Efficient Trapdoor Commitment as Secure as Factoring with Useful Properties**

Tack-Young YOUN†a), Young-Ho PARK†b), Nonmembers, and Jongin LIM†c), Member

SUMMARY Trapdoor commitment schemes are widely used for adding valuable properties to ordinary signatures or enhancing the security of weakly secure signatures. In this letter, we propose a trapdoor commitment scheme based on RSA function, and prove its security under the hardness of the integer factoring. Our scheme is very efficient in computing a commitment. Especially, it requires only three multiplications for evaluating a commitment when $e = 3$ is used as a public exponent of RSA function. Moreover, our scheme has two useful properties, key exposure freeness and strong trapdoor opening, which are useful for designing secure chameleon signature schemes and converting a weakly secure signature to a strongly secure signature, respectively.

**key words:** security, trapdoor commitment, key exposure freeness, strong trapdoor opening, integer factoring problem

1. Introduction

Trapdoor commitments permit a trapdoor key holder to open a commitment in many ways, and the property can be used for designing value-added signatures. One of the applications of trapdoor commitment schemes is the design of digital signatures with additional properties such as fast online signing and the untransferability of digital signatures. For providing the fast online signing and the untransferability, online/offline signatures [3] and chameleon signatures [5] are proposed by using trapdoor commitment schemes. The schemes are constructed by replacing ordinary hash functions in ordinary signatures with trapdoor commitment schemes. Trapdoor commitment schemes also can be used for enhancing the security of weakly secure signatures. By adopting trapdoor commitment schemes in place of ordinary hash functions, we can design strongly secure signatures that are secure against adaptive chosen message attacks from weakly secure signatures that are secure against random message attacks [9].

Useful Properties for TC Schemes. To provide valuable functionality or stronger security to ordinary signatures, we need trapdoor commitment schemes that have some useful properties. For each application, we need the key exposure freeness, fast trapdoor opening, or strong trapdoor opening of trapdoor commitments.

The trapdoor key holder is the recipient and signer in chameleon signatures and online/offline signatures, respectively, and nobody knows the trapdoor key when trapdoor commitments are used for providing stronger security. Chameleon signatures can provide untransferability of digital signatures since the recipient can open a signed commitment in more than one way to claim that the signature was generated for the message he chooses. The signer can prevent the recipient’s misbehavior by publishing a message that was used for evaluating the commitment. However, in this case, someone could discover the trapdoor key using two different inputs for the same commitment, and the leak of private trapdoor key is highly undesirable since all signatures ever generated to the associated public key of the recipient are invalidated. To resolve the weakness, the trapdoor commitment should achieve key exposure freeness which means that no one can find the trapdoor key even if he obtains a collision pair of the trapdoor commitment. In online/offline signatures, a signer issues a signature for a commitment of a random message and stores the signature and the commitment in his secure storage. When a message is given to sign, the signer opens the stored commitment for a new message, and gives the stored signature with the opened commitment as a signature of the new message. We need to compute one trapdoor opening operation in online phase, and so the efficiency of the trapdoor opening is a crucial factor for fast online signing. Hence, we need a trapdoor commitment scheme that achieves fast trapdoor opening when it is used for designing online/offline signatures. The property of the strong trapdoor opening of trapdoor commitment schemes was first considered in [9] which means that a trapdoor key holder can open a commitment without obtaining original inputs for the commitment. As indicated in [9], trapdoor commitment schemes that provide strong trapdoor opening can be used to convert any signature which is secure against random message attacks into a signature which is secure against adaptive chosen message attacks.

