Research Article

Authenticated Diffie-Hellman Key Agreement Scheme that Protects Client Anonymity and Achieves Half-Forward Secrecy

Hung-Yu Chien

Department of Information Management, National Chi-Nan University, 470 University Road, Puli, Nantou, Taiwan

Correspondence should be addressed to Hung-Yu Chien; hychien@ncnu.edu.tw

Received 3 January 2015; Revised 30 March 2015; Accepted 12 April 2015

Academic Editor: Francesco Gringoli

Copyright © 2015 Hung-Yu Chien. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Authenticated Diffie-Hellman key agreement (D-H key) is the de facto building block for establishing secure session keys in many security systems. Regarding the computations of authenticated D-H key agreement, the operation of modular exponentiation is the most expensive computation, which incurs a heavy loading on those clients where either their computational capacities or their batteries are limited and precious. As client's privacy is a big concern in several e-commerce applications, it is desirable to extend authenticated D-H key agreement to protect client's identity privacy. This paper proposes a new problem: the modified elliptic curves computational Diffie-Hellman problem (MECDHP) and proves that the MECDHP is as hard as the conventional elliptic curves computational Diffie-Hellman problem (ECDHP). Based on the MECDHP, we propose an authenticated D-H key agreement scheme which greatly improves client computational efficiency and protects client's anonymity from outsiders. This new scheme is attractive to those applications where the clients need identity protection and lightweight computation.

1. Introduction

Authentication is an essential security; however, many existent authentication schemes either did not provide key agreement during authentication [1–3], did not protect client privacy [4], or required many expensive modular exponentiations [4–16]. Authenticated key agreement aims at simultaneously providing authentication of communicating parties and establishing secure session keys. Nowadays it is very easy for any network operators or even outsiders to collect users' behavior and violate their privacy when they surf the Internet if key agreement schemes do not protect users' identities. It is, therefore, imperative to design key agreement schemes which protect client's identity. With such schemes, users' privacy can be protected when they enjoy Internet activities, e-commerce or m-commerce transactions.

Conventionally, the computational Diffie-Hellman problem over Galois field (called CDHP) and the same problem over elliptic curves (called ECDHP) are the most popular building blocks for many authenticated key agreement schemes [4–8, 10–19], owing to their hardness. However, the modular exponentiation computations over Galois field or the point multiplications over elliptic curves impose a heavy computational stress on those clients where either

their computing capacities or their batteries are limited. Such clients are called thin clients in the rest of this paper. Even a native D-H key scheme (not including authentication of communicating parties) that uses the CDHP requires of each party two modular exponentiations, and the corresponding version over elliptic curves would require of each party two point multiplications. Generally an authenticated version of D-H key scheme or an extended version of D-H key scheme which protects client's anonymity would require more modular exponentiations or more point multiplications [9–12, 14–16, 20], respectively. *It is, therefore, important to reduce the number of exponentiation computations/point multiplications for those thin clients.* In 2014, Chien [4] formulated the modified computational D-H problem (MCDHP) and proposed an authenticated D-H key agreement scheme, using the MCDHP. The scheme [4] effectively reduced the number of modular exponentiations but it did not protect client's anonymity.

Conventionally, key agreement schemes transmit participants' identities in plaintext. As privacy has been a big concern in many applications, it is desirable to protect participants' identities during the key agreement process. Such kinds of key agreement schemes are called authenticated

key agreement schemes with client anonymity (or simply an anonymous authenticated key agreement). Chien [12] classified four types of two-party key agreement schemes, according to the protection of participants' anonymity. Type 1: the privacy of identities of two communicating parties is not protected; this type corresponds to those conventional two-party key agreement schemes. Type 2: the client's identity is protected from the outsiders, but the identity of the server is not protected. Type 3: the client's identity is protected from the outsiders, but the anonymity of the server is only protected from unregistered entities. The scenarios for such a type are like that; in a mission-oriented ad hoc network, the clients and the servers want to protect their identities from outsiders, while all the preregistered clients know the IP address or MAC address of the servers. Type 4: both the identity of the client and the identity of the server are protected from outsiders. Type 2 is the most popular one for authenticated key agreement schemes that protect participant's anonymity, because it corresponds to the cases where clients want to protect their anonymity from outsiders. In this paper, we focus on authenticated two-party key agreement for Type 2 cases and aim to improve client's computational efficiency.

