Wireless Pers Commun (2014) 79:1975–1991
DOI 10.1007/s11277-014-1968-8

A Provably Secure ID-Based Mutual Authentication and Key Agreement Scheme for Mobile Multi-Server Environment Without ESL Attack

SK Hafizul Islam

Published online: 2 August 2014

Abstract In mobile multi-server authentication, a client can access different servers over an insecure channel like Internet and wireless networks for numerous online applications. In the literature, several multi-server authentication schemes for mobile clients have been devised. However, most of them are insecure against ephemeral secret leakage (ESL) attack and other vulnerabilities. For mutual authentication and key agreement, mobile client and server used ephemeral secrets (random numbers) and leakage of these secrets may be possible in practice. Since these are generated by an external source that may be controlled by an adversary. Also they are generally pre-computed and stored in insecure devices. Thus, if the secrets are leaked then the session key would turn out to be known and the private keys of client and server may be compromised from the eavesdropped messages. This phenomenon is called ESL attack. To defeat the weaknesses, in this paper, we design an ESL attack-free identity-based mutual authentication and key agreement scheme for mobile multi-server environment. The proposed scheme is analyzed and proven to be provably secure in the random oracle model under the Computational Diffie–Hellman assumption.

Keywords Multi-server authentication · Mobile client · ESL attack · Hash function · Elliptic curve cryptography · ID-based cryptography · Bilinear pairing

1 Introduction

With the tremendous development of Network technologies and the availability of portable mobile devices (e.g., PDA, mobile phone, notebook PC), people are accessing a number of servers using their mobile devices for different purpose like online shopping, online pay-TV, online bill payment, online banking transaction, file sharing, online game, distributed electronic medical records system, etc. [1–3]. In order to get access remote server, a mutual

S. H. Islam (✉)
Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani,Pilani 333031, Rajasthan, India
e-mail: hafi786@gmail.com; hafizul.ism@gmail.com; hafizul@pilani.bits-pilani.ac.in

authentication with session key agreement scheme is required in which a client first authenticates himself/herself to the server and then the client verifies the legitimacy of the server. After mutual authentication, both the client and the remote server compute a common session key, which will be used for secure information exchange between them in subsequent communications. To achieve this goal, various single server authentication systems [4–8] have been proposed in the literature. However, this architecture is not always suitable in some situation if client wants service from numerus servers. Based on single server authentication architecture, it is very difficult for a client to register himself/herself to multiple servers with different secrets (e.g., password or private key). To address these issues, varieties of multi-server authentication architecture have been proposed by the research community [9–13]. In this architecture, client(s) and remote server(s) first register to a registration center (RC) and then client(s) accesses all the servers using only one secret over Internet and wireless networks. It is known that the Internet and wireless networks are prone to errors and insecure since no integrated security mechanisms are available from their inception. Also the energy resources, storage and computing capability of mobile devices are very limited, therefore the design of a efficient and secure multi-server authentication scheme suitable for mobile environment is a challenging task.

1.1 Related Studies

In 2008, Tsai [9] proposed a mutual authentication and key agreement (MAKA) scheme for multi-server environment using hash function, which is shown to be insecure against impersonation attack and forward security [14]. In order to achieve client's anonymity, Geng and Zhang [10] proposed a dynamic ID-based MAKA protocol for multi-server environment based on password and bilinear pairing. In 2009, Liao and Wang [11] proposed another dynamic ID-based MAKA protocol for multi-server environment based on password. Later on, Hsiang and Shih [12] found which the scheme [11] is vulnerable to insider attack, masquerade attack, server spoofing attack and registration center spoofing attack. To overcome these weaknesses, Hsiang and Shih proposed an improved scheme that is also found to be susceptible to the masquerade attack and the server spoofing attack [15]. In 2011, Lee et al. [13] analyzed that Hsiang et al.'s MAKA scheme [12] does not provide mutual authentication and resilience against masquerade attack, server spoofing attack, and is not easily repairable. To overcome these weaknesses, Lee et al. proposed an improved scheme. Furthermore, Sood et al. [16] proposed an improved dynamic ID-based MAKA scheme over Hsiang et al.'s scheme and claimed that their protocol provides desired securities. However, Li et al. [17] showed that Sood et al.'s protocol is still vulnerable to leak-of-verifier attack, stolen smart card attack and impersonation attack. Besides, the authentication and session key agreement phase of Sood et al.'s protocol is incorrect. Li et al. [17] then proposed an improved protocol. However, Han [18] demonstrated the protocol [17] is unsuitable for practical applications and insecure against password guessing attack, impersonation attack and replay attack.

In 2013, Li et al. [19] design a smart card and dynamic ID-based MAKA scheme for multi-server environment, which is proven to be insecure against stolen smart card and off-line password guessing attack, replay attack, impersonation attack and server spoofing attack [20]. In 2013, Wang and Ma [21] proposed a password and smart card based MAKA scheme for multi-server environment and demonstrated that their scheme could overcome various attacks. However, He and Wu [22] analyzed that the scheme is vulnerable to the server spoofing attack, the impersonation attack, the privileged insider attack and the off-line password guessing attack. In 2012, Chuang and Tseng [23] proposed a provably secure ID-MAKA protocol for multi-server environment in the random oracle model using elliptic curve bilinear pairing and IBC. In the same year, Han and Zhu [24] proposed an efficient and pairing-free ID-MAKA protocols for multi-server environment using elliptic curve cryptography (ECC) and IBC. They proved that their sachem is provably secure in the random oracle model under the CDH assumption. However, all the scheme discussed in this paper and the newly proposed schemes [23,24] are vulnerable to ESL attack which is proven in the Sect. 6.

1.2 Motivations and Contributions

In the literature, three types of multi-server authentication schemes using password or certificate authority-based public key cryptography (CA-PKC) or identity-based cryptosystem (IBC) have been proposed. In the password based schemes, servers generally maintain the password tables. In addition to that, these schemes are susceptible to the risk of modifying the password table and are insecure against different attacks including online/offline password guessing attack, stolen-verifier attack, denial of service attack, user's impersonation attack, server's masquerade attack [12,14–16,18,20,25]. Furthermore, in the login phase of password-based schemes, authentication server communicates with RC for client authentication and it incurs many rounds and additional communication costs.

On the other hand, CA-PKC based protocols have some other limitations such as it needs a certificate authority (CA), which uses a global public key infrastructure (PKI) to maintain the certificates for users' public keys. For a large number of users, CA requires huge storage space and complex certificate management process to keep up and store the public keys and certificates, and users need to verify the corresponding certificate of others for which extra computations are involved. These problems degrade the overall performance of the system and thus, the CA-PKC based protocols are not suitable for resource constrained mobile devices. However, IBC based schemes removes the problem of CA-PKC based protocols. In IBC setting, user's publicly known identity e.g., email address, rather than a random number, is used as public key and the corresponding private key is generated key by the system's trusted authority, called private key generator (PKG) based on the user's public key and PKG's secret key [26,27].

In the schemes mentioned above, the mobile client and server used ephemeral secrets/random numbers for mutual authentication and session key generation. If the ephemeral secret keys are leaked, an adversary can reveal the session key and the secrets of server and client from the eavesdropped messages. This attack is called ephemeral secret leakage (ESL) attack [28–31] or known session-specific temporary information attack (KSS-TIA) [32–37]. Leakage of these secrets is possible in practical applications since they are generally pre-computed and stored in insecure memory devices and these secrets are generated by an external source that may be controlled by an adversary. If the ephemeral secret is not deleted completely after the protocol execution, an adversary may hijack the sender's computer and get the same [38].

Another most important security requirement of multi-server authentication system is client's anonymity. Most of the multi-server authentication systems available in the literature are designed for either general client [9,21,40,41] or anonymous client [10–13,16,17,19] i.e., they are not suitable for both purpose. In some situation (electronic voting, secret online-order placement, pay TV), client's anonymity (secrecy) is required, otherwise some personal information about the client may be leaked from the static identity. In addition, the security of most of the multi-server authentication schemes are not provably secure in the random oracle model [39] and thus they vulnerable to different known attacks. Thus, the designing of anonymous/dynamic ESL attack-free MAKA scheme based on IBC for both the general and anonymous mobile client [23,24] in the random oracle model is of great concern. In this paper, we designed an ESL attack-free ID-MAKA scheme using elliptic curve bilinear pairing [27,42,43] for mobile multi-server environments. The security of the proposed scheme is analyzed in our security model and shown to be provably secure under the Computational Diffie–Hellman (CDH) assumption.