Contributions. In this letter, we propose a trapdoor commitment scheme, and prove its security under the hardness of the integer factoring of Blum integer. Our scheme has three advantages. When value-added signatures are designed based on trapdoor commitment scheme, ordinary hash functions are replaced by trapdoor commitment schemes, and so
the efficiency of evaluating a commitment is important for a value-added signature which is constructed by using trapdoor commitment schemes. The first strong point of our scheme is the efficiency in computing a commitment. Especially, when we use $e = 3$ as a public exponent of RSA function, our scheme requires only three multiplications for evaluating a commitment. Different from our scheme, existing trapdoor commitment schemes require costly exponentiation [1, 5, 8, 9]. Moreover, our commitment scheme is key exposure free, and so it can be used for designing secure chameleon signatures. Since our scheme is very efficient in evaluating a commitment, we can construct an efficient chameleon signature with key exposure freeness by using our commitment scheme. Finally, our scheme provides strong trapdoor opening which can be used for designing adaptive-chosen-message-attacks secure signature from a random-message-attacks secure signature. Due to the efficiency of computing a commitment, the conversion is also efficient. A weakness of our scheme is that its security is proved in the random oracle model. However, it is regarded as a reliable security model and widely used for security proofs. Moreover, the computational efficiency in computing a commitment is sufficiently attractive and valuable.

The rest of this letter is organized as follows. In Sect. 2, we review formal models for trapdoor commitment schemes. In Sect. 3, we propose a fast trapdoor commitment scheme. We analyze our scheme in terms of security and efficiency in Sect. 4. We conclude this letter in Sect. 5.

2. Formal Models

Trapdoor commitment scheme is composed with the following three algorithms:
- \( \text{Gen}(k) \): On input a security parameter \( k \), the algorithm generates a public key \( pk \) and the corresponding private trapdoor key \( tk \).
- \( \text{Com}(pk, m, r) \): For given a message \( m \) in the message space \( \mathcal{M} \) and a public key \( pk \), the algorithm chooses a random \( r \) in the space of randomness \( \mathcal{R} \), and computes a commitment \( c \) on the message \( m \) and the random value \( r \).
- \( \text{Open}(pk, tk, c, m', r', m) \): For given a message \( m \in \mathcal{M} \) and a commitment \( c \) along with a pair \( (m', r') \) such that \( c = \text{Com}(pk, m', r') \), the algorithm finds an integer \( r \in \mathcal{R} \) such that \( c = \text{Com}(pk, m, r) \). For strong trapdoor opening, the algorithm does not use \( (m', r') \) as an input.

As indicated in [5], (non-interactive) trapdoor commitment schemes and chameleon hash functions are equivalent, and so we can prove the security of a trapdoor commitment scheme under the security models for chameleon hash functions. In this letter, we consider the security models for chameleon hash functions since they are more widely used for defining the security of trapdoor commitment schemes.

Basic security requirements for trapdoor commitment schemes are collision resistance and uniformity. The strong collision resistance requires that it should be hard to find a collision without the trapdoor key, and the uniformity requires that the random value obtained by running the trapdoor opening algorithm should be indistinguishable from uniformly selected random value. An additional security requirement, the key exposure freeness, requires that it should be hard to find a collision without the trapdoor key though an adversary can access to an oracle which responds to opening queries given by the adversary.

Definition 1. A secure key exposure free trapdoor commitment scheme \( \langle \text{Gen}, \text{Com}, \text{Open} \rangle \) holds the followings:

Collision Resistance & Key Exposure Freeness: For any polynomial time algorithm \( \mathcal{A} \), the following probability is negligible in security parameter \( k \):

\[
\Pr\left[ (pk, tk) \leftarrow \text{Gen}(k), (m, r, m', r') \leftarrow \mathcal{R}^{\text{Open}}(pk) : \text{Com}(pk, m, r) = \text{Com}(pk, m', r') \land m \neq m' \right]\]

where \( \mathcal{R}^{\text{Open}} \) is an oracle that responds to the opening queries asked by \( \mathcal{A} \).

Uniformity: For a message \( m \in \mathcal{M} \), an uniformly selected random \( r \in \mathcal{R} \), and a commitment \( c \) such that \( c = \text{Com}(pk, m, r) \), the outcome of

\[
\text{Open}(pk, tk, c, m, r, m')
\]

is indistinguishable from uniform in \( \mathcal{R} \) for any \( m' \in \mathcal{M} \).

