Wireless Pers Commun (2015) 82:2727–2750
DOI 10.1007/s11277-015-2375-5

Design of Two-Party Authenticated Key Agreement Protocol Based on ECC and Self-Certified Public Keys

SK Hafizul Islam · G. P. Biswas

Published online: 10 February 2015

Abstract A two-party authenticated key agreement (2PAKA) protocol based on Elliptic curve cryptography (ECC) and the self-certified public key (SC-PKC) of the user is proposed in this paper. Although several ECC-based 2PAKA protocols using either public key infrastructure (PKI) or Identity-based cryptosystem (IBC) have been proposed recently, they suffer from certain limitations. For instance, the former requires heavy computation and management of public key certificate (PKC) and the latter induces a private key escrow problem as the private key is generated by a trusted third party, called private key generator (PKG). Also the man-in-the-middle attack may occur from a malicious PKG and the resilience against such an attack for an authenticated key agreement protocol is needed. In this paper, we proposed the design of a 2PAKA protocol using ECC and SC-PKC that removes all the limitations as mentioned above. In SC-PKC, a trusted third party, called system authority (SA) generates the public key of a user based on user identity signed by SA and user generated signature based on the private key of the user. The proposed scheme is provably secure in the random oracle model under the Computational Diffie–Hellman assumption. Also the formal security validation of our scheme using Automated Validation of Internet Security Protocols and Applications software is done and simulation results prove that it is safe against both the active and passive adversaries. In addition, our protocol is computationally efficient and may be considered as an alternative of the PKI- or IBC-based 2PAKA protocol.

Keywords Authenticated key agreement · Elliptic curve · Random oracle model · Self-certified public key · AVISPA tool

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

G. P. Biswas
Department of Computer Science and Engineering, Indian School of Mines, Dhanbad 826004, Jharkhand, India
e-mail: gpbiswas@gmail.com

1 Introduction

A 2PAKA protocol establishes an authenticated and shared secret session key between two entities through an open network. The Diffie–Hellman protocol [1] is the first such an approach that uses the exchange of two messages and generates a secret session key between two entities using very simple operations. However, this technique, due to lack of authentication of the entities, is vulnerable to the man-in-the-middle attack (MIMA). Later on, several 2PAKA protocols to avoid MIMA have been proposed in the literature and some of their recently proposed schemes are now described. The 2PAKA protocols proposed in [10,23,26] used the PKI to circumvent the MIMA, however, due to need of execution of the time-consuming modular exponentiation operation, they are not suitable for resource constrained environments such as sensor networks, mobile devices, smart cards, low-end PDAs, where the computation capability and battery life time are limited.

The 2PAKA protocols using ECC [17] have been proposed in [27,35,36]. These are more efficient than PKI-based 2PAKA protocols, because instead of modular exponentiation, elliptic curve point addition/multiplications are used in ECC-based 2PAKA schemes. Also ECC-based 2PAKA protocol is more secure as its security lies on the difficulty of solving the elliptic curve discrete logarithm problem (ECDLP), which is more difficult than the discrete logarithm problem (DLP) used in PKI-based 2PAKA protocols. Since ECC-based scheme requires comparatively less computation and storage space, for instance, a 160-bit ECC key provides the same level of security with a 1,024-bit RSA key; it is widely accepted for designing different cryptographic techniques. The ECC-based 2PAKA protocols however have certain disadvantages. It needs a certificate authority (CA) to validate public key certificates and thus requires additional processing and extra storage space to maintain and store the public keys and certificates.

The IBC proposed by Shamir [31] and subsequently its full functional solution given by Boneh and Franklin [5] seems to be efficient for many reasons. One of the advantages of IBC is that the requirement of the public key certificate can be avoided completely, because the public key of an entity is computed from an identity of the entity such as email address, physical IP address, etc., rather than using a random number and the master secret key of a trusted authority, called PKG. Recently, several IBC- and ECC-based 2PAKA protocols have been proposed in [18,19,30,36,37] and [6,9,11,14,22,24,29,32–34,38,40] respectively. However, IBC-based 2PAKA protocols suffer from the private key escrow problem as the private key is generated by PKG and the MIMA may occur from a malicious PKG.

In 1991, SC-PKC proposed by Girault [16] appears to be better than the PKI- or IBC, where a trusted third party, called system authority (SA) generates the public key from the signature of the entity's private key with his identity signed by SA and the private key is computed by the entity only. The advantage of SC-PKC is that the authenticity of an entity's public key can be verified publicly without using any certificate issued by the SA and the private key known to the entity only, so the private key escrow problem is resolved. As compared with conventional public key cryptosystems (e.g., PKI or IBC), SC-PKC requires lower communication overheads, storage space, and computation efforts since no certificate is required.

1.1 Related Works

In 2002, Hsieh et al. [19] proposed an enhancement of Saeednia's protocol [30], where the reduction of computational cost and improvement of security aspects was made. However, Tseng et al. [37] demonstrated that Hsieh's scheme cannot resist key-compromise imperson-

ation (K-CI) attack and thus proposed an improved scheme to protect the same. Subsequently, Hölbl and Welzer [18] proposed an improvement of Hsieh's scheme [19] and Tseng's scheme [36] with much reduction in computation and communication cost. Later on, Zhang et al. [42] pointed out that the improved protocol is vulnerable to the impersonation attack (IA). Although Smart [33] proposed an identity-based key agreement protocol using Boneh and Franklin's IBE scheme [5], Chen and Kudla [9], and Shim [32] independently proved that the perfect forward secrecy (PFS) cannot be achieved in the Smart's protocol. Furthermore, Shim proposed an efficient identity-based key agreement protocol to remove the security pitfalls of Smart's protocol but, Sun and Hsieh [34] demonstrated that Shim's protocol [32] is susceptible to MIMA. An efficient identity-based key agreement protocol using pairings was proposed by Ryu et al. [29] with minimum computation and communication costs. Also Ryu et al.'s protocol achieves better efficiency as it reduces the on-line pairing computation cost to zero, Boyd and Choo [6] showed that Ryu et al.'s protocol does not satisfy K-CI resilience property. Recently, Wang et al. [38] identified a weakness in Ryu et al.'s protocol, called reflection attack (RA) and proposed an improved scheme to remove the weakness found. An identity-based key agreement protocol was proposed by McCullagh and Barreto [24], which has been analyzed by Xie [40], and Choo et al. [14] and pointed out that the protocol does not have the K-CI resilience property. Although they proposed an improvement of McCullagh and Barreto's protocol, it still contains the K-CI attack as mentioned by Li et al. [22].

