Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 825984, 8 pages
http://dx.doi.org/10.1155/2014/825984

Hindawi

Research Article

Provably-Secure (Chinese Government) SM2 and Simplified SM2 Key Exchange Protocols

Ang Yang,[1] Junghyun Nam,[2] Moonseong Kim,[3] and Kim-Kwang Raymond Choo[1]

[1] *Information Assurance Research Group, Advanced Computing Research Centre, University of South Australia, Mawson Lakes, SA 5095, Australia*
[2] *Department of Computer Engineering, Konkuk University, 268 Chungwondaero, Chungju, Chungcheongbuk-do 380-701, Republic of Korea*
[3] *Information Management Division, Korean Intellectual Property Office, 189 Cheongsaro, Daejeon 302-701, Republic of Korea*

Correspondence should be addressed to Junghyun Nam; jhnam@kku.ac.kr

Received 20 July 2014; Accepted 13 August 2014; Published 2 September 2014

Academic Editor: Jong-Hyouk Lee

We revisit the SM2 protocol, which is widely used in Chinese commercial applications and by Chinese government agencies. Although it is by now standard practice for protocol designers to provide security proofs in widely accepted security models in order to assure protocol implementers of their security properties, the SM2 protocol does not have a proof of security. In this paper, we prove the security of the SM2 protocol in the widely accepted indistinguishability-based Bellare-Rogaway model under the elliptic curve discrete logarithm problem (ECDLP) assumption. We also present a simplified and more efficient version of the SM2 protocol with an accompanying security proof.

1. Introduction

Due to the potential of elliptic curve cryptography (ECC) to offer similar security to established public-key cryptosystems at reduced key sizes, it has become a subject of research focus. For example, we observe an emerging trend in the use of identity-based (ID-based) cryptography, such as ID-based key agreement protocols using pairings. The latter include ID-based authenticated key agreement (ID-AKA) protocol. ID-AKA protocols (as well as other key establishment protocols such as [1–4]) allow a shared secret key to be established between two or more parties for subsequent cryptographic use. The first two-party ID-AKA protocol was proposed by Shamir, which is based on Weil Pairing [5]. Shamir's protocol requires a trusted key generation center (KGC). Challenges associated with KGC are well documented, and Alriyami and Paterson proposed the first certificateless two-party authenticated key agreement (CTAKA) protocol that does not require a KGC [6]. Since then, a number of CTAKA protocols have been proposed in the literature [7–9]. Most of these CTAKA protocols are, however, based on bilinear pairings. The latter is expensive, especially in comparison to RSA algorithm [10, 11].

A number of recently published certificateless ECC-based AKA protocols that do not require the use of pairings have been proposed. For example, in 2007, Zhu et al. proposed a pairing-free ID-AKA protocol [12]. However, the combination of a pairing-free ID-based signature scheme with the Diffie-Hellman key exchange in the proposed protocol results in larger computation complexity and message size. In addition, the protocol and the ECC-based pairing-free ID-AKA protocol of Cao et al. [13] require three rounds of message exchanges. Another later protocol of Cao et al. reduces the minimum message exchange rounds to two and the protocol was proven secure in the Bellare-Rogaway model [10]. He et al. also independently proposed a two-round certificateless ID-AKA protocol without the use of pairings [14] and a three-round certificateless ID-AKA protocol without the use of pairings [2], respectively.

In 2011, the Chinese government published an ECC-based key exchange protocol, SM2 [15]. According to the official report from the Chinese Government State Cryptography

Administration and various media releases, SM2 protocol is mandatory in various cryptographic applications used by Chinese government agencies from 1st July, 2011 [16–18]. A 2005 survey by Boyd and Choo revealed that the purported security of several published ID-based protocols is based on heurstic security arguments. A number of protocols were also found to be proven secure in a restricted model. This study highlighted the need for more rigorously tested identity-based protocols [19]. Surprisingly, we observed that despite the wide usage of the SM2 protocol among Chinese commercial applications/electronics, it does not have a security proof.

A protocol's goal is defined as the properties that the protocol aims to achieve. As Boyd and Mathuria suggested, any attack on a protocol is only valid if it violates some property that the protocol was intended to achieve [20]. Without identifying at an early stage the properties and/or goals that a protocol offers, one can debate the validity of attacks against a published protocol since it may not be clear whether the protocol is not intended to provide assurances against the properties being exploited [21]. This reinforced the importance of having a security proof for protocols, particularly those that are widely used by government agencies and in the private sector.

Our contributions in this paper are two-fold.

(1) We prove the SM2 protocol secure in the widely accepted indistinguishability-based model of Bellare and Rogaway under the ECDLP assumption.

(2) We propose a simplified version of SM2 protocol that is more efficient, and prove it secure in the Bellare-Rogaway model under the ECDLP assumption.

In the next section, we will briefly review the model that we work in. We revisit the SM2 protocol and prove it secure in Section 3. Section 4 describes our simplified SM2 protocol and its proof of security. Finally, the last section concludes the paper.

2. Overview of the Bellare-Rogaway Model

In the Bellare-Rogaway model [22, 23], the adversary (denoted by $\mathcal{A}$) controls the communication channel by interacting with a set of $\Pi^i_{U_1,U_2}$ oracles. $\Pi^i_{U_1,U_2}$ is defined to be the ith instantiation of a protocol participant, U_1 in a specific protocol run and U_2 is the other protocol participant, with whom U_1 wishes to establish a secret key. The predefined oracle queries are described informally as follows.