3. The Proposed Trapdoor Commitment

In this section, we propose a fast trapdoor commitment scheme. Let \( \sqrt{a} \) be a square root of \( a \in \mathbb{Z}_n^* \). Note that, we can compute \( \sqrt{a} \) as described in [7] when the condition \( p = 3 \mod 4 \) holds for all factors \( p \) of \( n \).

3.1 Description

- \( \text{Gen} \): Given security parameter \( k \), choose two primes \( p \) and \( q \) such that \( |p| = |q| = \ell_n/2 \) and \( p = q = 3 \mod 4 \). Let \( n = pq \). Note that, the size of \( \ell_n \) is determined so that the hardness of integer factoring of \( n \) guarantees \( k \)-bit security. Select a prime \( e \) such that \( \gcd(e, \phi(n)) = 1 \). The public key is \( pk = (n, e) \) and the trapdoor key is \( tk = (p, q) \). \( \mathcal{H} : \{0, 1\}^* \to \mathbb{Z}_n^* \) is a hash function that maps a string of arbitrary length to an element of \( \mathbb{Z}_n^* \).
- \( \text{Com} \): For given a message \( m \), a committer computes

\[
c = \text{Com}(pk, m, (r, s)) = \mathcal{H}(m|\{r\}^2 \cdot s^2) \mod n
\]

for randomly chosen \( r \in \{0, 1\}^{\ell_n} \) and \( s \in \mathbb{Z}_n^* \). The size of \( \ell_n \) is chosen so that the hash function \( \mathcal{H} \) guarantees the \( k \)-bit randomness for the same message \( m \). Then the commitment on \( (m, r, s) \) is \( c \).
- \( \text{Open} \): For given a commitment \( c \) and \( (m', r', s') \) such that \( c = \text{Com}(pk, m', (r', s')) \), \( \text{Com} \) computes
The proposed trapdoor commitment scheme is secure in terms of the security model formalized in Definition 1 if and only if the integer factorization of Blum integer is intractable.

Proof. Let \( n \) be a Blum integer given as a challenge. To prove the collision resistance and key exposure freeness of our scheme, we factorize \( n \) by using a polynomial time collision finding algorithm \( \mathcal{A} \). Note that, the algorithm can ask opening queries. We choose a prime \( e \), and set \( pk = [n, e] \) as a public key for the proposed scheme. We also choose a random \( a \in \mathbb{Z}^* \), compute \( b = a^2 \mod n \), and use it for simulating the hash oracle. Since we prove the collision resistance and key exposure freeness in the random oracle model, the hash function \( \mathcal{H} \) is treated as a random oracle. We respond to each query given by the algorithm \( \mathcal{A} \) as follows.

- **Hash Oracle** - We maintain a list \( L_H \) which contains previous queries, and respond to a query on a previously asked message by retrieving stored data. For given hash query on a message \( m \) and a random value \( r \), we choose a random \( \alpha \neq 0 \), compute \( \mu = b^\alpha \mod n \), store \( (m, r, \mu, \alpha) \) in \( L_H \), and give \( \mu \) to \( \mathcal{A} \) as the hash value of \( m||r \).

- **Opening Oracle** - For given opening query on a message \( m \) and a commitment \( c \) along with a pair \( (m', r', s') \) such that \( c = \text{Com}(pk, m, [r, s]) \), we choose an odd random value \( \alpha \) and compute \( \beta = (1 - ea)/2 \). Note that, \( ea + 2\beta = 1 \). Then we compute \( \mu = c^\beta \mod n \) and \( s = \sqrt{c} \mod n \), choose a random \( r \in [0, 1]^e \), set \( \mathcal{H}(m||r) = \mu \), store \( (m, r, \mu, 0) \) in \( L_H \), and give \( (m, r, s) \) to \( \mathcal{A} \).

