$$c = (\alpha_0 * \alpha_1)^{\frac{e_1}{2}} * (\frac{\alpha_0}{\alpha_1})^{\frac{e_2}{2}} = a^{e_2 m + e_1 r}$$

$$c^* = (\frac{\alpha_0}{\alpha_1})^{\frac{e_1}{2}} * (\alpha_0 * \alpha_1)^{\frac{e_1}{2}} * a^{e_1 e_2 + m * r} = a^{(e_2+m)(e_1+r)}$$

We finally show that the protocol is a special honest-verifier zero-knowledge. To do so, we define a simulator that, on a valid statement and uniformly sampled verifier challenge, produces transcripts indistinguishable from those of real proofs. Fix a valid prover statement (c, c^*) and sample a nonzero challenge $c \in \mathbb{Z}_l$. The simulator samples random $y_0, y_1 \in \mathbb{Z}_l$ and defines R_0, R_1 using Equations (1), respectively. The resulting simulated proof will be accepted by an honest verifier. Because e_1, e_2 are different primes, such a simulated proof is distributed identically to a real proof, and hence the protocol is special honest-verifier zero knowledge. This completes the proof.

This protocol may be made non-interactive via the Fiat-Shamir [33] technique, where the verifier challenge is replaced by a suitable transcript hash. This technique further allows for binding an arbitrary proof context into the transcript. Algorithm 2 shows an example non-interactive protocol.

Algorithm 2 Non-interactive protocol for $\mathbf{DC_{RSA}}$
$\mathcal{R}_1 = c, c^* \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = a^{e_2 m + e_1 r}, c^* = a^{(e_2+m)(e_1+r)}$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l \quad and \quad computes:$
$R_0 = a^{r_0(e_1+e_2)}, R_1 = a^{r_1(e_1-e_2)}$
$t = H_s(R_0 * R_1, c, c^*)$ *where* $H_s(*)$ *is a hash function*
$y_0 = a^{r_0+t(m+r)}, y_1 = a^{r_1+t(r-m)}$
$\mathbb{P} \to \mathbb{V}$: t, y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$H_s(y_0^{(e_1+e_2)} * y_1^{(e_1-e_2)}/c^{2t}, c, c^*) \stackrel{?}{=} t$

Theorem 8 Both non-interactive and interactive zero-knowledge authentication protocols for $\mathbf{DC_{RSA}}$ can be used to realize bidirectional authentication with only an initialization.

proof 10 According to definition 8 in Section 3.4, we know c^* is the dual of c, the reverse is also true, so Prover $\mathcal{P}$ and Verifier $\mathcal{V}$ are allowed exchange roles so that they can realize bidirectional authentication with only an initialization without additional initialization.

zero-knowledge authentication for $\mathbf{MC_{RSA}}$

Let p_1, p_2 are two $\ell/2$-bit primes , set $N = p_1 p_2$. Let e_1, e_2, a, r are three $2(\ell+1)$-bit primes that do not divide $\varphi(N)$. Let $S_1 = a^{e_2}, S_2 = a^{e_1}, m \in \{0,1\}^\ell$. The protocol presented in algorithm 3 as a sigma protocol for the relation $\mathbb{R}_2$.

Algorithm 3 Interactive protocol for $\mathbf{MC_{RSA}}$
$\mathcal{R}_1 = c, c^\star \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = a^{e_2 m + e_1 r}, c^\star = a^{e_2 r + e_1 m}$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$ and computes:
$R_0 = a^{r_0(e_1+e_2)}, R_1 = a^{r_1(e_1-e_2)}$
$\mathbb{P} \to \mathbb{V}$: R_0, R_1
$\mathbb{V} \leftarrow \mathbb{P}$: $t \xleftarrow{\$} \mathbb{Z}_l$
$\mathbb{P}$: computes:
$y_0 = a^{r_0+t(m+r)}, y_1 = a^{r_1+t(m-r)}$
$\mathbb{P} \to \mathbb{V}$: y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$y_0^{(e_1+e_2)}/(c * c^\star)^t \stackrel{?}{=} R_0$ (4), $y_1^{(e_1-e_2)}/(c/c^\star)^t \stackrel{?}{=} R_1$ (5)