(i) The Send (U_1, U_2, i, m) query allows $\mathcal{A}$ to send a message m to another protocol participant at will. In other words,

(a) $\Pi^i_{U_1,U_2}$, upon receiving the query, will compute what the protocol specification demands. The response message and/or decision will then be sent to $\mathcal{A}$,

(b) if $\Pi^i_{U_1,U_2}$ has either accepted with some session key or terminated, this will be made known to $\mathcal{A}$.

(ii) The Reveal (U, i) query allows $\mathcal{A}$ to expose a previously accepted session key. In other words, U^i, upon receiving this query and if it has accepted and holds some session key, will send this session key back to $\mathcal{A}$.

(iii) The Corrupt (U) query allows $\mathcal{A}$ to learn the complete internal state of U. This models the real world scenario of a corrupted insider.

(iv) The Test (U_1, U_2, i) query is the only oracle query that does not correspond to any of $\mathcal{A}$'s abilities. If $\Pi^i_{U_1,U_2}$ has accepted with some session key and is being asked a Test (U_1, U_2, i) query; then depending on a randomly chosen bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.

Definition 1 (Definition of Partnership). Let us denote $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ as two oracles in the protocol run. These two oracles are considered partners if and only if

(i) both $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ have accepted the same session key,

(ii) only $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ (i.e., no other oracle) have accepted with the same session ID (i.e., SID, which is defined to be the concatenation of the message flows) and agreed on the same set of principals (i.e., the initiator and the responder of the protocol).

Definition 2 (Definition of Freshness). Oracle $\Pi^i_{A,B}$ holds a fresh session key at the end of execution, if and only if all the following conditions are satisfied:

(i) $\Pi^i_{A,B}$ has accepted with, or without, a partner oracle $\Pi^j_{B,A}$,

(ii) $\Pi^i_{A,B}$ and $\Pi^j_{B,A}$ (if such a partner oracle exists) has/have not been sent a Reveal query,

(iii) both A and B (if such a partner exists) has/have not been sent a Corrupt query.

The definition of security depends on the notions of partnership as outlined in Definition 1 and freshness as outlined in Definition 2 and is defined using the game $\mathcal{G}$ and played between $\mathcal{A}$ and a collection of $\Pi^i_{U_x,U_y}$ oracles for players $U_x, U_y \in \{U_1, \ldots, U_{N_p}\}$ and instances $i \in \{1, \ldots, N_s\}$. $\mathcal{A}$ runs the game simulation $\mathcal{G}$, whose setting is as follows.

(i) Send, Reveal, and Corrupt oracle queries are sent by $\mathcal{A}$ in any order at will.

(ii) $\mathcal{A}$ chooses a fresh session on which to be tested by sending a Test query to the fresh oracle associated with the test session at some point during $\mathcal{G}$. This chosen test session must be fresh (in the sense of Definition 2). Depending on a randomly chosen bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.

(iii) $\mathcal{A}$ continues making any Send, Reveal, and Corrupt oracle queries of its choice.

(iv) $\mathscr{A}$ will eventually terminate $\mathscr{G}$ and outputs its guess of the value of b, denoted as b'.

We measure $\mathscr{A}$'s success in $\mathscr{G}$ in terms of $\mathscr{A}$'s advantage in distinguishing whether $\mathscr{A}$ receives the real key or a random value (i.e., whether $b' = b$).

Let k be a security parameter. Then, the advantage function of $\mathscr{A}$ is denoted by $\mathsf{Adv}^{\mathscr{A}}(k)$, where

$$\mathsf{Adv}^{\mathscr{A}}(k) = 2 \times \mathsf{Pr}\left[b' = b\right] - 1. \quad (1)$$

Definition 3 (Definition of Security). A protocol is secure in the Bellare-Rogaway model if both the following requirements are satisfied.

(1) Two oracles accept the same key when the protocol is run in the absence of a malicious adversary.

(2) For all probabilistic, polynomial-time (PPT) adversaries $\mathscr{A}$, $\mathsf{Adv}^{\mathscr{A}}(k)$ is negligible.

3. A Provably-Secure SM2 Key Exchange Protocol

3.1. SM2 Key Exchange Protocol. The notations used in SM2 protocol (Table 1) are as follows:

(i) A, B: two SM2 protocol participants with identities ID_A and ID_B respectively,

(ii) a, b: $a, b \in f_q$: the parameters of the elliptic curve E on F_q where elliptic function is $y^2 = x^3 + ax + b$,

(iii) F_q: the prime field F includes q elements,

(iv) $E(F_q)$: the set of all points on the elliptic curve E defined over F_q,

(v) G: the base point of elliptic curve, order of G is prime number that $G = (x_G, y_G)$,

(vi) h: cofactor, $h = \#E(F_q)/n$, n is the order of G,

(vii) $\mathscr{D}$: the space of number that $\mathscr{D} = [1, n-1]$,

(viii) (d, P): long-term private and public key pair,

(ix) K: session key,

(x) ID: identification of client,

(xi) Z: hash value of identification, the length of Z is $entlen_A$ bits,

(xii) $H_v()$: one-way hash function,

(xiii) (r, R): temporary private and public key pair,

(xiv) *klen* and *entlen*: the bit length of the key and ID, respectively,

(xv) KDF$(Z, klen)$: the one-way key derivation hash function whose output length is *klen*,

(xvi) $x \| y$: concatenation of two strings x and y,

(xvii) $\perp$: there is no message or the value is not known.