1.2 Our Contributions

As stated, the 2PAKA protocol based on PKI or IBC suffers from public key management and inherent private key escrow problem, and as a remedy of the same, we proposed an efficient and provably secure pairing-free 2PAKA protocol using ECC and SC-PKC. Our scheme, which is described in Sect. 3, has the following characteristics:

1. *Avoidance of public key certificate* Since PKI-based protocols have overheads to manage a public key certificate for certificate generation, verification, revocation, storage, delivery, etc., the self-certified public keys, which are implicitly verified for authentication, are included in our proposed scheme to eliminate the burden of verifying the public key before using it.
2. *Avoidance of key escrow problem* The security of IBC relies on PKG and since PKG knows the private key of each entity, MIMA from a malicious PKG is possible and considered as one of the most powerful attacks against a key agreement protocol. The proposed protocol, on the other hand, uses a third party, called SA to generate the entity's public key, the corresponding private key of which is known to the entity only, and thus the our protocol is free from private key escrow problem, which is inherent in all IBCs. Also the proposed protocol is free from the key escrow problem, because a malicious SA, due to the unknown of the users' private key, cannot compute a session key agreed between two users provided the entities' public keys are not replaced by the SA.
3. *Avoidance of bilinear pairings and map-to-point hash function* The proposed protocol does not require bilinear pairings and map-to-point (MTP) function, which is a special hash function that converts an identity to an underlying elliptic curve point. Since both are time consuming operations, they are not used in resource constrained (i.e. power, storage space) environments such as mobile devices, smart cards, low-end PDAs, etc. As an estimation of cost, one bilinear pairing operation requires two to three times more multiplication than an elliptic curve point multiplication in the same field and an MTP hash function is implemented as a probabilistic algorithm and is more expensive than an elliptic curve point multiplication [2].

4. *Formal security analysis and validation* We analyzed our scheme in the random oracle model and showed that the security of the session key depends on the Computational Diffie-Hellman (CDH) assumption. Also, it is found that the formal security analysis of many cryptographic protocols [44–49] has been done based on AVISPA tool [50,51], which is a well-known internet protocol security validation software developed for the detection of active and passive attacks. Accordingly, the proposed 2PAKA protocol is implemented in AVISPA tool and the simulation results illustrate that the scheme has no active attack and passive attack as well.
5. *Computation and communication efficiencies* Since the proposed scheme uses the SC-PKC and ECC, it is efficient in terms of both security and overheads. Because the security of the proposed scheme is based on the CDHP assumption and the proposed protocol satisfies all the security properties described in Blake-Wilson [4]. In addition, our protocol is efficient in terms of number of rounds, communication and computation costs, the detail of which is given in Sects. 5 and 6.

1.3 Organization of the Paper

The rest of the paper is organized as follows. The necessary technical backgrounds are given in Sect. 2 and the Sect. 3 describes the details of the proposed 2PAKA protocol. The attack model and the provable security analysis of the proposed scheme are given in Sect. 4. In Sect. 5, formal security analysis based on AVISPA tool is discussed. The informal security analysis against different attacks and the comparison of our proposed protocol with others are addressed in Sect. 6. The estimation and the computation efficiency of the proposed protocol over relevant protocols are given in Sect. 7 and finally, Sect. 8 concludes the paper.

2 Preliminaries

2.1 Elliptic Curve Cryptography

The ECC was initially proposed by Miller [25] and Koblitz [20], and its security was based upon the difficulty of ECDLP. Later on, it is widely accepted in designing different cryptographic protocols for its effectiveness in security, communication and computation and a number of efficient ECC-based PKCs have been proposed. For the sake of clarity, the basics of the elliptic curve cryptography and some related computationally hard problems are given below.

Let E/F_q be a set of elliptic curve points over the prime field F_q, defined by the following non-singular elliptic curve equation:

$$y^2 \bmod q = (x^3 + ax + b) \bmod q \tag{1}$$

where $x, y, a, b \in F_q$ and $(4a^3 + 27b^2) \bmod q \neq 0$. The additive elliptic curve group defined as $G_q = \{(x, y) : x, y \in F_q \text{ and } (x, y) \in E/F_q\} \cup \{O\}$, where the point *"O"* is known as *"point at infinity"* or *"zero point"*. A brief discussion about the elliptic curve group properties [17] is given below:

- *Point addition* Let P, Q are two points on the curve (1), then $P + Q = R$, where the line joining P and Q intersects the curve (1) at $-R$, and the reflection of it with respect to the *x-axis* is the point R.
- *Point subtraction* If $Q = -P$, then $P + Q = P - P = O$, the line joining P and $-P$ intersects the curve (1) at O.

- *Point doubling* Point doubling is the addition of a point P on the curve (1) to itself to obtain another point Q on (1). Let $2P = Q$, the tangent line at P intersects the curve (1) at $-Q$; reflection of it with respect to the *x-axis* is the point Q.
- *Scalar point multiplication* The scalar point multiplication on the cyclic group G_q is defined as $kP = P + P + \cdots + P$ (k times), where $k \in Z_q^*$ is a scalar.
- *Order of a point* A point P has order n if n is the smallest integer such that $nP = O$ and $n > 0$.