Authentication with anonymity protection is a popular topic, and there are some popular techniques for achieving client's anonymity during authentication process. Chien [2] classifies existing techniques into four categories: probabilistic encryption-based scheme like the OAKley protocol [13], pseudonym-based schemes like [20], hash-chain-based scheme like Ohkubo et al.'s scheme [1], and error-correction-codes-based scheme like [2, 3]. It is challenging to extend the existent CDHP-based key agreement schemes like [5–7, 13] (or ECDHP-based schemes like [8]) to their anonymous versions and reduce the modular exponentiation load (or the point multiplication load). In this paper, we first formulate a modified ECDHP (MECDHP) and prove its security. Then we propose a new authenticated two-party key agreement scheme with client anonymity, based on the MECDHP problem and Ohkubo et al.'s hashing chain technique [1]. *The new scheme protects client anonymity, effectively reduces the computational load for client, and preserves the strong security of the ECDHP.*

The rest of this paper is organized as follows. Section 2 introduces the MECDHP and proves its security. Section 3 proposes our new scheme. Section 4 examines its security and evaluates its performance. Section 5 states our conclusions and discussions.

2. The MECDHP Problem and the Security Requirements

In this section, we first propose the MECDHP and prove its security. Then we introduce the model and discuss the security requirements of an authenticated key agreement scheme with client anonymity.

Elliptic Curves over $GF(p)$. A nonsupersingular elliptic curve $E(F_p)$ is the set of points $P = (x, y)$ and the point O (called

the *point at infinity*), where $x, y \in Z_p$ satisfy the equation $y^2 \equiv x^3 + ax + b \pmod{p}$ ($a, b \in Z_p$ are constants, such that $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$). Two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on the elliptic curve E can be added together using the following rule: if $x_2 = x_1$ and $y_2 = -y_1$, then $P + Q = O$; otherwise, $P + Q = (x_3, y_3)$, where $x_3 = \lambda^2 - x_1 - x_2 \pmod{p}$, $y_3 = \lambda(x_1 - x_3) - y_1 \pmod{p}$, and $\lambda = (y_2 - y_1)/(x_2 - x_1)$ if $P \neq Q$ or $\lambda = (3x_1^2 + a)/(2y_1)$ if $P = Q$.

Definition 1. The computational elliptic curve Diffie-Hellman problem (ECDHP) is as follows: given an elliptic curve over a finite field F_p , a point $P \in E(F_p)$ of order q and points $A = aP$ and $B = bP \in \langle P \rangle$ find the point $C = abP$.

Now we formulate a new problem called the modified computational elliptic curve Diffie-Hellman problem (MECDHP) as follows.

Definition 2. The modified computational elliptic curve Diffie-Hellman problem (the MECDHP) is as follows: given an elliptic curve over a finite field F_p , a point $P \in E(F_p)$ of order q , $a + x$, and points $A = xP$ and $B = bP \in \langle P \rangle$ find the point $C = abP$.

We prove the hardness of the MECDHP to be as hard as the ECDHP as follows.

Theorem 3. *The MECDHP is as hard as the ECDHP.*

Proof. We prove this by reduction.

(1) *The MECDHP Is Reduced to the ECDHP.* Given an instance of the MECDHP problem ($x + a$, P , xP , and bP), we can compute $(x + a)P - xP = aP$ and get the instance (P , aP , and bP) for the ECDHP. Assume there is one oracle that can answer the ECDHP. Now we input the instance (P , aP , and bP) to the oracle, and we get the answer abP .

(2) *The ECDHP Is Reduced to the MECDHP.* Assume there is one oracle that can answer the MECDHP: given $(x + a$, P , xP , and bP), it outputs abP .