proof 11 Completeness follows trivially by inspection. We next show that the protocol is 2-special sound by a standard rewinding argument, where we define an extractor that produces valid witness elements on accepting transcripts using distinct verifier challenges. Fix an initial transcript $(c, c^\star, \mathcal{R}_0, \mathcal{R}_1)$, and let $c \neq c'$ be distinct verifier challenges for this transcript, with corresponding responses (y_0, y_1) and (y'_0, y'_1). We apply Equations (4) and (5) to these transcripts to obtain

$$(\frac{y_0}{y'_0})^{(e_1+e_2)} = (c * c^{\star})^{c-c'} (6)$$

$$(\frac{y_1}{y'_1})^{(e_1-e_2)} = (c * c^{\star})^{c-c'} (7)$$

and hence

$$(\frac{y_0}{y'_0})^{\frac{(e_1+e_2)}{(c-c')}} = c * c^{\star} (8)$$

$$(\frac{y_1}{y'_1})^{\frac{(e_1-e_2)}{(c-c')}} = c / c^{\star} (9)$$

Define $\alpha_0 = (\frac{y_0}{y'_0})^{\frac{1}{(c-c')}}$ and $\alpha_1 = (\frac{y_1}{y'_1})^{\frac{1}{(c-c')}}$, and note that both are well-defined since $c \neq c'$. According Equations (6) and (7), we obtain the following expressions for c and $c^{\star}$

$$c = (\alpha_0 * \alpha_1)^{\frac{e_1}{2}} * (\frac{\alpha_0}{\alpha_1)})^{\frac{e_2}{2}} = a^{e_2 m + e_1 r}$$

$$c^{\star} = (\frac{\alpha_0}{\alpha_1)})^{\frac{e_1}{2}} * (\alpha_0 * \alpha_1)^{\frac{e_1}{2}} = a^{(e_2 r)(e_1 m)}$$

We finally show that the protocol is a special honest-verifier zero knowledge. To do so, we define a simulator that, on a valid statement and uniformly sampled verifier challenge, produces transcripts indistinguishable from those of real proofs. Fix a valid prover statement $(c, c^{\star})$ and sample a nonzero challenge $c \in \mathbb{Z}_l$. The simulator samples random $y_0, y_1 \in \mathbb{Z}_l$ and defines R_0, R_1 using Equations (4) and (5), respectively. The resulting simulated proof will be accepted by an honest verifier. Because e_1, e_2 are different primes, such a simulated proof is distributed identically over a real proof, and hence the protocol is special honest-verifier zero knowledge. This completes the proof.

This protocol may be made non-interactive via the Fiat-Shamir [33] technique, where the verifier challenge is replaced by a suitable transcript hash. This technique further allows for binding an arbitrary proof context into the transcript. Algorithm 4 shows an example non-interactive protocol.

Theorem 9 Both non-interactive and interactive zero-knowledge authentication protocols for $\mathbf{MC_{RSA}}$ can be used to realize bidirectional authentication with only an initialization.

proof 12 According to definition 10 in Section 3.4, we know $c^{\star}$ is the mirror of c, the reverse is also true, so Prover $\mathcal{P}$ and Verifier $\mathcal{V}$ are allowed exchange roles so that they can realize bidirectional authentication with only an initialization without additional initialization.

Algorithm 4 Non-interactive protocol for $\mathbf{MC_{RSA}}$
$\mathcal{R}_1 = c, c^\star \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = a^{e_2 m + e_1 r}, c^\star = a^{e_2 r + e_1 m}$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$ and computes:
$R_0 = a^{r_0(e_1+e_2)}, R_1 = a^{r_1(e_1-e_2)}$
$t = H_s(R_0, R_1, c, c^\star)$ *where* $H_s(*)$ *is a hash function*
$y_0 = a^{r_0+t(m+r)}, y_1 = a^{r_1+t(m-r)}$
$\mathbb{P} \to \mathbb{V}$: t, y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$H_s(y_0^{(e_1+e_2)}/(c * c^\star)^t, y_1^{(e_1-e_2)}/(c/c^\star)^t, c, c^*) \stackrel{?}{=} t$

7 Open Problems

Finally, we list a few open problems related to the commitment and mirror commitment schemes. 1. Is it possible to construct efficient polynomial commitment schemes under weaker assumptions? 2. What other protocols do dual commitment and mirror commitment improves? (For example, can commitment and mirror commitment reduce communication of asynchronous VSS protocols or verifiable shuffles? See the protocol of Groth and Ishai [5]) 3. We have mainly focused on communication costs, but our construction asks for nontrivial computation. Is it possible to reduce computation costs as well? 4. Whether the dual commitment and mirror commitment can be used to construct the oblivious transfer protocols?