The oracle simulation fails only when the adversary asks \( (m, c) \) to the opening oracle after he asks \( (m, r) \) to the hash oracle and the opening oracle chooses the same value \( r \) as a random number for the opening query on \( (m, c) \). In this case, the hash value of \( m||r \) is already fixed by the hash oracle, and thus we cannot control the hash value for simulating the opening oracle. However, we did not consider the above described case since the event occurs with probability \( 1/2^e \) which is negligible for sufficiently large \( e \).

The algorithm \( \mathcal{A} \) returns \( (m_1, r_1, s_1) \) and \( (m_2, r_2, s_2) \) as a collision. Note that, the collision pair holds

\[
\text{Com}(pk, m_1, r_1, s_1) = \text{Com}(pk, m_2, r_2, s_2),
\]

and so we have

\[
\mathcal{H}(m_1||r_1)^e \cdot s_1^2 = \mathcal{H}(m_2||r_2)^e \cdot s_2^2 \mod n.
\]

At first, we retrieve \( (m_1, r_1, \mu_1, \alpha_1) \) and \( (m_2, r_2, \mu_2, \alpha_2) \) from \( L_H \). Note that two values \( \alpha_1 \) and \( \alpha_2 \) are not zero because the algorithm \( \mathcal{A} \) does not return a collision which was generated by the opening oracle. Since \( \mathcal{H}(m_i||r_i) = b^{\mu_i} \mod n \) for \( i \in [1, 2] \), we also have

\[
(b^{\mu_1})^e \cdot s_1^2 = (b^{\mu_2})^e \cdot s_2^2 \mod n,
\]

and the equation can be simplified as

\[
b^\gamma = (s_2/s_1)^2 \mod n,
\]

where \( \gamma = e(\alpha_1 - \alpha_2) \). Recall that, our goal is to factorize given \( n \) by finding a square root of \( b \). Let \( \tilde{a} \) be a square root of \( b \). Note that, we have \( \tilde{a}^2 = (\sqrt{b})^2 = b \mod n \), and we can recover \( \tilde{a}^\gamma \mod n \) by computing

\[
s_2/s_1 = \sqrt{b^\gamma} = (\sqrt{b})^\gamma = \tilde{a}^\gamma \mod n.
\]

If \( \gcd(2, \gamma) = 1 \), we can find two integers \( \zeta \) and \( \eta \) such that \( 2\zeta + \gamma\eta = 1 \). Then, we can find \( \tilde{a} \) by computing

\[
(\tilde{a}^\zeta)^\gamma \cdot (\tilde{a}^\eta)^\gamma = \tilde{a}^{2\zeta + \gamma\eta} = \tilde{a} \mod n.
\]

The condition \( \gcd(2, \gamma) = 1 \) holds with probability

\[
\Pr[\gcd(2, \gamma) = 1] = 1/2
\]

since \( e \) is odd prime and \( \alpha_1, \alpha_2 \) are randomly selected integers. Then, we can obtain two square roots of \( b \) with probability \( 1/2 \), and the values can be used to factorize \( n \) by computing

\[
f = \gcd(a - \tilde{a}, n).
\]

It is well-known fact that the resulting value \( f \) is a factor of \( n \) with probability \( 1/2 \). Hence, we can factorize \( n \) using the collision returned by the algorithm \( \mathcal{A} \) with probability \( 1/4 = \frac{1}{2} \times \frac{1}{2} \). Let \( e \) be the advantage of the algorithm \( \mathcal{A} \). Then we can solve the integer factorization of Blum integer with probability \( \Adv_{\text{fact}} = e/4 \). Recall that, as we stated in Sect. 2, the above proof shows that our scheme achieves the collision resistance and the key exposure freeness.