1.3 Organization of the Paper

The rest of the paper is organized as follows. The bilinear pairing and the related computational problem and its assumption are briefly described in Sect. 2. The Sect. 3 describes the adversarial model of ID-MAKA scheme. The proposed ESL attack-free ID-MAKA scheme for multi-server mobile client is described in Sect. 4. The correctness analysis and provable security analysis of the proposed ESL attack-free ID-MAKA scheme are given in Sect. 5. The comparative analysis of our scheme with others are addressed in Sect. 6. Finally, Sect. 7 concludes the paper.

2 Preliminaries

This section discussed the basics of elliptic curve bilinear pairing and related computational hard problems frequently used in modern cryptography.

2.1 Bilinear Pairing

Let G_q is a subgroup of the additive group of points on an elliptic curve over a finite field E/F_q and G_m is a cyclic subgroup of the multiplicative group over a finite field F_q. We also assume that both G_q and G_m have the same order q (q is a large prime number, where $q \geq 2^k$ and k is a security parameter). The bilinear map $e : G_q \times G_q \rightarrow G_m$ is called admissible bilinear map if it has the following properties:

(1) **Bilinearity:** For any $(P, Q) \in G_q$ and $a, b \in Z_q^*$, we have $e(aP, bQ) = e(P, Q)^{ab}$.
(2) **Non-degenerate:** For all $(P, Q) \in G_q$ such that $e(P, Q) \neq 1_m$ must hold, where 1_m is the identity element of G_m. It means all the pairs in $G_q \times G_q$ do not map to the identity element in G_m, i.e. if P and Q are two generators of G_q then $e(P, Q)$ is a generator of G_m.
(3) **Computability:** There must be an efficient algorithm that can compute $e(P, Q)$ for any $(P, Q) \in G_q$.

The map e will be derived either from the modified Weil pairing or Tate pairing over a finite field [27].

2.2 Computational Problems

In this section, we described the Computational Diffie–Hellman (CDH) Problem on elliptic curve that is are assumed to be intractable by a polynomial-time bounded algorithm. It is to noted that the security of the proposed scheme relies on the difficulty of CDH assumption.

Definition 1 (*Negligible function*) A function $\epsilon(k)$ is said to be negligible if, for every $c > 0$, there exists k_0 such that $\epsilon(k) \leq \frac{1}{k^c}$ for every $k \geq k_0$.

Definition 2 (*Computational Diffie–Hellman (CDH) problem*) Given a random instance (P, aP, bP), where $P \in G_q$, and $a, b \in Z_q^*$, computation of abP is computationally hard by a polynomial-time bounded algorithm. The probability that a polynomial time-bounded algorithm $\mathcal{A}$ can solve the CDH problem is defined as $Adv^{CDH}_{\mathcal{A},G_q} = Pr[\mathcal{A}(P, aP, bP) = abP : P \in G_q; a, b \in Z_q^*]$.

Definition 3 (*Computational Diffie–Hellman assumption*) For any probabilistic polynomial time-bounded algorithm $\mathcal{A}$, $Adv^{CDH}_{\mathcal{A},G_q}$ is negligible.

3 Attack Model of ID-MAKA Scheme

Based on the concept of [23,24,28], a security model for ID-MAKA scheme for multi-server environment is presented in this section. In this adversarial model three roles are involved, called a trusted registration center (RC), n mobile clients $\mathcal{U} = \{U_i : 1 \leq i \leq n\}$ with identitites ID_{Ui} $(1 \leq i \leq n)$ and m authentication servers $\mathcal{S} = \{S_j : 1 \leq i \leq m\}$ with identititiess ID_{Sj} $(1 \leq j \leq m)$. Here we assume that the public parameters and identities $\mathcal{ID} = \{ID_{Ui}, ID_{Sj}: U_i \in \mathcal{U}, S_j \in \mathcal{S}\}$ are known to everyone and each client U_i executes the scheme repeatedly with each server S_j. Instances of U_i (or S_j) model distinct executions of the scheme. We denote Π_i^s is an s^{th} instance of the participant $ID_i \in \mathcal{ID}$. We also assume that a probabilistic polynomial time bounded adversary $\mathcal{A}$ controls the communication channel, i.e., $\mathcal{A}$ can intercept, block, inject, remove, or modify, any messages transmitted in the media. In other words, we can say that all the messages between U_i and S_j are transmitted via $\mathcal{A}$. In order to violate the security of the scheme, adversary $\mathcal{A}$ can ask the following polynomial number of queries:

- **Extract**(ID_i): With this query, $\mathcal{A}$ registers a public key P_i on behalf of any client ID_i of his choice.
- **Send**(Π_i^t, m): Through this query, $\mathcal{A}$ can send a message m to the oracle Π_i^t. On receiving m, the oracle Π_i^t performs some calculation and replies to $\mathcal{A}$ according to the proposed scheme.
- $H_0(x_0)$: Through this query, $\mathcal{A}$ can ask a H_0 hash query for the input x_0 to the oracle Π_i^t. Then Π_i^t chooses a number $y_0 \in_R Z_q^*$, outputs y_0 and then incorporates the tuple $\langle x_0, y_0\rangle$ to the initial-empty list L^{list}_{H0}.
- $H_1(x_1)$: Through this query, $\mathcal{A}$ can ask a H_1 hash query for the input x_1 to the oracle Π_i^t. Then Π_i^t chooses a number $y_1 \in_R Z_q^*$, outputs $y_1 P$ and then incorporates the tuple $\langle x_1, y_1 P\rangle$ to the initial-empty list L^{list}_{H1}.
- $H_2(x_2)$: Through this query, $\mathcal{A}$ can ask a H_2 hash query for the input x_2 to the oracle Π_i^t. Then Π_i^t chooses a number $y_2 \in_R Z_q^*$, outputs y_2 and then incorporates the tuple $\langle x_2, y_2\rangle$ to the initial-empty list L^{list}_{H2}.
- **Reveal**(Π_i^t): This query is executed by $\mathcal{A}$ to obtain a session key SK from the oracle Π_i^t. Now Π_i^t returns the corresponding session key SK if it is accepted, otherwise it outputs a *null* value.
- **Corrupt**(ID_i): Through this query, $\mathcal{A}$ can corrupt a client ID_i and can get the identity-based private key of the corrupted client.
- **Test**$(\Pi_i^t:)$ The adversary $\mathcal{A}$ is allowed to send a *Test* query to the oracle Π_i^t. On receiving a *Test* query, Π_i^t chooses a bit $b \in \{0, 1\}$ and returns the session key if $b = 1$, otherwise outputs a random string $SK \in_R Z_q^*$ as the session key.

Definition 4 (*Partnership*) Two oracles Π_i^t and Π_j^s, where $i \in \mathcal{U}$ and $j \in \mathcal{S}$, are said to be partners if they mutually authenticates each other and computes a common session key between them.

Definition 5 (*Freshness*) An oracle Π_i^t with its partner Π_j^s is said fresh i.e., the participants $i \in \mathcal{U}$ and $j \in \mathcal{S}$ hold a fresh key SK if the following two conditions hold:

(1) Both the oracles Π_i^t and Π_j^s accept SK as session key, but the *Reveal* query has not been invoked by i and j.
(2) It is allowed to ask the *Send*(Π_i^t) query or *Send*(Π_j^s) query after the *Corrupt* query is executed.

Definition 6 (*Unforgeability and AKA security*) An ID-MAKA scheme for multi-server environment offers existential unforgeability and maintains session key secrecy against adaptive chosen identity attacks if no polynomial tome bounded adversary $\mathcal{A}$ has a non-negligible advantage in the following game played between $\mathcal{A}$ and infinite set of oracles Π_i^t for $ID_i \in \mathcal{ID}$.

(1) A private key is assigned to each client and server through the initialization phase related to the security parameter.
(2) $\mathcal{A}$ may ask several queries and get back the results from the corresponding oracles.
(3) There is no *Reveal*(Π_i^t) query or *Corrupt*(ID_i) query being asked before the *Test*(Π_i^t) query has been asked.
(4) $\mathcal{A}$ may ask other queries during asking the *Test*(Π_i^t) query where Π_i^t is fresh. $\mathcal{A}$ outputs its guess bit b' for the bit b which is chosen in the *Test*(Π_i^t) query eventually and the game is terminated.