Acknowledgments

National Natural Science Foundation of China(No.62132018)
National Key Project of China(No.2020YFB1005700)
The National Key Research and Development Program of China(No.2021YFA100 0600)
The National Key Research and Development Program of Guangdong Province Under Grant(No.2020B0101090002)
National Key Research and Development (R&D) Program, Young Scientist Scheme (No.2022YFB3102400)

Declarations

Ethical Approval not applicable
Competing interests not applicable
Authors' contributions
Xingyu.Li. designed the scheme and wrote the main manuscript text
Junyang.Li. assisted in writing manuscripts
Guoyu.Yang and Qi.Chen. Hongyang.Yan modified the text
Jin Li is the director

Conflict of interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Torben P. Pedersen. 1991. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO'91). Springer-Verlag, Berlin, Heidelberg, 129–140.
2. Feng Bao, Robert Deng, and HuaFei Zhu. Variations of diffie-hellman problem. In Sihan Qing, Dieter Gollmann, and Jianying Zhou, editors, Information and Communications Security, volume 2836 of Lecture Notes in Computer Science, pages 301–312. Springer Berlin / Heidelberg, 2003.
3. Ueli M. Maurer and Stefan Wolf. Diffie-Hellman oracles. In Neal Koblitz, editor, CRYPTO'96, volume 1109 of LNCS, pages 268–282. Springer, August 1996.
4. Adi Shamir. On the generation of cryptographically strong pseudorandom sequences. ACM Trans. Comput. Syst., 1(1):38–44, 1983.
5. J. Groth and Y. Ishai. Sub-linear zero-knowledge argument for correctness of a shuffle. In Proceedings of EUROCRYPT'08, volume 4965 of LNCS, pages 379–396. Springer, 2008.
6. Elgamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: CRYPTO'84. Springer
7. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: CRYPTO'91. Springer
8. Goldwasser, S., S. Micali, and C. Rackoff, The Knowledge Complexity of Interactive Proof Systems, in 17th Annual Symposium on Theory Of Computing (STOC), 1985.
9. SAHAI A. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security, In: Pro- ceedings of 40th Annual Symposium on Foundations of Computer Science (FOCS 1999). IEEE, 1999: 543–553.
10. Arun, A., Bonneau, J., Clark, J. (2022). Short-lived Zero-Knowledge Proofs and Signatures. In: Agrawal, S., Lin, D. (eds) Advances in Cryptology – ASIACRYPT 2022. ASIACRYPT 2022. Lecture Notes in Computer Science, vol 13793. Springer, Cham.
11. Du, R., Li, X., Liu, Y. (2022). A Cross-domain Authentication Scheme Based on Zero-Knowledge Proof. In: Lai, Y., Wang, T., Jiang, M., Xu, G., Liang, W., Castiglione, A. (eds) Algorithms and Architectures for Parallel Processing. ICA3PP 2021. Lecture Notes in Computer Science(), vol 13156. Springer, Cham.
12. MA S L, DENG Y, HE D B, et al. An efficient NIZK scheme for privacy-preserving transactions over account- model blockchain. IEEE Transactions on Dependable Secure Computing, 2021, 18(2): 641–65
13. CHASE M, GANESH C, MOHASSEL P. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials. In: Advances in Cryptology-CRYPTO 2016, Part III. Springer Berlin Heidelberg, 2016: 49
14. BEN-SASSON E, CHIESA A, GARMAN C, et al. Zerocash: Decentralized anonymous payments from bitcoin. In: Proceedings of 2014 IEEE Symposium on Security and Privacy (S&P 2014). IEEE, 2014: 459–4