A randomly selects $d_A \in [1, n-1]$ and computes $P_A = [d_A]G = (x_A, y_A)$, prior to sending P_A to B. B will also randomly select $d_B \in [1, n-1]$ and compute $P_B = [d_B]G = (x_B, y_B)$, before sending P_B to A. The public parameters are $(E(Fq), G, n, Z_A, Z_B, P_A, P_B)$. And Z_A and Z_B are hash values of the identification of A and B, respectively, where $Z_A = H(ENTL_A \| \mathrm{ID}_A \| a \| b \| x_G \| y_G \| x_A \| y_A)$ and $Z_B = H(ENTL_B \| \mathrm{ID}_B \| a \| b \| x_G \| y_G \| x_B \| y_B)$. To establish a session key with client B,

(1) client A will now run the protocol as follows:

(a) randomly selects $r_A \in [1, n-1]$,

(b) computes $R_A = [r_A]G = (x_1, y_1)$ and $\overline{x_1} = 2^w + (x_1 \& (2^w - 1))$ where $w = \lceil (\lceil \log_2(n) \rceil / 2) \rceil - 1$,

(c) computes $t_A = (d_A + \overline{x_1} \cdot r_A) \mod n$,

(d) sends R_A to client B;

(2) upon receiving the message R_A from client A, B will perform the following:

(a) randomly selects $r_B \in [1, n-1]$,

(b) computes $R_B = [r_B]G = (x_2, y_2)$, $\overline{x_2} = 2^w + (x_2 \& (2^w - 1))$, $t_B = (d_B + \overline{x_2} \cdot r_B) \mod n$, $V = [h \cdot t_B](P_A + [\overline{x_1}]R_A) = (x_V, y_V)$ and $K_B = \mathrm{KDF}(x_V \| y_V \| Z_A \| Z_B, klen)$,

(c) (optional for key confirmation) computes $S_B = H(0x02 \| y_V \| H(x_V \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$,

(d) Sends R_B to client A (optional for key confirmation, B will also send S_B to A);

(3) upon receiving the messages, R_B (and S_B), from B, client A will perform the following:

(a) computes $\overline{x_2} = 2^w + (x_2 \& (2^w - 1))$, $U = [h \cdot t_A](P_B + [\overline{x_2}]R_B) = (x_U, y_U)$ and $K_A = \mathrm{KDF}(x_U \| y_U \| Z_A \| Z_B, klen)$,

(b) (optional for key confirmation) computes $S_1 = H(0x02 \| y_U \| H(x_U \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$,

(c) verifies $S_1 \stackrel{?}{=} S_B$, and if it returns true then A is assured that B actually has possession of the session key, otherwise, terminates the protocol run and outputs $\perp$,

(i) (optional for key confirmation, computes $S_A = H(0x03 \| y_U \| H(x_U \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2)))$,

(ii) Sends S_A to client B;

(4) (optional for key confirmation) upon receiving the message (S_A) from client A, client B will perform the following:

(a) computes $S_2 = H(0x03 \| y_V \| H(x_V \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$,

(b) verifies whether $S_A \stackrel{?}{=} S_2$,

(c) if the verification returns wrong, then client B terminates the protocol run and outputs $\perp$,

Table 1: SM2 key exchange protocol.

A		B
$d_A, E(F_q), G, n, Z_A, Z_B, P_A, P_B, H(), \mathrm{KDF}(), G$		$d_B, E(F_q), G, n, Z_A, Z_B, P_A, P_B, H(), \mathrm{KDF}(), G$
Randomly selects $r_A \in [1, n-1]$		Randomly selects $r_B \in [1, n-1]$
Computes $R_A = [r_A]G = (x_1, y_1)$		Computes $R_B = [r_B]G = (x_2, y_2)$
Computes $\overline{x_1} = 2^w + (x_1 \& (2^w - 1))$		Computes $\overline{x_2} = 2^w + (x_2 \& (2^w - 1))$
Computes $t_A = (d_A + \overline{x_1} \cdot r_A) \bmod n$		Computes $t_B = (d_B + \overline{x_2} \cdot r_B) \bmod n$
	$\xrightarrow{R_A}$	
		Computes $\overline{x_1} = 2^w + (x_1 \& (2^w - 1))$
		Computes $V = [h \cdot t_B](P_A + [\overline{x_1}]R_A) = (x_V, y_V)$
		Computes $K_B = \mathrm{KDF}(x_V \| y_V \| Z_A \| Z_B, \mathrm{klen})$
		Computes $S_B = H(0x02 \| y_V \| H(x_V \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$
	$\xleftarrow{R_B, S_B}$	
Computes $\overline{x_2} = 2^w + (x_2 \& (2^w - 1))$		
Computes $U = [h \cdot t_A](P_B + [\overline{x_2}]R_B) = (x_U, y_U)$		
Computes $K_A = \mathrm{KDF}(x_U \| y_U \| Z_A \| Z_B, \mathrm{klen})$		
Computes $S_1 = H(0x02 \| y_U \| H(x_U \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$		
Verifies $S_1 \stackrel{?}{=} S_B$		
Computes $S_A = H(0x03 \| y_U \| H(x_U \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$		
	$\xrightarrow{S_A}$	
		Computes $S_2 = H(0x03 \| y_V \| H(x_V \| Z_A \| Z_B \| x_1 \| y_1 \| x_2 \| y_2))$
		Verifies $S_2 \stackrel{?}{=} S_A$
Key established $K_A = K_B$		Key established $K_B = K_A$
SID is $(R_A \| R_B)$ or $(R_A \| R_B \| S_B \| S_A)$ in the case where key confirmation is required.		