We can prove the uniformity of our scheme as in [6] by showing the uniformity of commitment values. Given a message \( m \) and a commitment \( c \in \mathbb{Z}_n^* \), there are exactly four \( s \in \mathbb{Z}_n^* \) such that

\[
\text{Com}(pk, m, [r, s]) = c
\]
for a randomly chosen \( r \), and the random value \( s \) can be computed as

\[
    s = \sqrt{c \cdot H(m||r)} \mod n.
\]

Consequently, the output of the trapdoor opening algorithm is uniform, and so our scheme achieves the uniformity. □

### 4.2 Performance

We compare our scheme with previously proposed schemes in terms of the number of operations. Let \( M \) be the cost of a multiplication, \( SR \) be the cost of a square root computation, and \( I \) be the cost of an inverse computation. Note that the cost of an exponentiation with a-bit exponent is \( 3a/2M \). For accurate comparison, we count the number of multiplication instead of the number of exponentiations since the cost of an exponentiation is changed along with the size of exponent.

For our scheme, we can use \( e = 3 \) as the public exponent of RSA function. In general, the security of RSA function is not weakened by the size of a public exponent, and so the use of small exponent does not matter. In this case, we can evaluate a commitment with three multiplications as

\[
    c = \mu^e \cdot s^2 = \mu^3 \cdot s^2 = (\mu \cdot s)^2 \cdot \mu \mod n,
\]

where \( \mu \) is a hashed message and \( s \) is a random value. Existing trapdoor commitment schemes require costly exponentiation for evaluating a commitment [1], [5], [8], [9], and so our scheme is remarkably efficient in evaluating a commitment than other trapdoor commitment schemes. The opening algorithm requires one square root computation and one exponentiation with short exponent. Note that, the opening algorithm is rarely used when our scheme is used for designing a chameleon signature scheme, and so the cost of opening is not critical in practical point of view. Moreover, if our scheme is used for enhancing the security of weakly secure signatures, the opening operation does not used in real. Hence, the efficiency in computing a commitment is remarkable advantage of our scheme for some applications such as the design of chameleon signatures and the security enhancement of weakly secure signatures.

### 5. Conclusion

In this letter, we proposed a trapdoor commitment scheme which is very efficient in computing a commitment. We prove that the scheme is secure as the integer factoring of Blum integer. Our scheme provides some useful properties including the key exposure freeness and the strong trapdoor opening which are useful for designing value-added signatures.

### References

[1] G. Ateniese and B. Medeiros, “On the key exposure problem in chameleon hashes,” Proc. SCN’04, LNCS 3352, pp.165–179, Springer-Verlag, Berlin, 2005.

[2] X. Chen, F. Zhang, and K. Kim, “Collision free accumulators and fail-stop signature scheme without trees,” Proc. ISC’03, LNCS 3225, pp.87–98, Springer-Verlag, Berlin, 2004.

[3] S. Even, O. Goldreich, and S. Micali, “On-line/off-line digital signatures,” J. Cryptol., vol.9, no.1, pp.35–67, 1996.

[4] R. Gennaro, S. Halevi, and T. Rabin, “Secure hash-and-sign signatures without the random oracle,” Proc. EUROCRYPT’99, LNCS 1592, pp.123–139, Springer-Verlag, Berlin, 1999.

[5] H. Krawczyk and T. Rabin, “Chameleon signatures,” Proc. NDSS’00, pp.143–154, Internet Society, 2000.

[6] K. Kurosawa and K. Schmidt-Samoa, “New online/offline signature schemes without random oracles,” Proc. PKC’06, LNCS 3958, pp.330–346, Springer-Verlag, Berlin, 2006.

[7] M.O. Rabin, “Digitalized signatures and public-key functions as intractable as factorization,” Technical Report: TR-212, MIT Laboratory for Computer Science, Jan. 1979.

[8] K. Schmidt-Samoa and T. Takagi, “Paillier’s cryptosystem modulo \( p^2q \) and its applications to trapdoor commitment schemes,” Proc. Mycrypt’01, LNCS 3715, pp.296–313, Springer-Verlag, Berlin, 2005.

[9] A. Shamir and Y. Tauman, “Improved online/offline signature schemes,” Proc. CRYPTO’01, LNCS 2139, pp.355–367, Springer-Verlag, Berlin, 2001.