15. DANEZIS G, FOURNET C, KOHLWEISS M, et al. Pinocchio coin: Building zerocoin from a succinct pairing- based proof system. In: Proceedings of the First ACM Workshop on Language Support for Privacy-enhancing Technologies (PETShop 2013). ACM, 2013: 27
16. GABIZON A, WILLIAMSON Z J, CIOBOTARU O. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. IACR Cryptology ePrint Archive, 20
17. ZHANG Z F, YANG K, HU X X, et al. Practical anonymous password authentication and TLS with anonymous client authentication[C]. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communica- tions Security (CCS 2016). ACM, 2016: 11791
18. LI Z P, WANG D, MORAIS E. Quantum-safe round-optimal password authentication for mobile devices[J]. IEEE Transactions on Dependable and Secure Computing, 2022, 19(3): 1885-1899.
19. WANG Q X, WANG D, CHENG C, et al. Quantum2FA: Efficient quantum-resistant two-factor authentication scheme for mobile devices[J]. IEEE Transactions on Dependable and Secure Computing, 2021: 1–1.
20. Kumar, S.B., Mandal, R.K., Mukherjee, K., Dwivedi, R.K. (2022). Security of Cloud Computing Using Quantum Zero-Knowledge Proof System. In: Dahal, K., Giri, D., Neogy, S., Dutta, S., Kumar, S. (eds) Internet of Things and Its Applications. Lecture Notes in Electrical Engineering, vol 825. Springer, Singapore.
21. Feneuil, T., Maire, J., Rivain, M., Vergnaud, D. (2022). Zero-Knowledge Protocols for the Subset Sum Problem from MPC-in-the-Head with Rejection. In: Agrawal, S., Lin, D. (eds) Advances in Cryptology – ASIACRYPT 2022. ASIACRYPT 2022. Lecture Notes in Computer Science, vol 13792. Springer, Cham.
22. Hazay, C., Venkitasubramaniam, M. & Weiss, M. ZK-PCPs from Leakage-Resilient Secret Sharing. J Cryptol 35, 23 (2022).
23. Huang, J., Huang, T., Zhang, J. (2023). zkChain: An Efficient Blockchain Privacy Protection Scheme Based on zero-knowledge-SNARKs. In: Xu, Y., Yan, H., Teng, H., Cai, J., Li, J. (eds) Machine Learning for Cyber Security. ML4CS 2022. Lecture Notes in Computer Science, vol 13656. Springer, Cham.
24. XIE T C, ZHANG J H, ZHANG Y P, et al. Libra: Succinct zero-knowledge proofs with optimal prover computation. In: Advances in Cryptology-CRYPTO 2019, Part III. Springer Cham, 2019: 733–764.
25. SETTY S. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In: Advances in Cryptology-CRYPTO 2020, Part III. Springer Cham, 2020: 704–737.
26. ZHANG J H, XIE T C, ZHANG Y P, et al. Transparent polynomial delegation and its applications to zero knowledge proof[C]. In: Proceedings of 2020 IEEE Symposium on Security and Privacy (S&P 2020). IEEE, 2020: 859–876.
27. ZHANG J H, LIU T Y, WANG W, et al. Doubly efficient interactive proofs for general arithmetic circuits with linear prover time[C]. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS 2021). ACM, 2021: 159–177
28. BOOTLE J, CERULLI A, CHAIDOS P, et al. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: Advances in Cryptology-EUROCRYPT 2016, Part II. Springer Berlin Heidelberg, 2016: 327–35
29. WAHBY R S, TZIALLA I, SHELAT A, et al. Doubly-efficient zkSNARKs without trusted setup. In: Proceedings of 2018 IEEE Symposium on Security and Privacy (S&P 2018). IEEE, 2018: 926–943.

30. GIACOMELLI I, MADSEN J, ORLANDI C. ZKBoo: Faster zero-knowledge for Boolean circuits. In: Pro- ceedings of the 25th USENIX Conference on Security Symposium (SEC'16). USENIX, 2016: 1069-1083.
31. GVILI Y, SCHEFFLER S, VARIA M. BooLigero: Improved sublinear zero knowledge proofs for Boolean cir- cuits. In: Financial Cryptography and Data Security-FC 2021, Part I. Springer Berlin Heidelberg, 2021: 476–496.
32. Cui, H., Zhang, K. (2021). A Simple Post-Quantum Non-interactive Zero-Knowledge Proof from Garbled Circuits. In: Yu, Y., Yung, M. (eds) Information Security and Cryptology. Inscrypt 2021. Lecture Notes in Computer Science(), vol 13007. Springer, Cham.
33. Fiat A, Shamir A.How to prove yourself:Practical solutions to identifcation and signature problems(CI/Advances in Cryptology-CRYPTO "86.Berlin:Springer,1987:186-194

Appendix

A $\mathbf{DC_{RSA}}$ for multiple messages

Setup$(1^\lambda, \ell, q)$ Randomly choose two $\ell/2$-bit primes p_1, p_2, set $N = p_1p_2$, and then choose $2(\ell+1)$-bit primes $e_1, ..., e_q, a$ that do not divide $\varphi(N)$. For $i = 1$ to q Compute,

$$S_i = a^{\prod_{j=1,j\neq i}^{q} e_j}$$

The public parameters pp are $(N, a, S_1, ..., S_q, e_1, ...e_q)$. The message space is $M = \{0,1\}^\ell$.