Definition 7 The authenticated key agreement (AKA) advantage $Adv_{\mathcal{P}}^{AKA}(\mathcal{A}) = |2Pr[\mathcal{A}$ Succeeds$] - 1|$ of the adversary $\mathcal{A}$ is defined as the success of probability to win the above game by violating the AKA security of an execution of a protocol $\mathcal{P}$, if $\mathcal{A}$ asks a single *Test*(Π_i^t) query and correctly guesses a bit b, which is selected by *Test*(Π_j^s) query.

Definition 8 The protocol $\mathcal{P}$ is AKA-secure if $Adv_{\mathcal{P}}^{AKA}(\mathcal{A}) \le \epsilon$, for some negligible function ϵ.

4 The Proposed ESL Attack-Free ID-MAKA Scheme

In this section, we presented an efficient ESL attack-free ID-MAKA scheme for mobile multi-server environment using elliptic curve cryptography and bilinear pairing. The description of the proposed scheme is given below:

4.1 Setup Phase

For given a security parameter $k \in Z^+$, the following actions are taken:

Step 1. RC chooses a k-bit prime number q and the tuple $\langle F_q, E/E_q, G_q, P\rangle$.
Step 2. RC chooses $x \in Z_q^*$ as his master key and $P_{pub} = xP$ as his public key.
Step 3. RC selects a bilinear map $e : G_q \times G_q \to G_m$ and three secure one-way hash functions $H_0 : \{0,1\}^* \to Z_q^*$, $H_1 : \{0,1\}^* \to G_q$ and $H_2 : \{0,1\}^* \to Z_q^*$.
Step 4. RC publishes $\Omega = \langle F_q, E/F_q, e, G_q, P, P_{pub}, H_0, H_1, H_2\rangle$ as system's parameter.

4.2 Extract Phase

Case 1: In this phase, a authentication server S_j, $(1 \le j \le m)$ registers himself/herself to the RC to get his identity-based private key. For this purpose, S_j sends his/her identity ID_{Sj} to RC and then the RC performs the followings:

Step 1. RC chooses a number $y_{Sj} \in_R Z_q^*$ and computes $Y_{Sj} = y_{Sj}P$.
Step 2. RC computes $l_{Sj} = H_0(ID_{Sj}, Y_{Sj})$ and $d_{Sj} = (y_{Sj} + xl_{Sj}) \bmod q$.
Step 3. RC delivers $\langle d_{Sj}, Y_{Sj}\rangle$ to S_j over a out of band (secure) channel. Here, the following approaches are be used:

(a) Off-line mode: In this approach, RC inserts $\langle d_{Sj}, Y_{Sj}\rangle$ into a smartcard and returns it to S_j through a secure channel.
(b) On-line mode: In this approach, S_j connects to RC over Internet and then RC uses Secure Socket Layer (SSL) in the *https* mode to deliver $\langle d_{Sj}, Y_{Sj}\rangle$ to S_j.

Step 4. On receiving $\langle d_{Sj}, Y_{Sj}\rangle$, S_j validates it by checking whether the equation $Y_{Sj} + H_0(ID_{Sj}, Y_{Sj})P_{pub} = d_{Sj}P$ holds. The private key d_{Sj} is valid if the above equation holds and vice-versa. Since we have

$$\begin{aligned} Y_{Sj} + H_0(ID_{Sj}, Y_{Sj})P_{pub} &= y_{Sj}P + (l_{Sj})xP \\ &= (y_{Sj} + xl_{Sj})P \\ &= d_{Sj}P \end{aligned}$$

Step 5. Finally, S_j computes his/her public key as $P_{Sj} = d_{Sj}P$.

Case 2: The client registration phase is quite different from the server's registration phase. According to [23,24], we first list the different validity periods as follows:

Scenario 1 (Long validity period): In this case, a client U_i $(1 \le i \le n)$ generally holds a long validity period and RC maintains an identity revocation list (IDRL) to manage the member revocations connected with the system.
Scenario 2 (Anonymous and short validity period): In this situation, a client U_i $(1 \le i \le n)$ can access anonymously the services (i.e. prepaid mobile phone cards, online prepaid services, guest temporary security cards, etc.) offered by the server S_j $(1 \le j \le m)$.

Similar to the S_j's registration phase, for the Scenario 1, U_i $(1 \le i \le n)$ sends his/her identity ID_{Ui} to RC and obtains his/her private key as follows:

Step 1. RC chooses a number $y_{Ui} \in_R Z_q^*$ and computes $Y_{Ui} = y_{Ui}P$.
Step 2. RC computes $l_{Ui} = H_0(ID_{Ui}, Y_{Ui}, \text{validity period})$ and $d_{Ui} = (y_{Ui}+xl_{Ui}) \bmod q$.
Step 3. RC delivers $\langle d_{Ui}, Y_{Ui}, \text{validity period}\rangle$ to U_i using either off-line or on-line mode as described earlier.
Step 4. Now U_i verifies his/her private key d_{Ui} by checking whether the equation $d_{Ui}P = Y_{Ui} + H_0(ID_{Ui}, Y_{Ui}, \text{validity period})P_{pub}$ holds and then computes the corresponding public key as $P_{Ui} = d_{Ui}P$.

For the Scenario 2, U_i $(1 \le i \le n)$ sends a registration requests to RC and obtains his/her private key as follows:

Step 1. RC chooses two numbers $x_{Ui}, y_{Ui} \in_R Z_q^*$ and computes $Y_{Ui} = y_{Ui}P$, anonymous identity $AID_{Ui} = H_0(x_i)$ for the client U_i.
Step 2. RC computes $l_{Ui} = H_0(AID_{Ui}, Y_{Ui}, \text{validity period})$ and $d_{Ui} = (y_{Ui} + xl_{Ui}) \bmod q$.
Step 3. RC sends $\langle d_{Ui}, Y_{Ui}, AID_{Ui}, \text{validity period}\rangle$ to U_i through either off-line or on-line mode and accordingly U_i'computes his/her public key as $P_{Ui} = d_{Ui}P$.

4.3 Mutual Authentication and Key Agreement Phase

In this section, we described the mutual authentication and key agreement phase of our ESL attack-free ID-MAKA scheme. This phase is identical for Scenarios 1 and 2. The difference is only the transmitted identity. For simplicity, we use the client U_i with the identity ID_{Ui} throughout this phase. Without loss of generality, assume that the client U_i wants to access the resources of the server S_j whose identity is ID_{Sj}. The description of the phase is given as follows:

Step 1. U_i chooses a number $r_{Ui} \in_R Z_q^*$ and a current timestamp T_{Ui}, then computes $R_{Ui} = r_{Ui}P$, $H_{Ui} = H_1(ID_{Ui}, ID_{Sj}, Y_{Ui}, R_{Ui}, T_{Ui}, \text{validity period})$ and $V_{Ui} = d_{Ui}H_{Ui}$. Then U_i sends the message $\langle ID_{Ui}, Y_{Ui}, R_{Ui}, T_{Ui}, V_{Ui}, \text{validity period}\rangle$ to the authentication server S_j over an open network.

Step 2. On receiving the message $\langle ID_{Ui}, Y_{Ui}, R_{Ui}, T_{Ui}, V_{Ui}, \text{validity period}\rangle$ at time T'_{Ui}, S_j checks whether ***valid period*** is overdue or not, whether $T'_{Ui} - T_{Ui} \leq \Delta T_{Ui}$ holds, and then checks whether ID_{Ui} exists in the IDRL or not. If the ***valid period*** is overdue or T_{Ui} is not fresh or $ID_{Ui} \in IDRL$, S_j rejects U_i's login request, else proceeds to the next step.

Step 3. S_j computes $H_{Ui} = H_1(ID_{Ui}, ID_{Sj}, Y_{Ui}, R_{Ui}, T_{Ui}, \text{validity period})$, $P_{Ui} = Y_{Ui} + H_0(ID_{Ui}, Y_{Ui}, \text{validity period})P_{pub}$ and verifies whether the equation $e(V_{Ui}, P) = e(H_{Ui}, P_{Ui})$ holds. If it is invalid, S_j terminates the session, otherwise S_j authenticates U_i and proceed to the next step.