Now given an instance of the CDHP (P , aP , and bP), we then choose a random value t and input the instance (t , P , aP , and bP) to the MECDHP oracle. The oracle will answer $b(tP - aP) = b(t - a)P = tbP - abP$. Using the response, we can derive $-(tbP - abP - t(bP)) = -(-abP) = abP$. That is, we get the answer for the ECDHP problem (P , aP , and bP).

Based on the above arguments, we prove the theorem. $\square$

Now we introduce the model and discuss the security requirements of authenticated D-H key agreement scheme with client anonymity as follows. Our model consists of three kinds of entities: clients, servers, and outsiders. Clients would like to establish secure session keys with servers via key agreement schemes and their identities should not be learned by any outsiders. An outsider can actively manipulate the communications via replay, modification, or interception.

(1) Consider Mutual authentication of client and server and resistance to various attacks like replay attack, impersonation attack, man-in-the-middle attack, and known key attack.

(2) Consider Partial forward secrecy and perfect forward secrecy. Here partial forward secrecy requires that even if we assume one party's long-term private key is disclosed someday, the previous communications (the session keys before the disclosure) are still secure. If partial forward secrecy is preserved only when one specific party's private key is compromised but it does not hold for the other party, then it is called half forward secrecy. While perfect forward requires that even if we assume both of the two parties' long-term private keys are disclosed someday, the previous communications (the session keys before the disclosure) are still secure.

(3) Anonymity of the client: the identities of clients should be well protected from outsiders.

3. New Authenticated D-H Key Agreement Scheme with Client Anonymity

We introduce the following notations. We will omit the mod q operation in the rest of this paper to simplify the presentation when the context is clear.

The Notations.

$E(F_p)$, P , and q are as follows: $E(F_p)$ is an elliptic curve over F_p . $P \in E(F_p)$ is a generator point for a group over $E(F_p)$.

$h()$, $g()$ are two cryptographic hash functions. Two different hash functions are chosen here to hinder outsiders from tracking users, using correlated data. This idea is inspired by [1]. These functions could be implemented using a pseudorandom function (PRF) with distinct paddings.

C , S are as follows: C and S , respectively, denote client and server.

ID_C , ID_S are as follows: ID_C and ID_S , respectively, denote the identity of the client and that of the server, where ID_S is static while ID_C is dynamically updated.

c , s are as follows: $c, s \in Z_q^*$, respectively, denote the private key of C and that of S .

P_C , P_S are as follows: $P_C = cP$ and $P_S = sP$, respectively, denote their corresponding public keys.

x , y are as follows: $x, y \in Z_q^*$, respectively, denote ephemeral private keys.

X , Y are as follows: $X = xP$, and $Y = yP$ denotes their corresponding public keys.

$\oplus$, $\parallel$ are as follows: $\oplus$ denotes the exclusive OR operation. $\parallel$ denotes concatenation. Here we abusively use the notation $\oplus$ between two elliptic curve points to represent $(x_1, y_1) \oplus (x_2, y_2) = (x_1 \oplus x_2, y_1 \oplus y_2)$.

The scheme is depicted in Figure 1 and is described as follows.

The scheme consists of two phases: initial phase and key agreement phase.

The Initialization Phase. Each registered client C shares its identity ID_C and public key P_C with the server S , where S keeps two values $ID_{C,old}$ and $ID_{C,new}$ to track the dynamically changing identity of the client. Both $ID_{C,old}$ and $ID_{C,new}$ are initialized as ID_C . Each client keeps the identity and the public key P_S of the server.

The Key Agreement Phase.

(1) $S \rightarrow C: ID_S, Y$.

The server randomly chooses an integer $y \in_R Z_q^*$, computes $Y = yP$, and sends (ID_S, Y) to the client.

(2) $C \rightarrow S: g(ID_C), c + x, Auth_C$.

The client chooses a random integer $x \in_R Z_q^*$ and computes $K_{C,S} = xY$, $BL = xP_S$, and $Auth_C = h((P_C + K_{C,S}) \oplus BL)$. He sends $(g(ID_C), c + x, Auth_C)$ to the server.