Commit$(m_1, ...m_q, pp)$ Compute

$$c = S_1^m ... S_q^{m_q} = a^{\sum_{i=1}^{q}(m_i \prod_{j=1,j\neq i}^{q} e_j)},\ c^* = a^{\prod_{i=1}^{q}(m_i + \sum_{j=1,j\neq i}^{q} e_j)}$$

and output $C = (c, c^*, aux)$ and the auxiliary information $aux = none$.

Open(m, i, pp) Compute

$$op_c^i = (\prod_{j=1,j\neq i}^{q} S_j^{m_j})^{\frac{1}{e_i}}\ mod\ N,\ op_{c^*}^i = (S_i^{m_i})^{\frac{1}{e_i}}(\prod_{j=1,j\neq i}^{q} S_j^{m_j})^{\frac{1}{e_i}}\ mod\ N$$

and output $op = (op_c^i, op_{c^*}^i)$. Notice that knowledge of pp allows to compute op_c^i efficiently without the factorization of N.

Verify$^{\mathbf{part}}(m, i, c|c^*, pp, op_c^i|op_{c^*}^i)$ Compute

$$b_1 = \begin{cases} 1, & \text{if } \prod_{i=1}^{q} S_i(op_{c^*}^i)^{e_i} a^{\prod_{i=1}^{q} m_i}\ mod\ N = op_{c^*} \\ 0 & \text{otherwise} \end{cases}$$

$$b_2 = \begin{cases} 1, & \text{if } \prod_{j=1,j\neq i}^{q} S_j^{m_j}(op_c^i)^{e_i}\ mod\ N = op_c \\ 0 & \text{otherwise} \end{cases}$$

and output $b = b_1 \vee b_2$.

Verify$^{\textbf{full}}$ $(r, m, c, c^*, pp, op_c^i, op_{c^*}^i)$ Compute

$$b_1 = \begin{cases} 1, & \text{if } \prod_{i=1}^{q} S_i (op_{c^*}^i)^{e_i} a^{\prod_{i=1}^{q} m_i} \ mod\ N = op_{c^*} \\ 0 & \text{otherwise} \end{cases}$$

$$b_2 = \begin{cases} 1, & \text{if } \prod_{j=1, j \neq i}^{q} S_j^{m_j} (op_c^i)^{e_i} \ mod\ N = op_c \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2,\ t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \lhd \rhd c^* \\ 0, & \text{if } b = 1 \text{ and } c \rhd c^* \\ -1 & \text{otherwise} \end{cases}$$

and output (b,t).

Update$^{\textbf{message}}$ (c, c^*, m, m', i) Compute the updated commitment $c' = c{*}S_i^{m'-m}$ and dual commitment $c^{*'} = c^* * a^{(e_i + \sum_{j=1,j\neq i}^{q})(m'-m)}$. Finally output $C' = (c', c^{*'})$ and $U = (m, m', i)$.

Update$^{\textbf{proof}}$ $(c, c^*, U, i, op_c^j, op_{c^*}^j)$ A client who owns a proof $op_c^j, op_{c^*}^j$, that is valid to c and c^* for some message at position j, can use the update information U to compute the updated commitment $c', c^{*'}$ and to produce a new proof $op_c^{j'}, op_{c^*}^{j'}$ which will be valid $c', c^{*'}$. We distinguish two cases:

1. $i \neq j$. Compute the updated commitment as $c' = comS_i^{m'-m}, c^{*'} = c^* * a^{(e_i + \sum_{j=1,j\neq i}^{q})(m'-m)}$ while the updated proof is $op_c^{j'} = op_c^j * (S_i^{\frac{m-m'}{e_j}}), op_{c^*}^{j'} = op_{c^*}^j * S_i^{\frac{m-m'}{e_j}}$ (notice that such e_{j-th} root can be efficiently computed using the elements in the public key).

2. $i = j$. Compute the updated commitment while the updated proof remains the same $op_c^i, op_{c^*}^i$

In order for the verification process to be correct, notice that one should also verify (only once) the validity of the public key by checking that the S_i's are correctly generated with respect to a and the exponents $e1, .., e_q$.