(d) otherwise, client B is assured that A actually has possession of the session key;

(5) session key established is $K_A = K_B$,

(6) SID is $(R_A \| R_B)$ or $(R_A \| R_B \| S_B \| S_A)$ in the case where key confirmation is required.

3.2. Security Proof. The security of the protocol—see Theorem 5—is based on the ECDLP assumption (see Definition 4) in the random oracle model.

Definition 4 (ECDLP Assumption). The ECDLP problem is defined as follows:

$$\begin{aligned} &\text{Instance: } E \setminus F_p, \quad P, Q \in E\left(F_p\right) \\ &\text{Output: } k \in N^*, \quad k > 1 \text{ such that } Q \equiv kP \bmod p. \end{aligned} \tag{2}$$

If we can solve the discrete logarithm problem (DLP) [26], then we can also (immediately) solve the ECDLP problem.

Theorem 5. *SM2 protocol is secure in the sense of Definition 3 when the underlying hash and key derivation schemes are modelled as random oracles and the elliptic curve discrete logarithm problem (ECDLP) assumption is satisfied in $E(F_q)$.*

The soundness requirement is trivial to verify. We will now concentrate on proving the indistinguishability requirement.

In the usual tradition of reductionist proofs, we assume that there exists an adversary $\mathscr{A}$ against the protocol (i.e., $\mathscr{A}$ has a nonnegligible advantage, $\eta(k)$, where k is the security parameter), and we then construct a solver $\mathscr{S}$ that makes use of $\mathscr{A}$ to solve the ECDLP problem. In other words, $\mathscr{S}$ will simulate the view of $\mathscr{A}$ by answering all Send, Reveal, Corrupt and Test queries of $\mathscr{A}$. $\mathscr{S}$ will start by randomly selecting two users, I and J, and a session number, i, as the test session. $\mathscr{S}$ will also manage two random oracles, H and KDF, in order to answer $\mathscr{A}$'s queries. More specifically when the H oracle is queried, $\mathscr{S}$ will check whether the tuple is already in the H-list and output the stored response. Otherwise, $\mathscr{S}$ will respond with the appropriate output, $H(\cdots)$, and adds the tuple $(H(\cdots), U, i)$ to the H-list. $\mathscr{S}$ will answer KDF queries in the same manner.

(i) Send $(\Omega_I, \Omega_J, j, m)$ queries: for any well-formed Send queries from $\mathscr{A}$, $\mathscr{S}$ can trivially answer with the right output as the protocol specification demands. Specifically, $\mathscr{S}$ answers the query as follows.

(a) If Ω_I = initiator and Ω_J = responder, then the $\mathscr{S}$ will output the message $m = (R_I, Z_I, Z_J, P_I, P_J)$.

(b) Consider the case that Ω_J = initiator, Ω_I = responder, and message $m = R_I$.

(1) If $\mathscr{S}$ has rejected the message m, then $\mathscr{S}$ will respond with $\perp$. Otherwise, $\mathscr{S}$ will verify whether m is the right format or not.

(2) If m verifies correctly, then $\mathcal{S}$ will output messages (R_J, S_J) to $\mathcal{A}$. Otherwise, $\mathcal{S}$ will abort the simulation and output $\perp$.

(c) Assume Ω_I = initiator, Ω_J = responder, and messages $m = R_J, S_J$.

(1) If $\mathcal{S}$ has rejected the messages m, then $\mathcal{S}$ will respond with $\perp$. Otherwise, $\mathcal{S}$ will verify whether m is the right format or not.

(2) If m verifies correctly, then $\mathcal{S}$ will output messages S_I to $\mathcal{A}$. Otherwise, $\mathcal{S}$ will abort the simulation and output $\perp$.

(ii) Reveal (Ω, j) queries: if $\Omega_j = I_i$ or $\Omega_j = J_i$, then $\mathcal{S}$ will abort the simulation and fail. Otherwise this query can be answered with the right session key as long as U_j has accepted and neither Ω nor its partner has been corrupted. However, such a session will be rendered unfresh.

(iii) Corrupt (Ω) queries: this query can be easily answered as per the protocol specifications, unless $\Omega = I$ or $\Omega = J$. In the latter scenario, $\mathcal{S}$ will abort the simulation and fail.

(iv) Test $(\Omega_1, \Omega_2, j, m)$ queries: if $\Pi^j_{U_1,U_2} \neq \Pi^i_{I,J}$, then $\mathcal{S}$ will abort the simulation and fail. Otherwise, $\mathcal{S}$ will check whether $\Pi^j_{\Omega_1,\Omega_2}$ has accepted and that the session is fresh. If so, $\mathcal{A}$ will be given either the actual session key or a session key drawn randomly from the session key distribution, depending on the randomly chosen bit b.