(3) $S \rightarrow C: Auth_S$.

Upon receiving the client's data, the server first searches its data base to look up the entry that satisfies either $g(ID_C) \stackrel{?}{=} g(ID_{C,old})$ or $g(ID_C) \stackrel{?}{=} g(ID_{C,new})$. The matched entry ($ID_{C,old}$ or $ID_{C,new}$) is referred as $ID_{C,matched}$. The server uses the matched entry to get the corresponding public key P_C and computes $X = (c + x)P - P_C = xP$, $BL = sX = sxP$, and $K_{C,S} = yX = xyP$. It then checks whether the equation $h((P_C + K_{C,S}) \oplus BL) \stackrel{?}{=} Auth_C$ holds. If the verification succeeds, then it updates $ID_{C,old} = ID_{C,matched}$ and $ID_{C,new} = h(ID_{C,matched})$ and computes $Auth_S = h(P_C \oplus K_{C,S} \oplus BL)$.

(4) C : the client uses its local values to verify the validity of $Auth_S$. If the verification succeeds, then it updates its identity ID_C as $h(ID_C)$.

The final session key of the session is computed using any secure hash function: one example is like $h(P_C, P_S, x + c, Y, K_{C,S})$.

4. Security Analysis and Performance Evaluation

4.1. Security Analysis. In this section, we examine the security of the proposed authenticated D-H key agreement scheme with client anonymity. The formulated MECDHP used in the scheme has been proved to be as hard as the conventional ECDHP. Our new scheme improves the computational efficiency of client while preserving the strong security properties of conventional secure D-H key agreement schemes under the assumption that the communicating parties' long-term secret keys are secure. But, the security properties might

FIGURE 1: Authenticated D-H key agreement with client anonymity.

change when client's long-term secret keys are compromised. We examine these properties as follows.

Authentication of the Client. The authentication of the client depends on its ability of providing a valid $Auth_C = h((P_C + K_{C,S}) \oplus BL)$ in response to server's challenge Y . To have a valid $Auth_C$, the client should be able to provide *well-formed* " $x+c$," on which the server can correctly derive the corresponding $X = (c + x)P - P_C = xP$, $BL = sX = sxP$, and $K_{C,S} = yX = xyP$ to verify the $Auth_C$. A well-formed " $x+c$ " implies that the client knows the private key c . This ensures the authentication of the client.

Authentication of the Server. The authentication of server depends on server's ability of providing a valid $Auth_S = h(P_C \oplus K_{C,S} \oplus BL)$, which depends on the server's ability of computing $BL = sX$, using $X = (c + x)P - P_C = xP$. To compute $BL = sX$, it requires the knowledge of the server's private key s . This ensures the authentication of the server.

Resistance to Replay Attack and Man-in-the-Middle Attack. The client and the server, respectively, issue new challenges $x+c$ and Y in each session. The server is expected to generate the corresponding value $BL = sX = s((x + c)P - P_C)$, where the value is derived from the well-formed challenge $x+c$. The client is expected to have the ability of computing the value $K_{C,S} = xY$, where Y is the challenge. All of the computations involve new challenges. As long as the challenges are random and fresh, this ensures the resistance to replay attack and man-in-the-middle attack.

Half Forward Secrecy. Partial forward secrecy concerns the privacy of session keys corresponding to those sessions before one party's private key was compromised.

- (1) If we assume that the server's long-term secret s is compromised, then the forward privacy of the session

key $h(P_C, P_S, x + c, Y, K_{C,S})$ is still preserved, because the computation of the session key involves $K_{C,S}$. Here the computation of $K_{C,S}$ needs the knowledge of either x or y . This ensures the partial forward secrecy against server's compromise.

- (2) If we assume the client's long-term secret c is compromised, then the attacker can derive the ephemeral secret x from $x + c$ and derive the session key; that is, our scheme does not preserve partial forward secrecy against client's compromise. So we call that our scheme owns *half forward secrecy* property but not perfect forward secrecy.