The correctness of the scheme can be easily verified by inspection. We prove its security via the proof method similar to theorem 2.

B MC$_{\textbf{CDH}}$ for multiple messages

Setup$(1^\lambda, q)$ Let $\mathbb{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ and let g be a generator of $\mathbb{G}$. Randomly choose $z_1, ..., z_q \leftarrow \mathbb{Z}_p$. Set $h_i = g^{z_i}$ for all $i = 1, 2, ..., q$. Set $pp = (g, h_1, ..., h_q)$. The message space is $\mathcal{M} = \mathbb{Z}_p$.

Commit$(m_1, .., m_q, pp)$ Compute

$$c = \prod_{i=1}^{q} h_i^{m_i},\ c^\star = \prod_{i=1}^{q} h_i^{(m_{q-i+1})}$$

and output $C = (c, c^\star, aux)$ and the auxiliary information $aux = (m_1, ...m_q)$.
Open(m_i, i, pp) Compute

$$op_c^i = \prod_{j=1, j\neq i}^{q} h_j^{m_j},\ op_{c^\star}^i = \prod_{j=1, j\neq q-i+1}^{q} h_j^{m_j}$$

and output $op = (op_c, op_{c^\star})$.
Verify$^{\mathbf{part}}(m, i, c|c^\star, pp, op_c^i|op_{c^\star}^i)$ Compute

$$b_1 = \begin{cases} 1, & \text{if } c/h_i^{m_i} = op_c^i \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c/h_{q-i+1}^{m_{q-i+1}} = op_{c^\star}^i \\ 0 & \text{otherwise} \end{cases}$$

and output b.
Verify$^{\mathbf{full}}(m, i, c, c^\star, pp, op_c^i, op_{c^\star}^i)$ Compute

$$b_1 = \begin{cases} 1, & \text{if } c/h_i^{m_i} = op_c^i \\ 0 & \text{otherwise} \end{cases},\quad b_2 = \begin{cases} 1, & \text{if } c/h_{q-i+1}^{m_{q-i+1}} = op_{c^\star}^i \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2,\quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \Leftrightarrow c^\star \\ 0 & \text{otherwise} \end{cases}$$

and output (b,t).
Update$^{\mathbf{message}}(c, c^\star, m, m', i)$ Compute the updated commitment $c' = c * h_i^{m'-m}$ and dual commitment $c^{\star'} = c^\star * h_{q-i+1}^{m'-m}$. Finally output $C' = (c', c^{\star'})$ and $U = (m, m', i)$.
Update$^{\mathbf{proof}}(c, c^\star, U, op_c^j, op_{c^\star}^j)$ A client who owns a proof $op_c^j, op_{c^\star}^j$, that is valid to c and $c^\star$ for the message m at position j, can use the update information $U = (m, m', i)$ to compute the updated commitment $c', c^{\star'}$ and produce a new proof $op_c^{j'}, op_{c^\star}^{j'}$ which will be valid w.r.t.$c', c^{\star'}$. We distinguish two cases: 1. $i \neq j$. Compute the updated commitment $c' = c * h_i^{m'-m}, c^{\star'} = c^\star * h_{q-i+1}^{m-m'}$ while the updated proof is $op_c^{j'} = op_c^j * h_i^{m'-m}, op_{c^\star}^{j'} = op_{c^\star}^{q-j+1} * h_{q-i+1}^{m'-m}$.

2. $i = j$. Compute the updated commitment as $c' = c * h_i^{m'-m}, c^{\star'} = c^\star * h_{q-i+1}^{m-m'}$ while the updated proof remains the same $op_c^i, op_{c^\star}^i$.

The correctness of the scheme can be easily verified by inspection. We prove its security via the proof method similar to theorem 3.

C MC$_{\mathbf{RSA}}$ for multiple messages

Setup$(1^\lambda, \ell, q)$ Randomly choose two $\ell/2$-bit primes p_1, p_2, set $N = p_1p_2$, and then choose $2(\ell + 1)$-bit primes $e_1, ..., e_q, a$ that do not divide $\varphi(N)$. For $i = 1$ to q Compute,

$$S_i = a^{\prod_{j=1, j\neq i}^{q} e_j}$$

The public parameters pp are $(N, a, S_1, ..., S_q, e_1, ...e_q)$. The message space is $M = \{0,1\}^\ell$.