For $\mathcal{A}$ to distinguish whether the value returned is the actual session key or a session key drawn randomly from the session key distribution, $\mathcal{A}$ has to determine the correct values of $U = (x_U, y_U)$ or $V = (x_V, y_V)$ to compute the session key (since $K_I = \mathrm{KDF}(x_U \| y_U \| Z_A \| Z_B, klen)$ and $K_J = \mathrm{KDF}(x_V \| y_V \| Z_A \| Z_B, klen)$). For this to happen,

(i) $\mathcal{A}$ has to guess the long-term private key d_I and short-term private key r_I in order to compute t_I and hence, the session key K_I. If $\mathcal{A}$ is able to successfully guess d_I and r_I via $P_I = [d_I]G$ and $R_I = [r_I]G$, then $\mathcal{S}$ would be able to use $\mathcal{A}$ to solve the ECDLP problem. Let $\mathrm{Succ}^{\mathsf{ECDLP}}_{d_I}$ and $\mathrm{Succ}^{\mathsf{ECDLP}}_{r_I}$ denote $\mathcal{A}$'s advantage in computing the correct values of d_I and r_I, respectively, and $\mathrm{Succ}^{U}_{\mathcal{A}}$ denotes the event that $\mathcal{A}$ can successfully guess the session key using the computed values of d_I and r_I. So we have

$$\Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{d_I}\right] = \frac{1}{|\mathscr{D}|}; \qquad \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{r_I}\right] = \frac{1}{|\mathscr{D}|},$$
$$\Pr\left[\mathrm{Succ}^{U}_{\mathcal{A}}\right] = \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{d_I}\right] \cdot \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{r_I}\right] = \frac{1}{|\mathscr{D}|^2}. \tag{3}$$

(ii) $\mathcal{A}$ has to guess the correct value of V. Similar to the above, we let $\mathrm{Succ}^{\mathsf{ECDLP}}_{d_J}$ and $\mathrm{Succ}^{\mathsf{ECDLP}}_{r_J}$ denote $\mathcal{A}$'s advantage in computing the correct value of d_J and r_J, respectively, and $\mathrm{Succ}^{V}_{\mathcal{A}}$ denotes the event that $\mathcal{A}$ can successfully guess the session key using the computed values of d_J and r_J. So we have

$$\Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{d_J}\right] = \frac{1}{|\mathscr{D}|}; \qquad \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{r_J}\right] = \frac{1}{|\mathscr{D}|},$$
$$\Pr\left[\mathrm{Succ}^{V}_{\mathcal{A}}\right] = \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{d_J}\right] \cdot \Pr\left[\mathrm{Succ}^{\mathsf{ECDLP}}_{r_J}\right] = \frac{1}{|\mathscr{D}|^2}. \tag{4}$$

There is, therefore, a negligible advantage of $\mathcal{A}$ distinguishing whether the value returned is the actual session key or a session key drawn randomly from the session key distribution. Let q_t and $\mathrm{Succ}^{\mathrm{SM2}}_{\mathcal{A}}$ denote the number of Test queries asked and the event that $\mathcal{A}$ can correctly distinguish the session key, respectively. We now have

$$\begin{aligned}\Pr\left[\mathrm{Succ}^{\mathrm{SM2}}_{\mathcal{A}}\right] &= \left(1 - \Pr\left[\overline{\mathrm{Succ}^{U}_{\mathcal{A}}}\right] \Pr\left[\overline{\mathrm{Succ}^{V}_{\mathcal{A}}}\right]\right) \cdot q_t \\ &= \left(1 - \left(1 - \frac{1}{|\mathscr{D}|^2}\right) \cdot \left(1 - \frac{1}{|\mathscr{D}|^2}\right)\right) \cdot q_t \\ &= \frac{2q_t}{|\mathscr{D}|^2} - \frac{q_t}{|\mathscr{D}|^4}.\end{aligned} \tag{5}$$

Since $\mathscr{D} = [1, n-1]$, $n \to \infty$, $\mathscr{D} \to \infty$, $1/|\mathscr{D}|^2 \to 0$, $1/|\mathscr{D}|^4 \to 0$, $q_t \ll \mathscr{D}$, and $q_t/|\mathscr{D}|^2 \to 0$, $q_t/|\mathscr{D}|^4 \to 0$, we have $((2q_t/|\mathscr{D}|^2) - (q_t/|\mathscr{D}|^4)) \to 0$ and $((2q_t/|\mathscr{D}|^2) - (q_t/|\mathscr{D}|^4)) = (q_t/|\mathscr{D}|^2)(2 - (1/|\mathscr{D}|^2)) > 0$. Therefore,

$$\Pr\left[\mathrm{Succ}^{\mathrm{SM2}}_{\mathcal{A}}\right] = \frac{2q_t}{|\mathscr{D}|^2} - \frac{q_t}{|\mathscr{D}|^4} \longrightarrow 0. \tag{6}$$

This concludes the proof for Theorem 5.

4. A Provably-Secure Simplified SM2 Key Exchange Protocol

In this section, we propose a more efficient version of the SM2 protocol—see Table 2—and prove its security in the Bellare-Rogaway model.

4.1. Protocol Description. A randomly selects $d_A \in [1, n-1]$ and computes long-term public key $P_A = [d_A]G = (x_A, y_A)$ and $Z_A = H(ENTL_A \| \mathrm{ID}_A \| a \| b \| x_G \| y_G \| x_A \| y_A)$. It then sends P_A and Z_A to B. B also randomly selects $d_B \in [1, n-1]$ and computes $P_B = [d_B]G = (x_B, y_B)$ and $Z_B = H(ENTL_A \| \mathrm{ID}_A \| a \| b \| x_G \| y_G \| x_B \| y_B)$, prior to sending P_B and Z_B to A. $E(Fq)$, G, n, $H()$, KDF(), G, P_A, P_B, Z_A, Z_B are system parameters. To establish a session key with B,

(1) A will now run the protocol as follows:

(a) randomly selects $r_A \in [1, n-1]$,

(b) computes $t_A = (d_A \cdot r_A) \bmod n$ and $R_A = [t_A]G = (x_1, y_1)$,

(c) sends R_A to B;

Table 2: Simplified SM2 key exchange protocol.