Client Anonymity. The client-identity-related data transmitted in our protocol include $g(ID_C)$, $Auth_C = h((P_C + K_{C,S}) \oplus BL)$, and $Auth_S = h(P_C \oplus K_{C,S} \oplus BL)$. The identity ID_C is updated as $h(ID_C)$ per successful authentication, and only its hashed value $g(ID_C)$ is transmitted. Therefore, an attacker could not learn clients' identities and could not track any client using eavesdropped data. The other identity-related data is client's public key P_C . The calculations of $Auth_C$ and $Auth_S$ first obscure P_C with random value BL and then apply hashing: this ensures that attackers cannot link the transmissions to any specific client.

4.2. Performance Evaluation. Now we examine the performance of our scheme in terms of communication cost and computational cost. Regarding communication cost, we concern the number of message steps and the length of message length. Our scheme requires three message steps. It demands the message length $3L_h + 2L_{point} + 1L_q$, where L_h denotes the length of hash function ($|h()|$ and $|g()|$), L_{point} denotes the length of string representation of one elliptic curve point, and L_q denotes the bit length of q .

TABLE 1: Summary of performance comparison among two-party key agreement schemes with client anonymity.

Scheme	Ours.	Yoon-You [11]	Chien [12]	Yang et al. [10]	Wu-Hsu [14]	Wang et al. [15]
Number of steps	3	3	3	3	3	3
Length of message	$3L_h + 2L_{\text{point}} + 1L_q$	$3L_N + 1L_{\text{Time}} + 1L_{\text{ID}}$	$3L_N + 1L_{\text{Time}} + 1L_{\text{ID}}$	$3L_N + 1L_{\text{Time}} + 1L_{\text{ID}}$	$3L_N + 1L_{\text{Time}}$	$5L_{\text{nonce}} + 4L_{\text{point}} + 1L_{\text{ID}}$
Computational cost of client	$2T_{\text{PS}} + T_{\text{PA}} + 3T_{\text{XOR}} + 3T_h$	$5T_{\text{exp}} + 2T_{\text{multi}} + 1T_{\text{enc}}$	$5T_{\text{exp}} + 1T_{\text{multi}} + 1T_{\text{enc}} + 2T_h$	$5T_{\text{exp}} + 3T_{\text{multi}} + 1T_{\text{enc}} + 1T_h$	$4T_{\text{exp}} + 3T_{\text{multi}} + 1T_h$	$4T_{\text{PS}} + 2T_{\text{enc}} + 3T_h$
Computational cost for server	$4T_{\text{PS}} + 2T_{\text{PA}} + 3T_{\text{XOR}} + (n+2)T_h$	$5T_{\text{exp}} + 2T_{\text{multi}} + 1T_{\text{enc}}$	$5T_{\text{exp}} + 2T_{\text{multi}} + 1T_{\text{enc}} + 2T_h$	$5T_{\text{exp}} + 2T_{\text{multi}} + 1T_{\text{enc}} + 1T_h$	$4T_{\text{exp}} + 2T_{\text{multi}} + 1T_h$	$4T_{\text{PS}} + 2T_{\text{enc}} + 3T_h$
Security properties	Mutual auth.	Mutual auth.	Mutual auth.	Client/server impersonation [11]	Client/server impersonation [11]	Mutual auth.
Client anonymity	Yes	Fail (Note 5)	Yes	Yes	Yes	Yes
Forward secrecy	Half forward secrecy	Yes	Yes	Yes	Yes	Note 3
NP problem	MCDHP	RSA, CDHP	RSA, CDHP	RSA, CDHP	RSA, CDHP	Note 3

Note 1: L_N : the bit length of RSA modulus N ; L_h : the length of hash function ($|h()|$ and $|g()|$); L_{point} : the length of string representation of one elliptic curve point, and L_q denotes the bit length of q . L_{Time} : the bit length of timestamp; L_{ID} : the length of identity; L_{nonce} : the length of nonce.