2.2 Computational Problem

Definition 1 (*Elliptic curve discrete logarithm problem (ECDLP)*) Given $P, Q \in G_q$, where $Q = aP$ and $a \in Z_q^*$. It is hard to compute a from Q.

Definition 2 (*Computational Diffie–Hellman (CDH) problem*) Given $(P, aP, bP) \in G_q$ for any $a, b \in Z_q^*$, computation of abP is hard to the group G_q.

Definition 3 (*Elliptic curve discrete logarithm problem (ECDLP) assumption*) The probability that a polynomial time-bounded algorithm $\mathcal{A}$ can solve the ECDLP is defined as $Adv_{\mathcal{A},G_q}^{ECDLP} = Pr[\mathcal{A}(P, aP) = a : P \in G_q; a \in Z_q^*]$. For any probabilistic polynomial time-bounded algorithm $\mathcal{A}$, $Adv_{\mathcal{A},G_q}^{ECDLP}$ is negligible.

Definition 4 (*Computational Diffie–Hellman (CDH) assumption*) The probability that a polynomial time-bounded algorithm $\mathcal{A}$ can solve the CDH problem is defined as $Adv_{\mathcal{A},G_q}^{CDH} = Pr[\mathcal{A}(P, aP, bP) = abP : P \in G_q; a, b \in Z_q^*]$. For any probabilistic polynomial time-bounded algorithm $\mathcal{A}$, $Adv_{\mathcal{A},G_q}^{CDH}$ is negligible.

3 Proposed 2PAKA Protocol

The proposed 2PAKA protocol consists of three phases—setup phase, user registration phase and key agreement phase, the details of them are described below.

3.1 Setup Phase

In this phase, SA generates the system's parameter Ω. For this, it selects a security parameter $k \in Z^+$ and an elliptic curve group G_q (Eq. 1), defined over the finite field F_q of prime order q (k-bit length), where P is a base point, is a large prime number. The SA selects a $s \in Z_q^*$ as his private key and computes the corresponding public key as $P_S = sP$. It also chooses three secure one-way hash functions $H_0, H_1, H_2 : \{0,1\}^* \rightarrow \{0,1\}^k$ and finally publishes the system's parameter $\Omega = \{F_q, E/F_q, G_q, P, P_S, H_0, H_1, H_2\}$.

3.2 User Registration phase

When a user i with identity ID_i is going to join the system, he selects a number $x_i \in_R Z_q^*$ and then computes

$$X_i = H_0(ID_i \parallel x_i)P \tag{2}$$

operation. Now the user ID_i sends (ID_i, X_i) to SA through a secure channel. Then SA chooses a $t_i \in_R Z_q^*$ for ID_i and computes

$$P_i = H_0(ID_i \parallel t_i)P_S + X_i \tag{3}$$

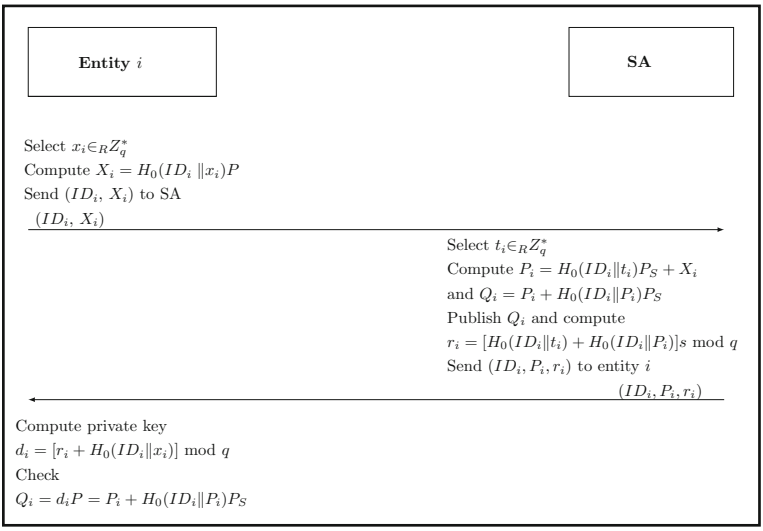

Fig. 1 Registration phase of the proposed protocol

$$r_i = [H_0(ID_i \parallel t_i) + H_0(ID_i \parallel P_i)]s \bmod q \tag{4}$$

Now SA sends (ID_i, P_i, r_i) to ID_i through a secure channel. Upon receiving (ID_i, P_i, r_i) from SA, user ID_i computes his private key

$$d_i = [r_i + H_0(ID_i \parallel x_i)] \bmod q \tag{5}$$

and verifies the validity of (ID_i, P_i, r_i) by checking that

$$d_i P = P_i + H_0(ID_i \parallel P_i)P_S \tag{6}$$

If the Eq. (6) holds, then ID_i accepts d_i as his private key and computes $Q_i = P_i + H_0(ID_i \parallel P_i)P_S$ as his public key. After registration, SA publishes the public key Q_i of ID_i. It is worth to note that, SA needs not issue any extra certificate with respect to Q_i. The verification of the equation (6) is as follows

$$\begin{aligned}
Q_i &= d_i P \\
&= (r_i + H_0(ID_i \parallel x_i))P && [Eq.\ (5)] \\
&= [H_0(ID_i \parallel t_i) + H_0(ID_i \parallel P_i)]sP + H_0(ID_i \parallel x_i)P && [Eq.\ (4)] \\
&= H_0(ID_i \parallel t_i)sP + H_0(ID_i \parallel P_i)sP + H_0(ID_i \parallel x_i)P \\
&= H_0(ID_i \parallel t_i)P_S + H_0(ID_i \parallel P_i)P_S + X_i && [Eq.\ (2)] \\
&= H_0(ID_i \parallel t_i)P_S + X_i + H_0(ID_i \parallel P_i)P_S \\
&= P_i + H_0(ID_i \parallel P_i)P_S && [Eq.\ (3)]
\end{aligned}$$

The details of the registration phase is given in Fig. 1.