Step 4. S_j selects a number $r_{Sj} \in_R Z_q^*$ and a current timestamp T_{Sj}, then computes $R_{Sj} = r_{Sj}P$, $H_{Sj} = H_1(ID_{Ui}, ID_{Sj}, Y_{Sj}, R_{Sj}, T_{Sj}, \text{validity period})$ and $V_{Sj} = d_{Sj}H_{Sj}$. S_j sends the message $\langle ID_{Sj}, Y_{Sj}, R_{Sj}, T_{Sj}, V_{Sj}\rangle$ to U_i over an open network. Now S_j computes $K_{ji} = (r_{Sj} + d_{Sj})(R_{Ui} + P_{Ui})$ and the session key as $SK_{ji} = H_2(ID_{Ui}, ID_{Sj}, R_{Ui}, R_{Sj}, T_{Ui}, T_{Sj}, K_{ji})$

Step 5. On receiving the message $\langle ID_{Sj}, Y_{Sj}, R_{Sj}, T_{Sj}, V_{Sj}\rangle$ at time T'_{Sj}, U_i checks whether $T'_{Sj} - T_{Sj} \leq \Delta T_{Sj}$ hold. If it is invalid, U_i quits the session. Otherwise, U_i goes to the next step.

Step 6. U_i computes $H_{Sj} = H_1(ID_{Ui}, ID_{Sj}, Y_{Sj}, R_{Sj}, T_{Sj}, \text{validity period})$, $P_{Sj} = Y_{Sj} + H_0(ID_{Sj}, Y_{Sj})$, and checks whether the equation $e(V_{Sj}, P) = e(H_{Sj}, P_{Sj})$ holds. If this is invalid, U_i terminates the session, otherwise authenticates the server S_j and computes the session key as $SK_{ij} = H_2(ID_{Ui}, ID_{Sj}, R_{Ui}, R_{Sj}, T_{Ui}, T_{Sj}, K_{ij})$, where $K_{ij} = (r_{Ui} + d_{Ui})(R_{Sj} + P_{Sj})$.

5 Analysis of the Proposed ESL Attack-Free ID-MAKA Scheme

5.1 Correctness

The message $\langle ID_{Ui}, Y_{Ui}, R_{Ui}, T_{Ui}, V_{Ui}, \text{validity period}\rangle$ received by S_j is correct and the client U_i correctly authenticates the server S_j provided $e(V_{Ui}, P) = e(H_{Ui}, P_{Ui})$ holds. Since we have

$$\begin{aligned} e(V_{Ui}, P) &= e(d_{Ui}H_{Ui}, P) \\ &= e(H_{Ui}, d_{Ui}P) \\ &= e(H_{Ui}, P_{Ui}) \end{aligned}$$

Accordingly the correctness of the message $\langle ID_{Sj}, Y_{Sj}, R_{Sj}, T_{Sj}, V_{Sj}\rangle$ and the authentication of the server S_j are also correctly checky by the client U_i through

$$\begin{aligned} e(V_{Sj}, P) &= e(d_{Sj} H_{Sj}, P) \\ &= e(H_{Sj}, d_{Sj} P) \\ &= e(H_{Sj}, P_{Sj}) \end{aligned}$$

Now, we claimed that both the client U_i and the authentication server S_j hold the same session key. Because we have

$$\begin{aligned} K_{ij} &= (r_{Ui} + d_{Ui})(R_{Sj} + P_{Sj}) \\ &= (r_{Ui} + d_{Ui})(r_{Sj} P + d_{Sj} P) \\ &= (r_{Ui} + d_{Ui})(r_{Sj} + d_{Sj}) P \\ &= (r_{Sj} + d_{Sj})(r_{Ui} + d_{Ui}) P \\ &= (r_{Sj} + d_{Sj})(R_{Ui} + P_{Ui}) \\ &= K_{ji} \end{aligned}$$

Thus, $SK_{ij} = SK_{ji}$ hold.

5.2 Security Analysis

Theorem 1 *In the random oracle model, our ESL attack-free ID-MAKA scheme is mutual authentication (MA)-secure i.e., it achieves server-to-client (S2C) authentication and client-to-server (C2S) authentication under the Computational Diffie–Hellman (CDH) assumption.*

Proof Assume that the adversary $\mathcal{A}$ violates S2C authentication of the proposed ESL attack-free ID-MAKA scheme, then we can develop a probabilistic polynomial time algorithm $\mathcal{C}$ that will solves an instance of CDH problem by interacting with $\mathcal{A}$. That is $\mathcal{C}$ will return syP from a given instance $\langle P, sP, yP\rangle$. To responds quickly and to avoid the data inconsistency, $\mathcal{C}$ maintains the following initial-empty lists:

- Extract list L_{Ex}^{list}: This an initial empty list and it contains the tuples of the form $\langle ID_i, Y_i, d_i\rangle$.
- H_0 list L_{H0}^{list}: This is an initial-empty list, and it contains the tuples of the form $\langle ID_i, Y_i, l_i\rangle$.
- H_1 list L_{H1}^{list}: This is an initial-empty list and it contains the tuples of the form $\langle ID_i, ID_j, Y_i, R_i, T_i$, validity period, $y_i P\rangle$.
- H_2 list L_{H2}^{list}: This is an initial-empty list and it contains the tuples of the form $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}, SK_{ij}\rangle$.

Case 1: Now we will present a challenge-response game, which is described below, played interactively by $\mathcal{C}$ and $\mathcal{A}$ to breach S2C authentication of the proposed ESL attack-free ID-MAKA scheme. In this game, we assume that q_U number of clients may be involved with q_S number of servers and the hash function H_i $(i = 0, 1, 2)$ closely behaves like true random oracle.

- **Setup:** In order to solve an instance of CDH problem, $\mathcal{C}$ picks at random $I \in \{1, 2, \ldots, q_U\}$, $J \in \{1, 2, \ldots, q_S\}$, sets $P_{pub} = sP$ and returns the system's parameter $\Omega = \langle F_q, E/F_q, e, G_q, P, P_{pub} = sP, H_0, H_1, H_2\rangle$ to the adversary $\mathcal{A}$. Now, $\mathcal{C}$ responds with $\mathcal{A}$'s as given below:

- **Hash queries to** H_0: When $\mathcal{A}$ submits a H_0 query with $\langle ID_i, Y_i\rangle$, $\mathcal{C}$ then searches L_{H0}^{list} and responds with the previous value l_i if there is a tuple $\langle ID_i, Y_i, l_i\rangle$, otherwise, choose a number $l_i \in_R Z_q^*$ such that there is no tuple of the form $\langle \cdot, \cdot, l_i\rangle$ in L_{H0}^{list}, outputs l_i as the answer and inserts the tuple $\langle ID_i, Y_i, l_i\rangle$ to L_{H0}^{list}.
- **Hash queries to** H_1: When $\mathcal{A}$ asked this query with $\langle ID_i$, ID_j, Y_i, R_i, T_i, validity period$\rangle$, $\mathcal{C}$ then responds with $y_i P$ if a tuple of the form $\langle ID_i$, ID_j, Y_i, R_i, T_i, validity period, $y_i P\rangle$ is found in L_{H1}^{list}. Otherwise, $\mathcal{C}$ selects a number $y_i \in_R Z_q^*$ and returns $y_i P$ to $\mathcal{A}$ such that there is no tuple $\langle \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, y_i P\rangle$ in L_{H1}^{list}. Now, $\mathcal{C}$ includes the tuple $\langle ID_i, ID_j, Y_i, R_i, T_i$, validity period, $y_i P\rangle$ to L_{H0}^{list}.
- **Hash queries to** H_2: When $\mathcal{A}$ asked this query with $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}\rangle$, $\mathcal{C}$ then responds with SK_{ij} if a tuple $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}, SK_{ij}\rangle$ is found in L_{H2}^{list}. Otherwise, $\mathcal{C}$ selects a number $SK_{ij} \in_R Z_q^*$ and returns it to $\mathcal{A}$ such that there is no tuple $\langle \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, SK_{ij}\rangle$ in L_{H2}^{list}. Now, $\mathcal{C}$ includes the tuple $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}, SK_{ij}\rangle$ to L_{H2}^{list}.
- **Extract**(ID_i) **queries:** When $\mathcal{C}$ received a *Extract*(ID_i) query, $\mathcal{C}$ returns d_i if a tuple $\langle ID_i, Y_i, d_i\rangle$ is found in the list L_{Ex}^{list}. Otherwise, $\mathcal{C}$ does as follows:
  - If $ID_i \in \{ID_{U_I}, ID_{S_J}\}$, $\mathcal{C}$ stops the protocol execution.
  - Else, $\mathcal{C}$ selects two numbers $l_i, a_i \in_R Z_q^*$, sets $d_i \leftarrow a_i$, $H_0(ID_i, Y_i) \leftarrow l_i$ and $Y_i \leftarrow a_i P - l_i P_{pub}$. Note that $d_i = a_i$ satisfies the equation $Y_i + H_0(ID_i, Y_i)P_{pub} = d_i P$. $\mathcal{C}$ returns d_i to $\mathcal{A}$ and adds the tuples $\langle ID_i, Y_i, d_i\rangle$ and $\langle ID_i, Y_i, l_i\rangle$ to L_{Ex}^{list} and L_{H0}^{list}, respectively.
- **Corrupt**(ID_i) **queries:** When $\mathcal{C}$ received a *Corrupt*(ID_i) query from $\mathcal{A}$, $\mathcal{C}$ does as follows:
  - $\mathcal{C}$ stops the simulation If $ID_i \in \{ID_{U_I}, ID_{S_J}\}$.
  - Else, $\mathcal{C}$ searches the lists L_{Ex}^{list} and returns the private key d_i if there is a tuple $\langle ID_i, Y_i, d_i\rangle$. Otherwise, $\mathcal{C}$ executes $H_0(ID_i)$ and *Extract*(ID_i) queries for the tuples $\langle ID_i, Y_i, l_i\rangle$ and $\langle ID_i, Y_i, d_i\rangle$, then outputs d_i as the private key. $\mathcal{C}$ adds the tuples $\langle ID_i, Y_i, d_i\rangle$ and $\langle ID_i, Y_i, l_i\rangle$ to L_{Ex}^{list} and L_{H0}^{list}, respectively.
- **Send queries:** This query can be executed interactively between $\mathcal{A}$ and $\mathcal{C}$ in the following ways:
  - When $\mathcal{A}$ makes a *Send*$(\Pi_i^t$, "$Start$"$)$, $\mathcal{C}$ responds as follows. If $ID_i \notin \{ID_{U_I}, ID_{S_J}\}$, $\mathcal{C}$ chooses a number $r_i \in_R Z_q^*$, a fresh timestamp T_i and computes $R_i \leftarrow r_i P$, $H_i \leftarrow H_1(ID_i, ID_j, Y_i, R_i, T_i$, validity period$)$, $V_i \leftarrow d_i H_i$. $\mathcal{C}$ then returns $M_1 = \langle ID_i, Y_i, R_i, T_i, V_i$, validity period, $r_i\rangle$. If $ID_i \in \{ID_{U_I}, ID_{S_J}\}$, $\mathcal{C}$ generates random numbers r_i, k_i, y_i, a timestamp T_i, then sets $R_i \leftarrow r_i P$, $H_i \leftarrow H_1(ID_i, ID_j, Y_i, R_i, T_i$, validity period$) \leftarrow y_i P$ and $V_i \leftarrow k_i H_i$ (Since $\mathcal{C}$ cannot compute correct V_i without ID_i's private key) and responds with $M_1 = \langle ID_i, Y_i, R_i, T_i, V_i$, validity period, $r_i\rangle$.
  - When $\mathcal{A}$ makes a *Send*(Π_j^s, M_1) query (here Π_j^s is a partner oracle of Π_i^t), $\mathcal{C}$ responds as follows. If $ID_i \notin \{ID_{U_I}, ID_{S_J}\}$, $\mathcal{C}$ checks the freshness of T_i and the validity of the message M_1 according to the proposed scheme. If the result is positive, $\mathcal{C}$ chooses a random number $r_j \in_R Z_q^*$, a current timestamp T_j then computes $R_j \leftarrow r_j P$, $H_j \leftarrow H_1(ID_i, ID_j, Y_j, R_j, T_j$, validity period$)$ and $V_j \leftarrow d_j H_j$, and responds with the message $M_2 = \langle ID_j, Y_j, R_j, T_j, V_j, r_j\rangle$. Then, $\mathcal{C}$ computes $K_{ji} = (r_j + d_j)(R_i + P_i)$ and the session key as $SK_{ji} = H_2(ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ji})$.

- When $\mathcal{A}$ makes a $Send(\Pi_i^t, M_2)$ query, $\mathcal{C}$ responds as follows. If $ID_i \notin \{ID_{U_I}, ID_{S_J}\}$, $\mathcal{C}$ checks the freshness of T_j and the validity of the message M_2 according to the proposed scheme. If the result is positive, $\mathcal{C}$ computes $K_{ij} = (r_i + d_i)(R_j + P_j)$ and the session key as $SK_{ij} = H_2(ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij})$.

Finally, $\mathcal{A}$ stops the protocol simulation and outputs a valid message $\langle ID_{U_I}, Y_{U_I}, R_{U_I}, T_{U_I}, V_{U_I}, \text{validity period}, r_I\rangle$ with a hash value H_{U_I} of the client U_I to login to the remote server S_J. The tuple $\langle Y_{U_I}, R_{U_I}, V_{U_I}, r_I\rangle$ can be considered as the signature of $\langle ID_{U_I}, Y_{U_I}, R_{U_I}, T_{U_I}, \text{validity period}, r_I\rangle$. According to the *forking lemma* [44], if $\mathcal{A}$ repeats the above queries by keeping the same random tape, then $\mathcal{C}$ looks into the lists $L_{Hi}^{list} (i = 0, 1, 2,)$ and outputs another valid signature $\langle Y_{U_I}, R_{U_I}, V_{U_I}^*, r_I\rangle$ with different hash value $H_{U_I}^*$ such that $H_{U_I} \neq H_{U_I}^*$ on the same message $\langle ID_{U_I}, Y_{U_I}, R_{U_I}, T_{U_I}, \text{validity period}\rangle$. Therefore we can write

$$e(V_{U_I}^*, P) = e(H_{U_I}^*, P_{U_I}) \tag{1}$$
$$e(V_{U_I}, P) = e(H_{U_I}, P_{U_I}) \tag{2}$$

From the Eqs. (1) and (2), we can derived

$$e(V_{U_I}^* - V_{U_I}, P) = e(H_{U_I}^* - H_{U_I}, P_{U_I}) \tag{3}$$

Let $H_{U_I}^* = y_1 P$, $H_{U_I} = y_2 P$, $y = (y_1 - y_2)$, $P_{U_I} = Y_{U_I} + l_{U_I} P_{pub}$ and $P_{pub} = sP$. Therefore from the Eq. (3), we have

$$\begin{aligned} e(V_{U_I}^* - V_{U_I}, P) &= e(y_1 P - y_2 P, Y_{U_I} + l_{U_I} sP) \\ &= e(yP, Y_{U_I} + l_{U_I} sP) \\ &= e(yP, Y_{U_I}) e(yP, l_{U_I} sP) \\ &= e(P, yY_{U_I}) e(l_{U_I} syP, P) \end{aligned} \tag{4}$$

From the Eq. (4), we have

$$\begin{aligned} e(l_{U_I} syP, P) &= e(V_{U_I}^* - V_{U_I}, P) e(P, yY_{U_I})^{-1} \\ &= e(V_{U_I}^* - V_{U_I}, P) e(yY_{U_I}, P) \\ &= e(V_{U_I}^* - V_{U_I} + yY_{U_I}, P) \end{aligned} \tag{5}$$

The events $ID_i = ID_{U_I}$ and $ID_j = ID_{S_J}$ occur with the probability $\frac{1}{q_U}$ and $\frac{1}{q_S}$. Therefore, $\mathcal{C}$ solves an instance of CDH problem as $syP = \frac{1}{l_{U_I}}[V_{U_I}^* - V_{U_I} + yY_{U_I}]$ with probability $\frac{\epsilon}{q_U q_S}$, which is negligible.

Case 2: Our protocol is symmetric, that is, the server S_j computes and responds similar to the client U_i. Thus, the S2C authentication can be proved in the same way as done in Case 1. □

Theorem 2 *The proposed ESL attack-free ID-MAKA scheme is AKA secure in the random oracle model under the Computational Diffie–Hellman assumption.*

Proof Assume an adversary $\mathcal{A}$ correctly output the guess bit b which is chosen in the *Test* query, then there must a polynomial time bounded algorithm $\mathcal{C}$ that can solve the the Computational Diffie–Hellman problem.