Note 2: T_{PS} : time complexity of one elliptic curve point multiplication; T_{PA} : time complexity for one elliptic curve point addition; T_{XOR} : time complexity for one XOR operation; T_h : time complexity for one hash operation; T_{exp} : time complexity of one modular exponentiation; T_{multi} : time complexity of one modular multiplication; T_{enc} : time complexity of one symmetric encryption.

Note 3: The session key security was based on the privacy of nonce and the fixed D-H key (the fixed key $sP_C = cP_C = csP$). If one private key of either the server or the client is compromised, then the long-term, fixed D-H key is compromised and one of the nonce is compromised; therefore, its security is not strong as the ECDHP.

Note 4: Our scheme applies hashing to protect the dynamic identity while other schemes use encryption of identity to protect anonymity; therefore, we can simplify the comparison by assuming $L_h = L_{\text{ID}}$.

Note 5: Chien [12] reported the failure of client anonymity of Yoon-You's scheme [11].

Next we examine the computational cost of our scheme. The client in our scheme demands $2T_{\text{PS}} + 1T_{\text{PA}} + 3T_{\text{XOR}} + 3T_h$, where T_{PS} denotes the time complexity of one elliptic curve point multiplication, T_{PA} denotes that of one elliptic curve point addition, T_{XOR} denotes that of one XOR operation, and T_h denotes that of one hash operation. The server averagely needs $4T_{\text{PS}} + 2T_{\text{PA}} + 3T_{\text{XOR}} + (n+2)T_h$ to find the matched client (each trial for one potential client takes two hash operations), where n denotes the number of entries of potential clients in the database.

Table 1 summarizes the performance comparison of the proposed scheme with its counterparts. The notations used in the table are introduced in the bottom of the table. The security of the schemes [10–12, 14] is based on RSA and the CDHP problem, and our scheme is based on the MECDHP problem. *Please note that even though the authors [15] claimed that their scheme was based on the ECDHP problem but actually its session key security was based on the privacy of nonce and the fixed D-H key (the fixed key $sP_C = cP_C = csP$), if one private key of either the server or the client is compromised, then the fixed D-H key and one nonce are compromised; therefore, its security is not strong as the ECDHP.*

Regarding the communicational performance, we focus on the number of message steps and the length of the messages. According to National Security Agency of the USA [18], to have an equivalent security of an 80-bit symmetric encryption, RSA algorithm and Diffie-Hellman algorithm need 1024-bit key while ECC only needs 160-bit key. From the table, we can see that all the schemes require three message steps. Because the schemes [9–12, 14] used RSA and the

CDHP, their message length is larger than our scheme and the scheme [15]. And, the message length of our scheme is shorter than its ECC-based counterpart [15].

Now we compare the computational performance. Recently there have been several publications like [21–24] discussing the implementation performances of ECC standard and other public-key standards (like RSA and El-Gamal public key), where [21] compared various software implementations of point multiplication, [22] aimed at improving hardware implementation of point multiplication, [23] discussed the hardware implementation performance of RSA digital signature and ECDSA, and [24] compared implementation performance of El-Gamal elliptic curve encryption method using two different libraries. These works focused on the implementation performances of elliptic curve point multiplication and related standards (like ECC standards, RSA digital signature, and El-Gamal elliptic curve encryption). For a fair comparison of custom-designed algorithms that consist of underlying field operations (like field multiplication and exponentiation in Z_p , field multiplication in Z_q , and elliptic curve point multiplication), the figures in NSA [18] and the algebra equations of elliptic curve operations in terms of underlying field operations from [19] have been widely adopted [15]. We adopt the same principle in the following evaluation, since it gives a fair comparison of custom-designed algorithms.