3.3 Key Agreement Phase

Assume that two entities A and B want to establish a secret session key between them. We assume that the entity A acts as an initiator and the entity B is a responder. Now the following two rounds are executed to establish a secret session key between them.

Step 1. Entity A performs the followings:

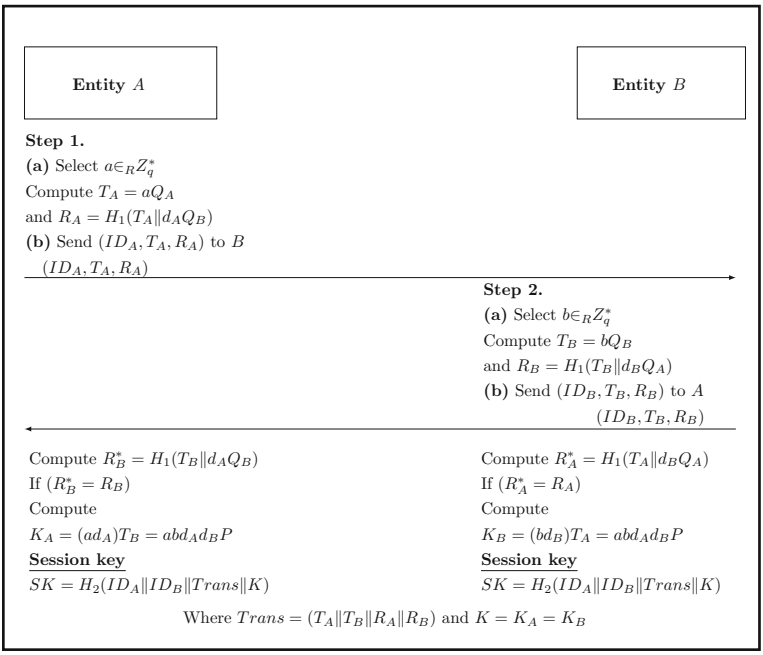

Fig. 2 Key agreement phase of the proposed protocol

(a) Select $a \in_R Z_q^*$, compute $T_A = aQ_A$ and $R_A = H_1(T_A \parallel d_A Q_B)$.
(b) Send (ID_A, T_A, R_A) to B.

Step 2. On receiving (ID_A, T_A, R_A) from A, B will:

(a) Choose $b \in_R Z_q^*$, compute $T_B = bQ_B$ and $R_B = H_1(T_B \parallel d_B Q_A)$.
(b) Send (ID_B, T_B, R_B) to A.

Now entity A computes $R_B^* = H_1(T_B \parallel d_A Q_B)$ and compares with received R_B, and if it is hold, i.e., if $R_B^* = R_B$, then A computes the partial session key as

$$\begin{aligned} K_A &= ad_A T_B \\ &= ad_A bd_B P \\ &= abd_A d_B P \end{aligned}$$

Similarly, entity B computes $R_A^* = H_1(T_A \parallel d_B Q_A)$ and compares with received R_A. If $R_A^* = R_A$, then B computes the partial session key as

$$\begin{aligned} K_B &= bd_B T_A \\ &= bd_B ad_A P \\ &= abd_A d_B P \end{aligned}$$

After successful completion of the above processes, entities A and B generate a common session key $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$, where $K = K_A = K_B$ and $Trans = (T_A \parallel T_B \parallel R_A \parallel R_B)$. Detail of the key agreement phase is illustrated in Fig. 2.

4 Formal Security Analysis of the Proposed Protocol

4.1 Attack Model

In this attack model, following assumptions have been made:

- Let $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ be a set of n users and the protocol can be executed by any two distinct users. We assume that each user U_i $(1 \leq i \leq n)$ computes their pair of long-term private/public keys (d_i, Q_i) before the *Key agreement phase*. We denote ID_i as the identity of the user U_i.
- The protocol Π is modelled as a collection of n programs running at n users. Each instance of Π within a user is defined as a session and each user may have multiple such sessions running concurrently.
- The communication channel is assumed to be fully controlled by a benign adversary $\mathcal{A}$, who can insert, delete or modify the protocol messages. It can also start multiple new instances of any of the users, modelling the parties engaging in many sessions simultaneously.

Let $\Pi_{i,j}^s$ the s-th instance of the party $U_i \in \mathcal{U}$ involved with the partner party $U_j \in \mathcal{U}$ in a session. The session identifier $sid_{i,j}^s$ is assumed to be the concatenation of the messages exchanged between the two users along with their identities. The partner identifier $pid_{i,j}^s$ of an oracle $\Pi_{i,j}^s$ is a set containing the identity ID_i of U_i and the identity of the party with whom U_i wants to establish a session. We define the security of a 2PAKA protocol by a series of games between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$ in which $\mathcal{A}$ must solve a challenge on a *Test* session. In this game, $\mathcal{A}$ is allowed to select the identities of all honest participants and issue the following polynomial number of queries:

- *Registration*(ID_i) With this query, $\mathcal{A}$ registers a public key Q_i on behalf of any user ID_i of his choice.
- *Execute*$(\Pi_{i,j}^s)$ With this query, $\mathcal{A}$ obtains the protocol messages that were exchanged during the honest execution of the oracle $\Pi_{i,j}^s$ of the protocol. This query models the passive attacks.
- *Send*$(\Pi_{i,j}^s, m)$ With this query, $\mathcal{A}$ can send a message m to the oracle $\Pi_{i,j}^s$ and then the user U_i responds to $\mathcal{A}$ according to the protocol description. This query models the capabilities of $\mathcal{A}$ who can initiate sessions and modify, delay or insert new protocol messages.
- *Reveal*$(\Pi_{i,j}^s)$ This query returns the session key of $\Pi_{i,j}^s$ to $\mathcal{A}$ if $\Pi_{i,j}^s$ has an accepted session key, otherwise it returns $\perp$. The goal of the known key security. This query models the goal of the known key security.
- *RevealEph*$(\Pi_{i,j}^s)$ With this query, $\mathcal{A}$ obtains the ephemeral secret key (random number) of U_i used by the oracle $\Pi_{i,j}^s$. This query models the goal of the known session-specific temporary information attack or ephemeral secret leakage attack.
- *Corrupt*(ID_i) With this query, $\mathcal{A}$ obtains the long-term private key of U_i. However, this query returns neither the session key nor the internal state. This query captures the security goals related to the compromise of long-term private key and the behavior of malicious users. These goals include forward secrecy, key compromise impersonation resilience and security against unknown key share and insider attacks.
- *Test*$(\Pi_{i,j}^s)$ The adversary $\mathcal{A}$ is allowed to send a *Test* query to the oracle $\Pi_{i,j}^s$ only once provided $\Pi_{i,j}^s$ is accepted. However, $\mathcal{A}$ can make this query at any time. On receiving a *Test* query, $\Pi_{i,j}^s$ chooses a random bit $b \in \{0,1\}$ and returns the real session key if $b = 1$.Otherwise, a random value uniformly chosen from the session key distribution is returned.

Definition 5 An oracle $\Pi_{i,j}^{s}$ is called accepted if it holds (1) a session key (2) a session identifier $sid_{i,j}^{s}$ and (3) a partner identifier $pid_{i,j}^{s}$.

Definition 6 Two oracles of $\Pi_{i,j}^{s}$ and $\Pi_{j,i}^{t}$ are said to be partnered iff the following holds: (1) $\Pi_{i,j}^{s}$ and $\Pi_{j,i}^{t}$ have accepted, (2) $sid_{i,j}^{s} = sid_{j,i}^{t}$ and (3) $pid_{i,j}^{s} = pid_{j,i}^{t}$.

Definition 7 An oracle $\Pi_{i,j}^{s}$ $(i \neq j)$ is called fresh iff the following holds: (1) $\Pi_{i,j}^{s}$ if it has accepted and therefore holds a session key; (2) $\Pi_{i,j}^{s}$ or its partner $\Pi_{j,i}^{t}$ (if any) has not been asked a *Reveal* query after their acceptance; (3) a *Corrupt* query has not been asked by the user U_j.

Definition 8 An adversary $\mathcal{A}$ is called *benign* if it restricts its action to choosing a pair of oracles $\Pi_{i,j}^{s}$ and $\Pi_{j,i}^{t}$, and then faithfully conveying each flow from one oracle to the other.

Definition 9 A key agreement protocol maintains session key secrecy, if no polynomial time bounded adversary $\mathcal{A}$ has a non-negligible advantage in the following game played between $\mathcal{A}$ and infinite set of oracles $\Pi_{i,j}^{s}$ for $U_i \in \mathcal{U}$.

(1) A long-term private key is assigned to each user.
(2) $\mathcal{A}$ may ask several queries and get back the response from the corresponding oracles.
(3) There is no $Reveal(\Pi_{i,j}^{s})$ query or $Corrupt(ID_i)$ query being asked before the $Test(\Pi_{i,j}^{s})$ query has been asked.
(4) $\mathcal{A}$ may ask other queries during asking the $Test(\Pi_{i,j}^{s})$ query where $\Pi_{i,j}^{s}$ is fresh. $\mathcal{A}$ outputs its guess bit b' for the bit b which is chosen in the $Test(\Pi_{i,j}^{s})$ query eventually and the game is terminated.

Definition 10 The authenticated key agreement (AKA) advantage $Adv_{\mathcal{A}}^{2PAKA}(k) = [2Pr|b^{'} = b| - 1]$ of $\mathcal{A}$ is defined as the success probability to win the above game by violating the AKA security of an execution of a two-party key agreement protocol (2PAKA), if $\mathcal{A}$ asks a single $Test(\Pi_{i,j}^{s})$ query and correctly guesses a bit b, which is selected by $Test(\Pi_{i,j}^{s})$ query.

Definition 11 A two-party authenticated key agreement (2PAKA) is AKA secure if the following holds:

(1) In the presence of a probabilistic polynomial time bounded adversary $\mathcal{A}$, two oracles $\Pi_{i,j}^{s}$ and $\Pi_{j,i}^{t}$ accept the same key $SK_{i,j}$.
(2) For every $\mathcal{A}$, $Adv_{\mathcal{A}}^{2PAKA}(k)$ is negligible.

4.2 Security Analysis

Theorem 1 *The proposed 2PAKA protocol is provably secure in the random oracle model under the Computational Diffie-Hellman (CDH) assumption.*

Proof Let $Adv_{\mathcal{A}}^{2PAKA}(k)$ be the success probability of a probabilistic polynomial time bounded adversary who tries to breach our protocol, then we can construct an algorithm $\mathcal{C}$ that will solve the CDH problem with the help of $\mathcal{A}$ i.e., $\mathcal{C}$ outputs abP from a given CDH instance $(P, Q_1 = aP, Q_2 = bP)$. We define the following challenge-response game played between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$. In this game, we assume that n_1 is the number of participants, n_2 is the number of sessions each participant may be involved with its partner and $\mathcal{A}$ asks at most q_i times H_i queries, where the hash function H_i $(i = 0, 1, 2)$ closely

behaves like true random oracle. To solve the CDH problem, $\mathcal{C}$ chooses a $P_0 \in G_q$, sets $P_S = P_0$ and then returns system parameter $\Omega = \{F_q, E/F_q, G_q, P, P_0, H_0, H_1, H_2\}$ to $\mathcal{A}$. Then $\mathcal{A}$ selects $I, J \in \{1, 2, \ldots, n_1\}$ $(I \neq J)$, $T \in \{1, 2, \ldots, n_2\}$ and asks the following queries adaptively to $\mathcal{C}$ and then $\mathcal{C}$ answers $\mathcal{A}$'s queries as follows:

- *Hash queries to* H_0 $\mathcal{C}$ maintains an initially-empty list L_{H0}^{list} that contains tuples of the form (ID_i, P_i, h_0^i). If the adversary $\mathcal{A}$ a asks a H_0 query with (ID_i, P_i), then $\mathcal{C}$ returns the previous value h_0^i if a tuple (ID_i, P_i, h_0^i) is found in L_{H0}^{list}. Otherwise, $\mathcal{C}$ chooses a $h_0^i \in_R Z_q^*$ such that there is no item $(\cdot, \cdot, h_0^i)$ in L_{H0}^{list} and returns h_0^i to $\mathcal{A}$. Now $\mathcal{C}$ inserts the tuple (ID_i, P_i, h_0^i) into the list L_{H0}^{list}.
- *Hash queries to* H_1 $\mathcal{C}$ maintains an initially-empty list L_{H1}^{list} that contains tuples of the form (ID_i, T_i, R_i). If the adversary $\mathcal{A}$ a asks a H_1 query with (ID_i, T_i), then $\mathcal{C}$ returns the previous value R_i if a tuple (ID_i, T_i, R_i) is found in L_{H1}^{list}. Otherwise, $\mathcal{C}$ chooses a $R_i \in_R Z_q^*$ such that there is no item $(\cdot, \cdot, R_i)$ in L_{H1}^{list} and returns R_i to $\mathcal{A}$. Now $\mathcal{C}$ inserts the tuple (ID_i, T_i, R_i) into the list L_{H1}^{list}.
- *Registration*(ID_i) *queries* $\mathcal{C}$ maintains an initially-empty list L_R^{list} consisting of tuples of the form (ID_i, d_i, Q_i). If the adversary $\mathcal{A}$ a asks a $Registration$ query with ID_i, $\mathcal{C}$ searches the list L_{H0}^{list}. If a tuple (ID_i, P_i, h_0^i) is on L_{H0}^{list}, $\mathcal{C}$ will do nothing. Otherwise, $\mathcal{C}$ selects $d_i, h_0^i \in_R Z_q^*$, computes $P_i = d_i P - h_0^i P_0$, $Q_i = d_i P$ and then answers as follows:
  - If $ID_i = ID_I$, $\mathcal{C}$ returns $(ID_i, \perp, aP)$ to $\mathcal{A}$.
  - If $ID_i = ID_J$, $\mathcal{C}$ returns $(ID_i, \perp, bP)$ to $\mathcal{A}$.
  - Else, $\mathcal{C}$ returns (ID_i, d_i, Q_i) to $\mathcal{A}$.

  Finally, $\mathcal{C}$ inserts the tuples (ID_i, P_i, h_0^i) and (ID_i, d_i, Q_i) into the lists L_{H0}^{list} and L_R^{list}.
- *Send*$(\Pi_{i,j}^s, m)$ *queries* $\mathcal{C}$ maintains an initially-empty list L_S^{list} consisting of tuples of the form $(\Pi_{i,j}^s, Trans_{i,j}^s, a_i)$, where $Trans_{i,j}^s$ is the transcript of $\Pi_{i,j}^s$. $\mathcal{C}$ answers the query as follows:
  - If $\Pi_{i,j}^s = \Pi_{I,J}^T$ and m is the first message, $\mathcal{C}$ searches a tuple (ID_I, T_I^T, R_I^T) into L_{H1}^{list}, chooses $a_I \in_R Z_q^*$ and returns $(a_I Q_1, R_I^T)$ as the answer. $\mathcal{C}$ then inserts the tuple $(\Pi_{I,J}^T, Trans_{I,J}^T, a_I)$ to the list L_S^{list}.
  - If $\Pi_{i,j}^s = \Pi_{I,J}^T$ and m is the second message, $\mathcal{C}$ searches a tuple (ID_J, T_J^T, R_J^T) into L_{H1}^{list}, chooses $a_J \in_R Z_q^*$ and returns $(a_J Q_2, R_J^T)$ as the answer. $\mathcal{C}$ then inserts the tuple $(\Pi_{I,J}^T, Trans_{I,J}^T, a_J)$ to the list L_S^{list}.
  - Else, $\mathcal{C}$ searches a tuple (ID_i, T_i^s, R_i^s) into L_{H1}^{list}, chooses a number $a_i \in_R Z_q^*$ and then replies with $(a_i Q_i, R_i^s)$. $\mathcal{C}$ updates the tuple indexed by $\Pi_{i,j}^s$ in the list L_S^{list}.
- *Corrupt*(ID_i) *queries* $\mathcal{C}$ answers this query as follows:
  - If $ID_i = ID_I$ or $ID_i = ID_J$, $\mathcal{C}$ aborts.
  - Else, $\mathcal{C}$ searches the list L_R^{list} for a tuple (ID_i, d_i, Q_i) and returns d_i to $\mathcal{A}$.
- *Reveal*$(\Pi_{i,j}^s, m)$ *queries* $\mathcal{C}$ maintains an initially-empty list L_{RV}^{list} consisting of tuples of the form $(ID_{in}, ID_{re}, \Pi_{i,j}^s, T_{i,j}^s, T_{j,i}^s, SK_{i,j}^s)$, where ID_{in} is the identification of the initiator in the session which $\Pi_{i,j}^s$ engages in and ID_{re} is the identification of the responder. $\mathcal{C}$ answers the query as follows:

– If $\Pi_{i,j}^s = \Pi_{I,J}^T$, $\mathcal{C}$ aborts.