A		B
pub: $E(F_q), G, n, H(), \mathrm{KDF}(), G, P_A, P_B, Z_A, Z_B$		
Randomly selects $r_A \in [1, n-1]$		Randomly selects $r_B \in [1, n-1]$
Computes $t_A = (d_A \cdot r_A) \bmod n$		Computes $t_B = (d_B \cdot r_B) \bmod n$
Computes $R_A = [t_A]G = (x_1, y_1)$		Computes $R_B = [t_B]G = (x_2, y_2)$
	$\xrightarrow{R_A}$	
		Computes $V = [t_B] \cdot R_A + d_B \cdot P_A = (x_V, y_V)$
		Computes $K_B = \mathrm{KDF}(x_V \Vert y_V \Vert Z_A \Vert Z_B, \mathrm{klen})$
		Computes $S_B = H(0x02 \Vert y_V \Vert H(x_V \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$
	$\xleftarrow{R_B, S_B}$	
Computes $U = [t_A] \cdot R_B + d_A \cdot P_B = (x_U, y_U)$		
Computes $K_A = \mathrm{KDF}(x_U \Vert y_U \Vert Z_A \Vert Z_B, \mathrm{klen})$		
Computes $S_1 = H(0x02 \Vert y_U \Vert H(x_U \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$		
Verifies $S_1 \stackrel{?}{=} S_B$		
Computes $S_A = H(0x03 \Vert y_U \Vert H(x_U \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$		
	$\xrightarrow{S_A}$	
		Computes $S_2 = H(0x03 \Vert y_V \Vert H(x_V \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$
		Verifies $S_2 \stackrel{?}{=} S_A$
Key established $K_A = K_B$		Key established $K_B = K_A$
SID is $(R_A \Vert R_B)$ or $(R_A \Vert R_B \Vert S_B \Vert S_A)$ in the case where key confirmation is required.		

(2) upon receiving P_A from A, B will perform the following:

(a) randomly selects $r_B \in [1, n-1]$,

(b) computes $t_B = (d_B \cdot r_B) \bmod n$, $R_B = [t_B]G = (x_2, y_2)$, $V = [t_B] \cdot R_A + d_B \cdot P_A = (x_V, y_V)$, and $K_B = \mathrm{KDF}(x_V \Vert y_V \Vert Z_A \Vert Z_B, \mathrm{klen})$,

(c) (optional for key confirmation) computes $S_B = H(0x02 \Vert y_V \Vert H(x_V \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$,

(d) sends R_B to A (optionally for key confirmation, B will also send S_B to A);

(3) upon receiving R_B (and S_B, optionally for key confirmation) from B, A will perform the following:

(a) computes $U = [t_A] \cdot R_B + d_A \cdot P_B = (x_U, y_U)$ and $K_A = \mathrm{KDF}(x_U \Vert y_U \Vert Z_A \Vert Z_B, klen)$,

(b) (optional for key confirmation) compute $S_1 = H(0x02 \Vert y_U \Vert H(x_U \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$,

(c) verifies that $S_1 \stackrel{?}{=} S_B$, and if it returns true, then A is assured that B actually has possession of the session key, otherwise, terminates the protocol run and outputs $\perp$,

(i) (optional for key confirmation) computes $S_A = H(0x03 \Vert y_U \Vert H(x_U \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$,

(ii) Send S_A to client B;

(4) (optional for key confirmation) upon receiving S_A, B will perform the following:

(a) computes $S_2 = H(0x03 \Vert y_V \Vert H(x_V \Vert Z_A \Vert Z_B \Vert x_1 \Vert y_1 \Vert x_2 \Vert y_2))$,

(b) verifies whether $S_A \stackrel{?}{=} S_2$,

(c) if the verification returns wrong, then client B terminates the protocol run and outputs $\perp$. If it returns true, then B is assured that A actually has possession of the session key. Otherwise, terminates the protocol run and outputs $\perp$;

(5) session key established is $K_A = K_B$,

(6) SID is $(R_A \Vert R_B)$ or $(R_A \Vert R_B \Vert S_B \Vert S_A)$ in the case where key confirmation is required.

4.2. Security Proof

Theorem 6. *The simplified SM2 protocol (Table 2) is secure in the sense of Definition 3 when the underlying hash and key derivation schemes are modelled as random oracles and the ECDLP assumption is satisfied in $E(F_q)$.*

The proof process is similar to that of Section 3.2.

(i) Send $(\Omega_I, \Omega_J, j, m)$ queries: for any well-formed Send queries from $\mathcal{A}$, $\mathcal{S}$ can trivially answer with the right output as the protocol specification demands. Specifically, $\mathcal{S}$ answers the query as follows.

(a) If Ω_I = initiator and Ω_J = responder, then the $\mathcal{S}$ will output the message, $m = R_I$, to the query.

(b) Consider the case that Ω_J = initiator, Ω_I = responder, and messages $m = R_I$.

(1) If $\mathcal{S}$ has rejected the message m, then $\mathcal{S}$ will respond with $\perp$. Otherwise, $\mathcal{S}$ will verify whether m is the right format or not.

(2) If m verifies correctly, then $\mathcal{S}$ will output messages (R_J, S_J) to $\mathcal{A}$. Otherwise, $\mathcal{S}$ will abort the simulation and output $\perp$.

(c) Assume Ω_I = initiator, Ω_J = responder, and messages $m = R_J, S_J$.