The security of ECC with 160-bit key is equivalent to that of RSA with 1024-bit key or D-H algorithm with 1024-bit key. Under the above figures, $T_{\text{multi},p}$ (the time complexity of a field multiplication in Z_p , where p is 1024-bit) is 41 times

$T_{\text{multi},q}$ (the time complexity of field multiplication in Z_q , where q is 160-bit), $T_{\text{PS}} \sim 29T_{\text{multi},p}$, $T_{\text{exp}} \sim 240T_{\text{multi},p} \sim 8T_{\text{PS}}$, $T_{\text{PA}} \sim 0.12T_{\text{multi},p}$, and $T_{\text{PS}} \sim 241T_{\text{PA}}$, where $\sim$ means “roughly equal.” To simplify the comparison and get an insight of the computational performance, we can focus on the number of ECC point multiplications, point addition, modular exponentiation, and modular multiplication only because the other operations are not computationally significant. In this simplification, the client in our scheme needs $2T_{\text{PS}} + T_{\text{PA}} \sim 58.12T_{\text{multi},p}$, the client in [11] needs $5T_{\text{exp}} + 2T_{\text{multi},p} \sim 1202T_{\text{multi},p}$, the client in [12] takes $5T_{\text{exp}} + 1T_{\text{multi},p} \sim 1201T_{\text{multi},p}$, the client in [10] requires $5T_{\text{exp}} + 3T_{\text{multi},p} \sim 1203T_{\text{multi},p}$, the client in [14] demands $4T_{\text{exp}} + 3T_{\text{multi},p} \sim 9633T_{\text{multi},p}$, and the client in Wang et al.’s scheme [15] requires $4T_{\text{PS}} \sim 116T_{\text{multi},p}$. Based on these figures, we can get an insight that the client in our scheme only takes roughly 1/20 computational complexity of those RSA-based schemes like [10–12, 14] and takes only 1/2 time complexity of Wang et al.’s ECC-based scheme [15].

5. Conclusions and Further Investigations

This paper has formulated a new problem called the MECDHP and has proved its security being equivalent to the conventional ECDHP. Based on the MECDHP, we have proposed a new authenticated D-H key agreement scheme which protects client anonymity and greatly improves client’s computational efficiency. Based on the figures of 160-bit ECC key and 1024-bit RSA key (or 1024-bit D-H algorithm), the client of our scheme requires only 1/20 time complexity of its RSA-based counterparts [10–12, 14] and takes only 1/2 time complexity of Wang et al.’s ECC-based scheme [15]. Security analysis shows that the scheme achieves half forward secrecy: it achieves forward secrecy if server’s private key is compromised but the session keys of a compromised client would be disclosed. Here we note that it only affects the session keys of compromised clients. These excellent performances make our scheme quite attractive to those thin clients that require anonymity protection. One interesting open issue is whether we can improve the computational efficiency of existent authenticated D-H key agreement schemes while still preserving the perfect forward secrecy.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This project is partially supported by the National Science Council, Taiwan, under Grant no. MOST 103-2221-E-260-022.

References

- [1] M. Ohkubo, K. Suzki, and S. Kinoshita, “Cryptographic approach to ‘privacy-friendly’ tags,” in *Proceedings of the*

RFID Privacy Workshop, Massachusetts Institute of Technology, November 2003.