– If $ID_i \neq ID_I$ or $ID_i \neq ID_J$
  – $\mathcal{C}$ searches the lists L_S^{list}, L_R^{list} and L_{H1}^{list} for corresponding tuples $(\Pi_{i,j}^s, Trans_{i,j}^s, a_i)$, (ID_i, d_i, Q_i) and (ID_i, T_i^s, R_i^s) separately. Then, $\mathcal{C}$ computes $K_{i,j}^s = (a_i d_i) T_{j,i}^s$.
  – $\mathcal{C}$ makes a H_2 query. If $\Pi_{i,j}^s$ is the initiator oracle then the query is of the form $(ID_i \parallel ID_j \parallel Trans_{i,j}^s \parallel K_{i,j}^s)$.
– Else, $\mathcal{C}$ chooses $SK_{i,j}^s \in_R Z_q^*$.

- *RevealEph*$(\Pi_{i,j}^s)$ *queries* If $\mathcal{A}$ asks a *RevealEph*$(\Pi_{i,j}^s)$ query, $\mathcal{C}$ searches the list L_S^{list} for a tuple $(\Pi_{i,j}^s, Trans_{i,j}^s, a_i)$ and then returns the ephemeral secret a_i to $\mathcal{A}$.
- *Hash queries to* H_2 $\mathcal{C}$ maintains an initial-empty list L_{H2}^{list} of the form $(ID_i^u, ID_j^u, T_i^u, T_j^u, K_{i,j}^u, h_2^u)$ and responds with H_2 queries as follows:
  - If a tuple indexed by $(ID_i^u, ID_j^u, T_i^u, T_j^u, K_{i,j}^u, h_2^u)$ is already in L_{H2}^{list}, $\mathcal{C}$ replies with the corresponding h_2^u.
  - Else, if there is such a tuple indexed by $(ID_i^u, ID_j^u, T_i^u, T_j^u)$ in the list L_{RV}^{list}, then $\mathcal{C}$ obtains the corresponding $SK_{i,j}^s$ and sets $SK_{i,j}^s = h_2^u$. Otherwise $\mathcal{C}$ chooses $h_2^u \in_R Z_q^*$.
  - Else $\mathcal{C}$ chooses $h_2^u \in_R Z_q^*$ and inserts the tuple $(ID_i^u, ID_j^u, T_i^u, T_j^u, K_{i,j}^u, h_2^u)$ into the list L_{H2}^{list}.
- *Test*$(\Pi_{I,J}^T)$ *queries* At some point, $\mathcal{C}$ will ask a *Test* query on some oracle. If $\mathcal{C}$ does not choose one of the oracles $\Pi_{I,J}^T$ to ask the *Test* query, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ simply outputs a random value from the set Z_q^*.

The challenger $\mathcal{C}$ chooses $\Pi_{I,J}^T$ as the *Test* oracle i.e., $ID_i = ID_I, ID_j = ID_J$ and $s = T$ with probability $\frac{1}{n_1^2} \times \frac{1}{n_2}$. In this case, $\mathcal{C}$ would not have made $Corrupt(\Pi_{I,J}^T)$ or $Reveal(\Pi_{I,J}^T)$ queries, and so $\mathcal{C}$ would not have aborted. To win this game $\mathcal{C}$ must have made the corresponding H_1 and H_2 queries of the form (ID_I^T, T_I^T, R_I^T) and $(ID_I^T, ID_J^T, T_I^T, T_J^T, K_{I,J}^T)$. If $\Pi_{I,J}^T$ is the initiator oracle or else (ID_J^T, T_J^T, R_J^T) and $(ID_J^T, ID_I^T, T_J^T, T_I^T, K_{J,I}^T)$ with overwhelming probabilities because H_1 and H_2 are random oracles. Thus, $\mathcal{C}$ can find the corresponding R_I^T and $K_{I,J}^T$ into the lists L_{H1}^{list} and L_{H2}^{list} with probabilities $\frac{1}{q_1}$ and $\frac{1}{q_2}$ respectively, and then outputs $abP = (a_I a_J)^{-1} K_{I,J}^T$ as the solution to the CDH instance $(P, Q_1 = aP, Q_2 = bP)$.

Thus, the advantage of $\mathcal{C}$ solving the CDH problem is such that

$$Adv_{\mathcal{C}}^{CDH}(k) \geq \frac{1}{q_1} \times \frac{1}{q_2} \times \frac{1}{n_1^2} \times \frac{1}{n_2} Adv_{\mathcal{A}}^{2PAKA}(k)$$

5 Formal Security Validation of our Protocol Using AVISPA Software

In this section, we simulated the proposed protocol using AVISPA software [50,51]. This is a widely used internet security protocols and its application validation tool and several protocols [44–49] have been analyzed using it. It is a push-button tool developed based on the intruder model initiated by Dolev and Yao [52]. In this model following assumptions have been made, (1) the communication network is totally under control by an intruder (active and passive) that is, the messages exchanged by the different users go to the intruder and he has the ability to send them everywhere in the network, (2) the intruder can forward, modify, replay, suppress and synthesize the messages however, he cannot violate the underlying