Similar to the previous theorem, $\mathcal{C}$ maintains the initial-empty lists L_{Ex}^{list} and L_{Hi}^{list} $(i = 0, 1, 2)$ to responds quickly and to avoid the data inconsistency. Let q_U, q_S, q_{Se}, and $q_{hi}(i = 0, 1, 2)$ be the total number of client, number of server, number of session and number hash queries. Suppose $\mathcal{A}$ is challenged with a CDH problem instance $(P, P_1 = aP, P_2 = bP)$ and is tasked to compute abP. $\mathcal{A}$ picks $P_0 \in G_q$, $l_1, l_2 \in \{1, 2, \ldots, q_{Se}\}$, $I \in \{1, 2, \ldots, q_U\}$, $J \in \{1, 2, \ldots, q_S\}$ at random, and gives the system's parameter $\Omega = \langle F_q, E/F_q, e, G_q, P, P_{pub} = P_0, H_0, H_1, H_2 \rangle$ to the adversary $\mathcal{A}$ and then responds to $\mathcal{A}$'s queries in the following ways:

- **Hash queries to** H_0: When $\mathcal{A}$ submits a H_0 query with $\langle ID_i, Y_i \rangle$, $\mathcal{C}$ then searches L_{H0}^{list} and responds with the previous value l_i if there is a tuple $\langle ID_i, Y_i, l_i \rangle$, otherwise, choose a number $l_i \in_R Z_q^*$ such that there is no tuple of the form $\langle \cdot, \cdot, l_i \rangle$ in L_{H0}^{list}, outputs l_i as the answer and inserts the tuple $\langle ID_i, Y_i, l_i \rangle$ to L_{H0}^{list}.
- **Hash queries to** H_1: When $\mathcal{A}$ asked this query with $\langle ID_i, ID_j, Y_i, R_i, T_i,$ validity period$\rangle$, $\mathcal{C}$ then responds with $y_i P$ if a tuple of the form $\langle ID_i, ID_j, Y_i, R_i, T_i,$ validity period, $y_i P \rangle$ is found in L_{H1}^{list}. Otherwise, $\mathcal{C}$ selects a number $y_i \in_R Z_q^*$ and returns $y_i P$ to $\mathcal{A}$ such that there is no tuple $\langle \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, y_i P \rangle$ in L_{H1}^{list}. Now, $\mathcal{C}$ includes the tuple $\langle ID_i, ID_j, Y_i, R_i, T_i,$ validity period, $y_i P \rangle$ to L_{H0}^{list}.
- **Hash queries to** H_2: When $\mathcal{A}$ asked this query with $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij} \rangle$, $\mathcal{C}$ then responds with SK_{ij} if a tuple $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}, SK_{ij} \rangle$ is found in L_{H2}^{list}. Otherwise, $\mathcal{C}$ selects a number $SK_{ij} \in_R Z_q^*$ and returns it to $\mathcal{A}$ such that there is no tuple $\langle \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, SK_{ij} \rangle$ in L_{H2}^{list}. Now, $\mathcal{C}$ includes the tuple $\langle ID_i, ID_j, R_i, R_j, T_i, T_j, K_{ij}, SK_{ij} \rangle$ to L_{H2}^{list}.
- **Extract**(ID_i) **queries:** When $\mathcal{C}$ received a *Extract*(ID_i) query, $\mathcal{C}$ returns d_i to $\mathcal{A}$ if a tuple $\langle ID_i, Y_i, d_i \rangle$ is found in L_{Ex}^{list}. Otherwise, $\mathcal{C}$ does as follows:

  - If $ID_i = ID_{U_I}$, $\mathcal{C}$ chooses $l_i \in_R Z_q^*$, sets $Y_i \leftarrow aP - l_i P_0$, $d_i \leftarrow \perp$ and then returns d_i to $\mathcal{A}$.
  - If $ID_i = ID_{S_J}$, $\mathcal{C}$ chooses $l_i \in_R Z_q^*$, computes $Y_i \leftarrow bP - l_i P_0$, $d_i \leftarrow \perp$ and then returns d_i to $\mathcal{A}$.
  - Else, $\mathcal{C}$ chooses $d_i, l_i \in_R Z_q^*$, computes $Y_i \leftarrow d_i P - l_i P_0$, and then returns $\langle ID_i, Y_i, d_i \rangle$ to $\mathcal{A}$.

  Finally, $\mathcal{C}$ inserts $\langle ID_i, Y_i, l_i \rangle$ and $\langle ID_i, Y_i, d_i \rangle$ into the lists L_{H0}^{list} and L_{Ex}^{list}.
- **Send**(Π_i^s, M) **queries:** When $\mathcal{A}$ asks a *Send* query, $\mathcal{C}$ responds in the following ways:

  - If $\Pi_i^s = \Pi_{U_I}^{l_1}$ and $M = \lambda$, $\mathcal{C}$ chooses $r_i, y_i \in_R Z_q^*$ and then sets $R_i \leftarrow r_i P + P_1$, $H_i \leftarrow H_1(ID_{U_i}, ID_{S_j}, Y_i, R_i, T_i,$ validity period$) \leftarrow y_i P$, $V_i \leftarrow y_i P_1$. $\mathcal{C}$ then outputs $M_1 = \langle ID_{U_i}, Y_i, R_i, T_i, V_i,$ validity period, $r_i \rangle$, where T_i represents the current timestamp.
  - Else, if $\Pi_i^s = \Pi_{S_J}^{l_2}$ and $M = M_1$, $\mathcal{A}$ verifies whether the equation $e(V_i, P) = e(H_i, P_i)$ holds. $\mathcal{C}$ aborts the simulation if the $e(V_i, P) \neq (H_i, P_i)$, otherwise selects $r_j, y_j \in_R Z_q^*$ and then sets $R_j \leftarrow r_j P + P_2$, $H_j \leftarrow H_1(ID_{U_i}, ID_{S_j}, Y_j, R_j, T_j) \leftarrow y_j P$, $V_j \leftarrow y_j P_2$. $\mathcal{C}$ then outputs $M_2 = \langle ID_{S_j}, Y_j, R_j, T_j, V_j, r_j \rangle$, where T_j represents the current timestamp.
  - Else, If $\Pi_i^s = \Pi_{U_I}^{l_1}$ and $M = M_2$, $\mathcal{A}$ verifies whether the equation $e(V_i, P) = e(H_i, P_i)$ holds. $\mathcal{C}$ aborts the simulation if the $e(V_i, P) \neq (H_i, P_i)$, otherwise $\mathcal{C}$ accepts the session.
  - Else, $\mathcal{C}$ answers $\mathcal{A}$'s queries according to the proposed protocol.
- **Corrupt**(ID_i) **queries:** When $\mathcal{C}$ received a *Corrupt*(ID_i) query, $\mathcal{C}$ does as follows:

  - $\mathcal{C}$ stops the simulation if $ID_i \in \{ID_{U_I}, ID_{S_J}\}$.

  - Else, $\mathcal{C}$ returns the private key d_i if there is a tuple $\langle ID_i, Y_i, d_i \rangle$ in the list L_{Ex}^{list}. Otherwise $\mathcal{C}$ executes $H_0(ID_i)$ and *Extract*(ID_i) queries for the tuples $\langle ID_i, Y_i, l_i \rangle$ and $\langle ID_i, Y_i, d_i \rangle$, then outputs d_i as the private key. $\mathcal{C}$ adds the tuples $\langle ID_i, Y_i, d_i \rangle$ and $\langle ID_i, Y_i, l_i \rangle$ to L_{Ex}^{list} and L_{H0}^{list}, respectively.

- **Reveal**(Π_i^s) **queries:** If $\Pi_i^s = \Pi_{U_I}^{l_1}$ or $\Pi_i^s = \Pi_{S_J}^{l_2}$, $\mathcal{C}$ aborts the execution. If Π_i^s is not accepted, $\mathcal{C}$ returns a *null* value, otherwise searches the list L_{H2}^{list} and returns the session key.
- **Test**(Π_i^s) **queries:** At some point, $\mathcal{C}$ will ask a *Test* query on some oracle. If $\mathcal{C}$ does not choose one of the oracles $\Pi_{U_I}^{l_1}$ to ask the *Test* query, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ chooses a bit $b \in \{0, 1\}$ and returns the correct session key if $b = 1$, otherwise outputs a random string $SK_{ij} \in_R Z_q^*$ as the session key.