Commit$(m_1, ...m_q, pp)$ Compute

$$c = \prod_{i=1}^{q} S_i^{m_i} = a^{\sum_{i=1}^{q}(m_i \prod_{j=1,j\neq i}^{q} e_j)},\ c^\star = \prod_{i=1}^{q} S_i^{m_{q-i+1}} = a^{\sum_{i=1}^{q}(m_{q-i+1} \prod_{j=1,j\neq q-i+1}^{q} e_j)}$$

and output $C = (c, c^\star, aux)$ and the auxiliary information $aux = (m_1, ...m_q)$.

Open(m, i, pp) Compute

$$op_c = \prod_{j=1,i\neq j}^{q} (S_j^{m_j})^{e_i}\ mod\ N,\ op_{c^\star} = \prod_{j=1,j\neq q-i+1}^{q} S_j^{\frac{m_j}{e_{q-i+1}}}\ mod\ N$$

and output $op = (op_c, op_{c^\star})$. Notice that knowledge of pp allows to compute op_c efficiently without the factorization of N.

Verify$^{\textbf{part}}(m, i, c|c^\star, pp, op_c^i|op_{c^\star}^i)$ Compute

$$b_1 = \begin{cases} 1, & \text{if } (S_i^m (op_c^i)^{e_i}\ mod\ N = c \\ 0 & \text{otherwise} \end{cases},\ b_2 = \begin{cases} 1, & \text{if } (S_{q-i+1}^{m_{q-i+1}} (op_{c^\star}^{q-i+1})^{e_{q-i+1}}\ mod\ N = c^\star \\ 0 & \text{otherwise} \end{cases}$$

and output $b = b_1 \vee b_2$.

Verify$^{\textbf{full}}(r, m, c, c^\star, pp, op_c, op_{c^\star})$ Compute

$$b_1 = \begin{cases} 1, & \text{if } (S_i^m (op_c^i)^{e_i}\ mod\ N = c \\ 0 & \text{otherwise} \end{cases},\ b_2 = \begin{cases} 1, & \text{if } (S_{q-i+1}^{m_{q-i+1}} (op_{c^\star}^{q-i+1})^{e_{q-i+1}}\ mod\ N = c^\star \\ 0 & \text{otherwise} \end{cases}$$

$$b = b_1 \wedge b_2, \quad t = \begin{cases} 1, & \text{if } b = 1 \text{ and } c \Leftrightarrow c^\star \\ 0 & \text{otherwise} \end{cases}$$

and output (b,t).

Update$^{\textbf{message}}(c, c^\star, m, m', i)$ Compute the updated commitment $c' = c * S_i^{m'-m}$ and dual commitment $c^{\star'} = c^\star * S_{q-i+1}^{m'_{q-i+1} - m_{q-i+1}}$. Finally output $C' = (c', c^{\star'})$ and $U = (m, m', i)$.

Update$^{\textbf{proof}}(c, c^\star, i, U, op_c^j, op_{c^\star}^j)$ A client who owns a proof $op_c^j, op_{c^\star}^j$, that is valid to c and $c^\star$ for some message at position j, can use the update information U to compute the updated commitment $c', c^{\star'}$ and to produce a new proof $op_c^{j'}, op_{c^\star}^{j'}$ which will be valid $c', c^{\star'}$. We distinguish two cases:

1. $i \neq j$. Compute the updated commitment as $c' = c * S_i^{m'-m}, c^{\star'} = c^\star * S_{q-i+1}^{m'-m}$ while the updated proof is $op_c^{j'} = op_c^j * S_i^{\frac{m-m'}{e_j}}, op_{c^\star}^{j'} = op_{c^\star}^j * S_{q-i+1}^{\frac{m-m'}{e_{q-j+1}}}$ (notice that such e_{j-th} root can be efficiently computed using the elements in the public key).

2. $i = j$. Compute the updated commitment while the updated proof remains the same $op_c^i, op_{c^\star}^i$

In order for the verification process to be correct, notice that one should also verify (only once) the validity of the public key by checking that the S_i's are correctly generated with respect to a and the exponents $e_1, .., e_q$. The correctness of the scheme can be easily verified by inspection. We prove its security via the proof method similar to theorem 4.