- [2] H.-Y. Chien and C.-S. Lai, “ECC-based lightweight authentication protocol with untraceability for low-cost RFID,” *Journal of Parallel and Distributed Computing*, vol. 69, no. 10, pp. 848–853, 2009.
- [3] H.-Y. Chien, “Combining Rabin cryptosystem and error correction codes to facilitate anonymous authentication with untraceability for low-end devices,” *Computer Networks*, vol. 57, no. 14, pp. 2705–2717, 2013.
- [4] H. Y. Chien, “Provably secure authenticated Diffie-Hellman key exchange for resource-limited smart card,” *Journal of Shanghai Jiaotong University (Science)*, vol. 19, no. 4, pp. 436–439, 2014.
- [5] A. Brusilovsky, I. Faynberg, Z. Zeltsan, and S. Patel, “RFC683-Password-Authenticated Key (PAK) Diffie-Hellman Exchange,” 2010, <http://tools.ietf.org/html/rfc683>.
- [6] V. Boyko, P. MacKenzie, and S. Patel, “Provably secure password-authenticated key exchange using Diffie-Hellman,” in *Advances in Cryptology—EUROCRYPT 2000*, vol. 1807 of *Lecture Notes in Computer Science*, pp. 156–171, Springer, Berlin, Germany, 2000.
- [7] ISO/IEC 9798-3 Authentication SASL Mechanism, <http://www.faks.org/rfc/rfc3163.html>.
- [8] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Moeller, “Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS),” RFC 4492, Internet Engineering Task Force (IETF), 2006.
- [9] W.-B. Lee and C.-C. Chang, “User identification and key distribution maintaining anonymity for distributed computer networks,” *Computer Systems Science and Engineering*, vol. 15, no. 4, pp. 113–116, 2000.
- [10] Y. Yang, S. Wang, F. Bao, J. Wang, and R. H. Deng, “New efficient User identification and key distribution scheme providing enhanced security,” *Computers and Security*, vol. 23, no. 8, pp. 697–704, 2004.
- [11] E.-J. Yoon and K.-Y. Yoo, “Cryptanalysis of two user identification schemes with key distribution preserving anonymity,” in *Information and Communications Security: 7th International Conference, ICICS 2005, Beijing, China, December 10–13, 2005. Proceedings*, vol. 3783 of *Lecture Notes in Computer Science*, pp. 315–322, Springer, Berlin, Germany, 2005.
- [12] H.-Y. Chien, “Practical anonymous user authentication scheme with security proof,” *Computers & Security*, vol. 27, no. 5-6, pp. 216–223, 2008.
- [13] P. Szalachowski and Z. Kotulski, “Enhancing the Oakley key agreement protocol with secure time information,” in *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS ’12)*, pp. 1–8, Genoa, Italy, July 2012.
- [14] T.-S. Wu and C.-L. Hsu, “Efficient user identification scheme with key distribution preserving anonymity for distributed computer networks,” *Computers and Security*, vol. 23, no. 2, pp. 120–125, 2004.
- [15] R.-C. Wang, W.-S. Juang, and C.-L. Lei, “Provably secure and efficient identification and key agreement protocol with user anonymity,” *Journal of Computer and System Sciences*, vol. 77, no. 4, pp. 790–798, 2011.
- [16] M. S. Farash, M. Bayat, and M. A. Attari, “Vulnerability of two multiple-key agreement protocols,” *Computers and Electrical Engineering*, vol. 37, no. 2, pp. 199–204, 2011.

- [17] N.-Y. Lee, C.-N. Wu, and C.-C. Wang, "Authenticated multiple key exchange protocols based on elliptic curves and bilinear pairings," *Computers and Electrical Engineering*, vol. 34, no. 1, pp. 12–20, 2008.
- [18] US National Security Agency, The Case for Elliptic Curve Cryptography, <https://www.nsa.gov/business/programs/elliptic-curve.shtml>.
- [19] A. Jurisic and A. J. Menezes, "Elliptic curves and cryptography," Certicom Whitepaper, 1997.
- [20] K. Xue, P. Hong, and C. Ma, "A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture," *Journal of Computer and System Sciences*, vol. 80, no. 1, pp. 195–206, 2014.
- [21] M. Rivain, "Fast and regular algorithms for scalar multiplication over elliptic curve," The International Association for Cryptologic Research (IACR) Cryptology ePrint Archive 2011/338, <https://eprint.iacr.org/2011/338.pdf>.
- [22] G. D. Sutter, J.-P. Deschamps, and J. L. Imana, "Efficient elliptic curve point multiplication using digit-serial binary field operations," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 1, pp. 217–225, 2013.
- [23] R. Sinha, H. K. Srivastava, and S. Gupta, "Performance based comparison study of RSA and elliptic curve cryptography," *International Journal of Scientific & Engineering Research*, vol. 4, no. 5, pp. 720–725, 2013.
- [24] D. F. Pigatto, N. B. F. d. Silva, and K. R. L. J. C. Branco, "Performance evaluation and comparison of algorithms for elliptic curve cryptography with el-gamal based on MIRACL and RELIC libraries," *Journal of Applied Computing Research*, vol. 1, no. 2, pp. 95–103, 2012.