(1) If $\mathcal{S}$ has rejected the messages m, then $\mathcal{S}$ will respond with $\perp$. Otherwise, $\mathcal{S}$ will verify whether m is the right format or not.
(2) If m verifies correctly, then $\mathcal{S}$ will output messages S_I to $\mathcal{A}$. Otherwise, $\mathcal{S}$ will abort the simulation and output $\perp$.

Simulations for the Reveal, Corrupt and Test follow that of Section 3.2.

For $\mathcal{A}$ to distinguish whether the value returned is the actual session key or a session key drawn randomly from the session key distribution (i.e., whether $b = 0$ or $b = 1$), $\mathcal{A}$ has to determine the correct values of $U = (x_U, y_U)$ or $V = (x_V, y_V)$ (since $K_I = \mathrm{KDF}(x_U \| y_U \| Z_A \| Z_B, klen)$ and $K_J = \mathrm{KDF}(x_V \| y_V \| Z_A \| Z_B, klen)$). For this to happen, $\mathcal{A}$ has to obtain the correct value of $d_I \cdot r_I$ in order to compute t_I and consequently, the session key K_I. For $\mathcal{A}$ to obtain t_I, $\mathcal{A}$ has to be able to compute from R_I since $R_I = [t_I]G$.

Let $\mathrm{Succ}_{t_I}^{\mathrm{ECDLP}}$ denote $\mathcal{A}$'s advantage in computing t_I from R_I, and we have

$$\Pr\left[\mathrm{Succ}_{t_I}^{\mathrm{ECDLP}}\right] = \frac{1}{|\mathscr{D}|}. \tag{7}$$

Let $\mathrm{Succ}_{t_J}^{\mathrm{ECDLP}}$ denote $\mathcal{A}$'s advantage in computing t_J from R_J, and we have

$$\Pr\left[\mathrm{Succ}_{t_J}^{\mathrm{ECDLP}}\right] = \frac{1}{|\mathscr{D}|}. \tag{8}$$

Let $\mathrm{Succ}_{\mathcal{A}}^{\mathrm{ESM2}}$ denote the event that $\mathcal{A}$ is able to distinguish whether the value returned is the actual session key or a session key drawn randomly from the session key distribution. We then have

$$\begin{aligned}\Pr\left[\mathrm{Succ}_{\mathcal{A}}^{\mathrm{ESM2}}\right] &= \left(1 - \Pr\left[\overline{\mathrm{Succ}_{t_I}^{\mathrm{ECDLP}}}\right] \cdot \Pr\left[\overline{\mathrm{Succ}_{t_J}^{\mathrm{ECDLP}}}\right]\right) \cdot q_t \\ &= \left(1 - \left(1 - \frac{1}{|\mathscr{D}|}\right)^2\right) \cdot q_t \\ &= \frac{2q_t}{|\mathscr{D}|} - \frac{q_t}{|\mathscr{D}|^2}.\end{aligned} \tag{9}$$

Since $\mathscr{D} = [1, n-1]$, $n \to \infty$, $\mathscr{D} \to \infty$, $2/|\mathscr{D}| \to 0$, $1/|\mathscr{D}|^4 \to 0$, $q_t \ll \mathscr{D}$, and $2q_t/|\mathscr{D}| \to 0$, $q_t/|\mathscr{D}|^2 \to 0$, we have $((2q_t/|\mathscr{D}|) - (q_t/|\mathscr{D}|^2)) \to 0$ and $((2q_t/|\mathscr{D}|) - (q_t/|\mathscr{D}|^2)) = (q_t/|\mathscr{D}|)(2 - (1/|\mathscr{D}|)) > 0$. It follows that $\Pr[\mathrm{Succ}_{\mathcal{A}}^{\mathrm{ESM2}}] = (2q_t/|\mathscr{D}|) - (q_t/|\mathscr{D}|^2) \to 0$.

This concludes the proof for Theorem 6.

Table 3: Protocol comparison.

Protocol	Cost	Both implicit and explicit key confirmation?
Cao et al. (2008) [13]	$10m + 4a + 4h + 1e$	No
Cao et al. (2010) [10]	$8m + 3a + 2h$	No
He et al. (2012) [2]	6m + 5a + 2e + 2h	No
Yang and Tan (2011) [24]	$6m + 2h$	No
He et al. (2011) [14]	$5m + 4a + 2h$	No
Chen and Han (2013) [25]	$9m + 2a + 7h + 1e$	No
SM2 [15]	$5m + 4a + 3h$	Yes
Our simplified SM2 protocol	$4m + 1a + 3h$	Yes

5. Conclusion

Key exchange protocols are the cornerstone of any secure communication. By proving the widely used Chinese Government SM2 protocol secure in the Bellare-Rogaway model under the ECDLP assumption, we hope that this provides a strong assurance to protocol implementers that the protocol is behaving as desired. In addition, we presented a more efficient version of the SM2 protocol with a proof of security in the Bellare-Rogaway model under the ECDLP assumption.

A comparison with six existing pairing-free protocols reveals that the computational load of our simplified SM2 protocol is no more than that of the six and the SM2 protocols, yet provides both implicit key confirmation (A is assured that B can compute the session key) and explicit key confirmation (A is assured that B has actually computed the session key)—see Table 3. In Table 3, a, m, e, and h denote addition, multiplication, exponentiation, and hash operations, respectively.

Conflict of Interests

As the authors of the paper, we do not have a direct financial relation with any institution or organization mentioned in our paper that might lead to a conflict of interests for any of the authors.