In this game, $\mathcal{A}$'s simulation is perfectly indistinguishable from the proposed protocol except that the forgeries of the signatures $\langle Y_{U_I}, R_{U_I}, V_{U_I}, r_I \rangle$ and $\langle Y_{S_J}, R_{S_J}, V_{S_J}, r_J \rangle$ are not possible in the *Send* queries. As we have shown in the previous theorem that the success probabilities of the forgeries of these signatures are negligible. The adversary $\mathcal{A}$ does not aborted this game as he chooses $\Pi_{U_I}^{l_1}$ as the *Test* oracle and the oracles *Reveal*$(\Pi_{U_I}^{l_1})$ or *Reveal*$(\Pi_{S_J}^{l_2})$ have not been asked. The probability that $\mathcal{A}$ chooses *Test*$\Pi_{U_I}^{l_1}$ as *Test* oracle is $\frac{1}{q_{Se}}$. If $\mathcal{A}$ can win this game, then he must have made the corresponding H_2 hash query of the form $\langle ID_{U_I}, ID_{S_J}, R_{U_I}, R_{S_J}, T_{U_I}, T_{S_J}, K_{IJ} \rangle$ if $\Pi_{U_I}^{l_1}$ is the initiator oracle. Thus, $\mathcal{A}$ can find the corresponding item in the list L_{H2}^{list} and outputs $abP = K_{IJ} - r_I r_J P - r_I P_2 - r_J P_1$ as the solution to the CDH problem.
Thus, the advantage of $\mathcal{C}$ solving the CDH problem is such that

$$Adv_{\mathcal{C}}^{CDH}(k) \geq \frac{1}{q_{Se}} \times \frac{1}{q_U} \times \frac{1}{q_S} \times \frac{1}{q_{H2}} \times Adv_{\mathcal{A}}^{ID-MAKA}(k)$$

Here $Adv_{\mathcal{C}}^{CDH}(k)$ denotes the success probability of breaching the CDH problem by the challenge $\mathcal{C}$. □

6 Performance Comparison of the Proposed Scheme with Others

Similar to the schemes [23,24], the proposed scheme is also designed for both the general and dynamic clients. Thus, we evaluate the security and computation cost efficiency of the proposed scheme with them. The multi-server authentication schemes proposed in the literature are vulnerable to ESL attack since they computed their session key using only ephemeral secrets chosen by client and server. Thus, the leakage of these ephemeral secrets leads to the compromise of the session key. Compared to others, the schemes [23,24] are most efficient in terms of computation and security, however we argued that they are also vulnerable to ESL attack. In these schemes, if ephemeral secrets are leaked to an outsider then he/she can compute the session key easily. Now, we illustrates the ESL attack on these scheme as follows:

- In [23], client U_i and server S_j compute the following:
  - U_i chooses a number $a_i \in_R Z_q^*$ and computes $q_j = H(ID_{Sj}), t_i = g^{a_i}, h_i = H_1(t_i)$, $v_i = H_1(h_i)$, $Q_j = P_{pub} + q_j P$, $X_i = a_i Q_j$ and $Y_i = (a_i + h_i) DID_{U_i}$. Then U_i sends $\langle ID_{U_i}, X_i, Y_i, v_i$, validity period$\rangle$ S_j.

  - S_j selects a number $b_j \in_R Z_q^*$ and computes $X_j = b_j Q_j$, $t_i = e(X_i, DID_{S_j})$ and $z_j = H_1(t_i \parallel X_i \parallel X_j \parallel Y_i)$. Now S_j sends $\langle z_j, X_j \rangle$ to U_j.
  - U_i and S_j compute the session key as $SK_{ij} = SK_{ji} = H_1(t_i \parallel K \parallel X_i \parallel Y_j \parallel z_j)$, where $K = a_i X_j = b_j X_i = a_i b_j Q_j$.

  Therefore, if a_i and b_j are leaked, then the outsider easily figure out the session key SK_{ij}. Furthermore, U_i's secret key can be computed as $DID_{U_i} = (a_i + h_i)^{-1} Y_i$.
- In [24], client U_i and server S_j compute the following:
  - U_i chooses a number $a_{U_i} \in_R Z_q^*$, computes $X_{U_i} = a_{U_i} P$, $b_{U_i} = H_3(ID_{U_i} \parallel ID_{S_j} \parallel R_{U_i} \parallel X_{U_i} \parallel T_{U_i} \parallel \text{valid period})$ and $s_{U_i} = (a_{U_i} + b_{U_i})^{-1} x_{U_i}$, where T_{U_i} is the current timestamp. Then U_i send $\langle ID_{U_i}, R_{U_i}, X_{U_i}, s_{U_i}, T_{U_i}, \text{valid period} \rangle$ to S_j.
  - S_j chooses a number $b_{S_j} \in_R Z_q^*$, computes $X_{S_j} = a_{S_j} P$, $b_{S_j} = H_4(ID_{U_i} \parallel ID_{S_j} \parallel R_{U_i} \parallel X_{U_i} \parallel \text{valid period} \parallel R_{S_j} \parallel X_{S_j} \parallel T_{S_j})$ and $s_{S_j} = (a_{S_j} + b_{S_j})^{-1} x_{S_j}$, where T_{S_j} is the current timestamp. Then S_j send $\langle ID_{S_j}, R_{S_j}, X_{S_j}, s_{S_j}, T_{S_j} \rangle$ to U_i.
  - Now U_i and S_j compute the session key as $SK_{ij} = SK_{ji} = H_5(ID_{U_i} \parallel ID_{S_j} \parallel X_{U_i} \parallel X_{S_j} \parallel K)$ where $K = a_{U_i} X_{S_j} = a_{S_j} X_{U_i} = a_{U_i} a_{S_j} P$.

  It is to be noted that if a_{U_i} and a_{S_j} are compromised, then the session key SK_{ij} will be compromised to the outsider. Furthermore, the secret keys of U_i and S_j can be computed as $x_{U_i} = s_{U_i}(a_{U_i} + b_{U_i})$ and $x_{S_j} = s_{S_j}(a_{S_j} + b_{S_j})$.

For the performance comparison with respect to computation cost, we define following notations [37]:

- t_m: Time required for modular multiplication.
- t_e: Time required for modular exponentiation operation, $t_e \approx 240 t_m$.
- t_s: Time required for elliptic curve scalar point multiplication operation, $t_s \approx 29 t_m$.
- t_p: Time required for bilinear pairing operation, $t_p \approx 87 t_m$.
- t_i: Time required for modular inversion operation, $t_i \approx 11.6 t_m$.

It is to be noted that the mutual authentication with session key agreement phase is the dominating process in terms of computation cost than setup and extract phases as they are executed only once. Thus, we consider only the mutual authentication with session key agreement phase and accordingly compare the proposed scheme with [23,24]. We demonstrated

Table 1 Computation cost comparison of the proposed scheme with others

Scheme/attributes	Scheme [23]	Scheme [24]	Ours
F_1	$t_e + 4t_s$	$5t_s + t_i$	$2t_p + 4t_s$
F_2	$2t_p + t_e + 3t_s$	$5t_s + t_i$	$2t_p + 4t_s$
F_3	$757t_m$	$312t_m$	$600t_m$
F_4	Yes	Yes	Yes
F_5	No	No	Yes
F_6	Yes	Yes	Yes
F_7	2	2	2

F_1: Client's computation cost for mutual authentication and session key agreement phase; F_2: Servers's computation cost for mutual authentication and session key agreement phase; F_3: Over all computation cost for mutual authentication and session key agreement phase; F_4: Provides provable security in the random oracle model; F_5: Provides resilience against ESL attack; F_6: Adaptability for general and dynamic clients; F_7 : Number of rounds the comparative result in Table 1. The proposed scheme bears lower computation cost than [23] and slightly increases the same over [24]. However, only the proposed scheme provides resilience against ESL attack whereas others do not.

7 Conclusion

In this paper, we proposed an ID-MAKA scheme for mobile multi-server environment. The proposed scheme is suitable for both the general and dynamic mobile clients. The earlier schemes do not resist the ESL attack whereas our scheme fulfills this objective. The security analysis of the proposed scheme is done in the random oracle model and proven to be provably secure under the Computational Diffie–Hellman assumption.