D zero-knowledge authentication through $\mathbf{DC_{CDH}}$ and $\mathbf{MC_{CDH}}$

Let $\mathbb{G}$ be a multiplicative cyclic group of order p proportional to the security parameter λ where the discrete logarithm problem is hard. Let $\mathbb{F}_l$ be its scalar field and g be a generator of $\mathbb{G}$. Let $0 \neq \mathbb{G}, h_1, h_2 \in \mathbb{G}$ be group elements with no efficiently-computable discrete logarithm relation. We assume that $\mathbb{G}, G, H$ are implicit public parameters where needed. The interactive zero-knowledge authentication protocol presented in algorithm 5 is complete, special sound, and special honest-verifier zero knowledge as a sigma protocol for the relation R3. The corresponding non-interactive protocol is shown in algorithm 6

Algorithm 5 Interactive protocol for $\mathbf{DC_{CDH}}$
$\mathcal{R}_1 = c, c^* \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = h_1^m h_2^r, c^* = m * h_1 + r * h_2$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$and computes:
$R = h_1^{r_0+r_1} h_2^{r_0-r_1}$
$\mathbb{P} \rightarrow \mathbb{V}$: R
$\mathbb{V} \leftarrow \mathbb{P}$: $t \xleftarrow{\$} \mathbb{Z}_l$
$\mathbb{P}$: computes:
$y_0 = r_0 + t(m + r),\ y_1 = r_1 + t(m - r)$
$\mathbb{P} \rightarrow \mathbb{V}$: y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$h_2^{y_0-y_1} * h_1^{y_0+y_1} c^{-2t} \stackrel{?}{=} R$

Algorithm 6 Non-interactive protocol for $\mathbf{DC_{CDH}}$
$\mathcal{R}_1 = c, c^* \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = h_1^m h_2^r, c^* = m * h_1 + r * h_2$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$ and computes:
$R = h_1^{r_0+r_1} h_2^{r_0-r_1}$
$t = H_s(R, c, c^*)$ where $H_s(*)$ is a hash function
$y_0 = r_0 + t(m + r),\ y_1 = r_1 + t(m - r)$
$\mathbb{P} \rightarrow \mathbb{V}$: t, y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$H_s(h_2^{y_0-y_1} * h_1^{y_0+y_1} c^{-2t}, c, c^*) \stackrel{?}{=} t$

The interactive zero-knowledge authentication protocol presented in algorithm 7 is complete, special sound, and special honest-verifier zero knowledge as a sigma protocol for the relation R3. The corresponding non-interactive protocol is shown in algorithm 8

Algorithm 7 Interactive protocol for $\mathbf{MC_{CDH}}$
$\mathcal{R}_1 = c, c^\star \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = h_1^m h_2^r, c^\star = h_1^r h_2^m$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l$ and computes:
$R_0 = (h_1 h_2)^{r_0},\ R_1 = (h_1/h_2)^{r_1}$
$\mathbb{P} \to \mathbb{V}$: R_0, R_1
$\mathbb{V} \leftarrow \mathbb{P}$: $t \xleftarrow{\$} \mathbb{F}$
$\mathbb{P}$: computes:
$y_0 = r_0 + t(m + r),\ y_1 = r_1 + t(m - r)$
$\mathbb{P} \to \mathbb{V}$: y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$(h_1 h_2)^{y_0}(cc^\star)^{-t} \stackrel{?}{=} R_0,\ (h_1/h_2)^{y_1}(c(c^\star)^{-1})^{-t} \stackrel{?}{=} R_1$

Algorithm 8 Non-interactive protocol for $\mathbf{MC_{CDH}}$
$\mathcal{R}_1 = c, c^\star \in \mathbb{G}; m_1, m_2 \in \mathbb{Z}_l : c = h_1^m h_2^r, c^\star = h_1^r h_2^m$

$\mathbb{P}$: $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_l \quad and \quad computes:$
$R_0 = (h_1 h_2)^{r_0}, R_0 = (h_1/h_2)^{r_1}$
$t = H_s(R_0, R_1, c, c^\star)\ where\ H_s(*)\ is\ a\ hash\ function$
$y_0 = r_0 + t(m + r), y_1 = r_1 + t(m - r)$
$\mathbb{P} \to \mathbb{V}$: t, y_0, y_1
$\mathbb{V}$: returns *Accept* if and only if the following hold:
$H_s((h_1 h_2)^{y_0}(cc^\star)^{-t}, (h_1/h_2)^{y_1}(c(c^\star)^{-1})^{-t}, c, c^\star) \stackrel{?}{=} t$