References

[1] Z. Cheng, Y. Liu, C. Chang, and C. Guo, "A fault-tolerant group key agreement protocol exploiting dynamic setting," *International Journal of Communication Systems*, vol. 26, no. 2, pp. 259–275, 2013.

[2] D. He, J. Chen, and J. Hu, "A pairing-free certificateless authenticated key agreement protocol," *International Journal of Communication Systems*, vol. 25, no. 2, pp. 221–230, 2012.

[3] D. He, C. Chen, M. Ma, S. Chan, and J. Bu, "A secure and efficient password-authenticated group key exchange protocol for mobile ad hoc networks," *International Journal of Communication Systems*, vol. 26, no. 4, pp. 495–504, 2013.

[4] K. R. Choo, J. Nam, and D. Won, "A mechanical approach to derive identity-based protocols from Diffie-Hellman-based protocols," *Information Sciences*, vol. 281, pp. 182–200, 2014.

[5] A. Shamir, "Identity-based cryptosystems and signature schemes," in *in Advances in Cryptology—CRYPTO 1984*, vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, Springer, Berlin, Germany, 1985.

[6] S. Al-Riyami and K. Paterson, "Certificateless public key cryptography," in *Advances in Cryptology—ASIACRYPT 2003*, vol. 2894 of *Lecture Notes in Computer Science*, pp. 452–473, Springer, Berlin, Germany, 2003.

[7] G. Lippold, C. Boyd, and J. Nieto, "Strongly secure certificateless key agreement," in *Pairing-Based Cryptography—Pairing 2009*, vol. 5671 of *Lecture Notes in Computer Science*, pp. 206–230, Springer, Berlin, Germany, 2009.

[8] C. Swanson, *Security in key agreement: two-party certificateless schemes [M.S. thesis]*, University of Waterloo, 2008.

[9] L. Zhang, F. Zhang, Q. Wu, and J. Domingo- Ferrer, "Simulatable certificateless two-party authenticated key agreement protocol," *Information Sciences*, vol. 180, no. 6, pp. 1020–1030, 2010.

[10] X. Cao, W. Kou, and X. Du, "A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges," *Information Sciences*, vol. 180, no. 15, pp. 2895–2903, 2010.

[11] M. Joye and G. Neven, *Identity-Based Cryptography*, IOS Press, 2009.

[12] R. W. Zhu, G. Yang, and D. S. Wong, "An efficient identity-based key exchange protocol with KGS forward secrecy for low-power devices," *Theoretical Computer Science*, vol. 378, no. 2, pp. 198–207, 2007.

[13] X. Cao, W. Kou, Y. Yu, and R. Sun, "Identity-based authenticated key agreement protocols without bilinear pairings," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E91-A, no. 12, pp. 3833–3836, 2008.

[14] D. He, Y. Chen, J. Chen, R. Zhang, and W. Han, "A new two-round certificateless authenticated key agreement protocol without bilinear pairings," *Mathematical and Computer Modelling*, vol. 54, no. 11-12, pp. 3143–3152, 2011.

[15] Chinese Government State Cryptography Administration, "Public key cryptographic algorithm SM2 based on elliptic curves," (Chinese), 2010, http://www.oscca.gov.cn/UpFile/20101222148226692.pdf.

[16] Chinese Government State Cryptography Administration, "Chinese government state cryptography administration no: 24 announcement," 2012 (Chinese), http://www.oscca.gov.cn/News/201212/News_1234.htm.

[17] Chinese Government State Cryptography Administ ration and Chinese Administration of Customs, "Chinese government st ate cryptography administration no.64 announcement," 2012, http://www.oscca.gov.cn/WebSite/smb/Upload/File/201301/20130125170704188.pdf.

[18] J. Xu and D. Feng, "Comments on the SM2 key exchange protocol," in *Cryptology and Network Security*, vol. 7092 of *Lecture Notes in Computer Science*, pp. 160–171, Springer, Berlin, Germany, 2011.

[19] C. Boyd and K. K. R. Choo, "Security of two-party identity-based key agreement," in *Progress in Cryptology—Mycrypt 2005*, vol. 3715 of *Lecture Notes in Computer Science*, pp. 229–243, Springer, Berlin, Germany, 2005.

[20] C. Boyd and A. Mathuria, *Protocols for Authentication and Key Establishment*, Springer, Berlin, Germany, 2003.

[21] K.-K. R. Choo, *Secure Key Establishment*, vol. 41 of *Advances in Information Security*, Springer, 2009.

[22] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Advances in Cryptology—CRYPTO 1993*, vol. 773 of *Lecture Notes in Computer Science*, pp. 232–249, Springer, Berlin, Germany, 1994.

[23] M. Bellare and P. Rogaway, "Provably secure session key distribution—the three party case," in *Proceedings of of 27th ACM Symposium on Theory of Computing*, pp. 57–66, 1995.

[24] G. Yang and C. Tan, "Strongly secure certificateless key exchange without pairing," in *Proceedings of the 6th International Symposium on Information, Computer and Communications Security (ASIACCS '11)*, pp. 71–79, Hong Kong, March 2011.

[25] Y. Chen and W. Han, "Efficient identity-based auth enticated multiple key exchange protocol," *Acta Scientiarum, Technology*, vol. 35, no. 4, pp. 629–636, 2013.

[26] D. Boneh and R. Lipton, "Algorithms for black-box fields and their application to cryptography," in *Advances in Cryptology—CRYPTO '96*, vol. 1109 of *Lecture Notes in Computer Science*, pp. 283–297, Springer, Berlin, Germany, 1996.