References

1. He, D. (2012). Cryptanalysis of an authenticated key agreement protocol for wireless mobile communications. *ETRI Journal*, *34*(3), 482–484.
2. He, D. (2012). An efficient remote user authentication and key exchange protocol for mobile client-server environment from pairings. *Ad Hoc Networks*, *10*(6), 1009–1016.
3. He, D., Chen, J., & Hu, J. (2012). An ID-based client authentication with key agreement protocol for mobile client-server environment on ECC with provable security. *Information Fusion*, *13*(3), 223–230.
4. He, D., Chen, J., & Zhang, R. (2012). A more secure authentication scheme for telecare medicine information systems. *Journal of Medical Systems*, *36*(3), 1989–1995.
5. He, D., Chen, J., & Hu, J. (2011). Further improvement of Juang et al. 's password-authenticated key agreement scheme using smart cards. *Kuwait Journal of Science & Engineering*, *38*(2A), 55–68.
6. He, D., Chen, J., & Hu, J. (2012). Improvement on a smart card based password authentication scheme. *Journal of Internet Technology*, *13*(3), 405–410.
7. He, D., Chen, J., & Chen, Y. (2012). A secure mutual authentication scheme for session initiation protocol using elliptic curve cryptography. *Security and Communication Networks*. doi:10.1002/sec.506.
8. He, D., Chen, Y., & Chen, J. (2012). Cryptanalysis and improvement of an extended chaotic maps-based key agreement protocol. *Nonlinear Dynamics*, *69*(3), 1149–1157.
9. Tsai, J. L. (2008). Efficient multi-server authentication scheme based on one-way hash function without verification table. *Computers and Security*, *27*(3–4), 115–121.
10. Geng, J., & Zhang, L. (2008). A dynamic ID-based user authentication and key agreement scheme for multi-server environment using bilinear pairings. In *Proceedings of the power electronics and intelligent transportation system*, pp. 33–37.
11. Liao, Y. P., & Wang, S. S. (2009). A secure dynamic ID based remote user authentication scheme for multi-server environment. *Computer Standards and Interfaces*, *31*(1), 24–29.
12. Hsiang, H. C., & Shih, W. K. (2009). Improvement of the secure dynamic ID based remote user authentication scheme for multi-server environment. *Computer Standards and Interfaces*, *31*(6), 1118–1123.
13. Lee, C.-C., Lin, T.-H., & Chang, R.-X. (2011). A secure dynamic ID based remote user authentication scheme for multi-server environment using smart cards. *Expert Systems with Applications*, *38*, 13863–13870.
14. Lee, S. G. (2009). Cryptanalysis of multiple-server password-authenticated key agreement scheme using smart cards. Cryptology ePrint Archive 2009; Report 2009/490.
15. Chuang, Y. H., & Tseng, Y. M. (2009). Security weaknesses of two dynamic ID-based user authentication and key agreement schemes for multi-server environment. In *Proceedings of the national computer symposium (NCS2009), vol. 5*, pp. 250–257.
16. Sood, S. K., Sarje, A., & Singh, K. (2011). A secure dynamic identity based authentication protocol for multi-server architecture. *Journal of Network and Computer Applications*, *34*, 609–618.
17. Li, X., Xiong, Y., Ma, J., & Wang, W. (2012). An efficient and secure dynamic identity based authentication protocol for multi-server architecture using smartcards. *Journal of Network and Computer Applications*, *35*, 763–769.
18. Han, W. (2012). Weaknesses of a dynamic identity based authentication protocol for multi-server architecture. arXiv preprint archive 2012. http://arxiv.org/ftp/arxiv/papers/1201/1201.0883.pdf.

19. Li, X., Ma, J., Wang, W., Xiong, Y., & Zhang, J. (2013). A novel smart card and dynamic ID based remote user authentication scheme for multi-server environments. *Mathematical and Computer Modelling*, *58*, 85–95.
20. Zhaoa, D., Peng, H., Li, S., & Yang, Y. (2013). An efficient dynamic ID based remote user authentication scheme using self-certified public keys for multi-server environment. arXiv preprint archive 2013. http://arxiv.org/pdf/1305.6350.pdf.
21. Wang, B., & Ma, M. (2013). A smart card based efficient and secured multi-server authentication scheme. *Wireless Personal Communications*, *68*, 361–378.
22. He, D., & Wu, S. (2013). Security flaws in a smart card based authentication scheme for multi-server environment. *Wireless Personal Communications*, *70*, 323–329.
23. Chuang, Y.-H., & Tseng, Y.-M. (2013). Towards generalized ID-based user authentication for mobile multi-server environment. *Intrnational Journal of Communication Systems*, *25*, 447–406.
24. Han, W., & Zhu, Z. (2013). An ID-based mutual authentication with key agreement protocol for multi-server environment on elliptic curve cryptosystem. *Intrnational Journal of Communication Systems*. doi:10.1002/dac.2405.
25. Cao, X., & Zhong, S. (2006). Breaking a remote user authentication scheme for multi-server architecture. *IEEE Communications Letters*, *10*(8), 580–581.
26. Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In *Proceedings of the advances in cryptology (CRYPTO'84), LNCS 196*, Springer, Berlin, pp. 47–53.
27. Boneh, D., & Franklin, M. (2003). Identity-based encryption from the Weil pairing. *SIAM Journal on Computing*, *32*, 586–615.
28. Tseng, Y.-M., Tsai, T.-T., & Huang, S.-S. (2013). Leakage-free ID-based signature. *The Computer Journal*. doi:10.1093/comjnl/bxt116.
29. Canetti, R., & Krawczyk, H. (2001). Analysis of key exchange protocols and their use for building secure channels. In *Proceedings of advances in cryptology (Eurocrypt'01), LNCS*, pp. 453–474.
30. Cheng, Z., Nistazakis, M., Comley, R., & Vasiu, L. (2005). On the indistinguishability-based security model of key agreement protocols-simple cases, Cryptology ePrint Archieve, Report 2005/129.
31. Mandt, T., & Tan, C. (2008). Certificateless authenticated two-party key agreement protocols. In *Proceedings of the ASIAN'08, LNCS 4435*, pp. 37–44.
32. Islam, S. H., & Biswas, G. P. (2011). Comments on ID-based client authentication with key agreement protocol on ECC for mobile client-server environment. In *Proceedings of the international conference on advanced in computing and communications (ACC 2011), CCIS 191*, Springer, Berlin, pp. 628–635.
33. Islam, S. H., & Bisws, G. P. (2011). A more efficient and secure ID-based remote mutual authentication with key agreement scheme for mobile devices on elliptic curve cryptosystem. *Journal of Systems and Software*, *84*, 1892–1898.
34. LaMacchia, B., Lauter, K., & Mityagin, A. (2007). Stronger security of authenticated key exchange. In *Proceeding of the ProvSec'07*, pp. 1–16.
35. Swanson, C. M. (2008). Security in key agreement: Two-party certificateless schemes. Master's thesis, University of Waterloo, Canada.
36. Islam, S. H., & Biswas, G. P. (2013). Design of improved password authentication and update scheme based on elliptic curve cryptography. *Mathematical and Computer Modelling*, *57*, 2703–2717.
37. Islam, S. H., & Biswas, G. P. (2012). A pairing-free identity-based authenticated group key agreement protocol for imbalanced mobile networks. *Annals of Telecommunications*, *67*(11–12), 547–558.
38. Hou, M., Xu, Q., Shanqing, G., & Jiang, H. (2010). Cryptanalysis of identity-based authenticated key agreement protocols from parings. *Journal of Networks*, *5*(7), 826–855.
39. Ballare, M., & Rogaway, P. (1993). Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on computer and communications security (CCS'93)*, pp. 62–73.
40. Pippal, R. S., Jaidhar, C. D., & Tapaswi, S. (2013). Robust smart card authentication scheme for multi-server architecture. *Wireless Personal Communications*, *72*, 729–745.
41. Tsai, J.-L., Lo, N.-W., & Wu, T.-C. (2013). A new password-based multi-server authentication scheme robust to password guessing attacks. *Wireless Personal Communications*, *71*, 1977–1988.
42. Miller, V. S. (1985). Use of elliptic curves in cryptography. In *Proceeding of the advances in cryptology (Crypto'85)*, pp. 417–426.
43. Koblitz, N. (1987). Elliptic curve cryptosystem. *Journal of Mathematics of Computation*, *48*(177), 203–209.
44. Pointcheval, D., & Stern, J. (2000). Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, *13*, 361–396.

SK Hafizul Islam has received his Ph.D in Computer Science and Engineering from Indian School of Mines (ISM), Dhanbad, India, under the INSPIRE Fellowship Ph.D Program, funded by DST, Govt. of India and Information Security Education and Awareness (ISEA) program, funded by Department of Information Technology, Ministry of Communication and Information Technology, Govt. of India. He received his B.Sc (Hons.) in Mathematics and M.Sc in Applied Mathematics from Vidyasagar University, West Bengal, India in 2004 and 2006, respectively. He also did M.Tech in Computer Application from ISM Dhanbad in 2009. Presently, he has been working as an Assistant Professor in the Department of Computer Science & Information Systems, BITS Pilani, Rajasthan 333031, India. He served as reviewer in many reputed International Journals and Conferences. He has published thirty research papers in Journals and Conference Proceedings of International reputes. His research interest includes Cryptography and Information Security.