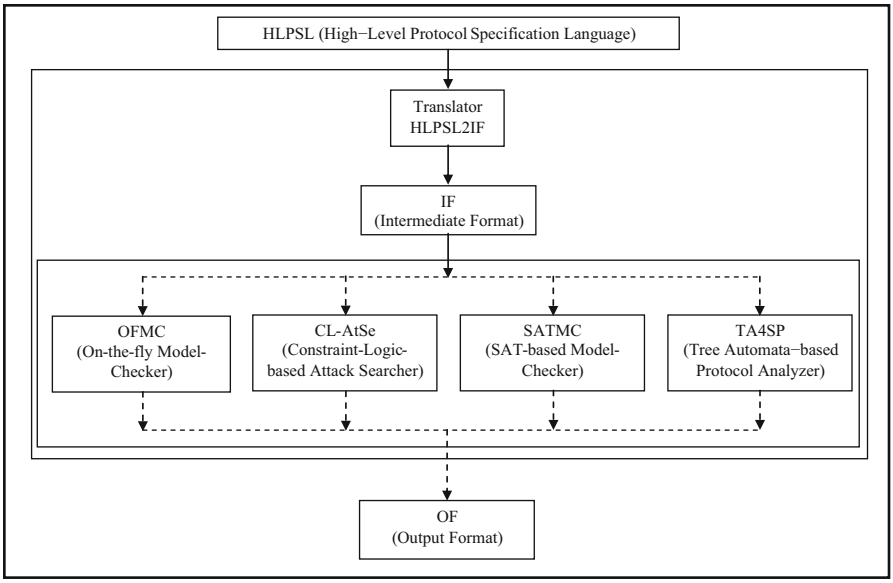

Fig. 3 Architecture of the AVISPA tool

cryptographic computational problems and (3) the intruder can play role(s) of user(s) and gain knowledge of compromised users. For the implementation of cryptographic protocols, the HLPSL (High Level Protocol Specification Language) is used and four model checkers/back-ends, called OFMC (On-the-fly Model-Checker), CL-AtSe (Constraint-Logic-based Attack Searcher), SATMC (SAT-based Model-Checker) and TA4SP (Tree Automata-based Protocol Analyzer) [51] are used for the simulation purpose and to analyze various security properties such as secrecy of keys, authentication, freshness and robustness against replay attacks.

The HLPSL is an expressive, modular, role-based and formal language that helps to describe each participant's role, composition rules for the representation of basic roles, control-flow patterns, adversary models, as well as security features. The architecture of the AVISPA software [49–51] is given in Fig. 3 for further understanding and the brief descriptions of these model checkers are given as follows:

- *OFMC* This back-end builds the infinite tree defined by the protocol analysis problem and executes different symbolic techniques to search the state space in a demand-driven way i.e., on-the-fly. It helps to detect the attacks, verify the correctness of the protocol for a bounded number of sessions, but without limiting the number of messages an intruder can generate.
- *CL-AtSe* This back-end is used to detect the attacks on the protocol by using a set of constraints that are obtained by translating the security protocol specification written in the Intermediate format (IF). The detection of attacks and the translation of protocol specification, which are designed based on the adversary's knowledge, are fully automated and internally performed by CL-AtSe model checker.
- *SATMC* This back-end is used to explore the state space through several symbolic techniques. Note that it also detects attacks on protocols and validates the security requirements by using a bounded number of sessions.

```
role alice(
      A, B : agent,
      SND, RCV : channel(dy),
      H, Union, Pred : hash_func,
      Ka, Kb, P :  symmetric_key)

  played_by A
  def=
  local State : nat, IDa, IDb,
      Pa, Pb : public_key,
      Da, Db, Sk : symmetric_key,
     Ra, Rb : text,
     Ta, Tb, Sa, Sb : message
const aliceid : protocol_id

init State:=0

transition
      1.State=0/\RCV(start)=|>
        State' :=2/\Ra' :=new()
                  /\Ta' :=Pred(Pred(Ra, Da), P)
                  /\Ka' :=Pred(Da, Pb)
                  /\Sa' :=H(Union(Ta, Ka))
                  /\SND(IDa.Ta.Sa)

  2.State=2/\RCV(IDb.Tb'.Sb')/\Sb'=H(Union(Tb, Ka))=|>
    State':=4/\Sk':=H(Union(Ta, Union(Tb, Union(Sa, Union(Sb, Pred(Pred(Ra, Da), Tb))))))
              /\secret(Da, aliceid, {a, b})
              /\request(A, B, x1, Da)
  end role
```

Fig. 4 Role specification of the Initiator (entity A) of the proposed 2PAKA protocol in HLPSL

- *TA4SP* Based on propositional formula and regular tree languages, this back-end approximates the intruder knowledge (over or under) using unbounded number of sessions.

For the formal verification and security analysis of the protocols, the AVISPA tool has been integrated with a graphical user interface, called SPAN (Security Protocol ANimator). The protocol specification in HLPSL has four sections, called role, session, environment and goal. To analyze a cryptographic protocol on AVISPA the following steps are executed (1) the protocol is implemented in HLPSL specification, (2) the AVISPA tool converts this specification into IF automatically by using a translator, called HLPSL2IF translator and (3) the IF specification is given to the back-ends of the AVISPA tool to analyze whether there is any active and passive attacks.

The IF is a low-level language which contains some information about IF syntax for back-ends, the description of mathematical properties of operators (e.g., exponentiation, bit-wise XOR etc.) and the intruder's behavior. After the execution of IF, each model checker of AVISPA returns the simulation results of the protocol by analyzing to output format (OF), which illustrates that the given protocol was safe or unsafe against the intruder(s). Now, we designed our 2PAKA protocol on AVISPA tool using HLPSL and the role specifications of the initiator and responder are shown in Figs. 4 and 5. The Figs. 6 and 7 are the simulation results of our scheme through OFMC and CL-AtSe back-ends. The simulation results prove that the proposed scheme is safe against both the active and passive adversaries.

```
role bob(
      B, A : agent,
      SND, RCV : channel(dy),
      H, Union, Pred: hash_func,
     Kb, Ka, P :  symmetric_key)

 played_by B
 def=
 local State : nat, IDa, IDb,
     Pa, Pb : public_key,
    Da, Db, Sk : symmetric_key,
    Ra, Rb : text,
    Tb,Ta, Sb, Sa : message
 const bobid: protocol_id

 init State:=1

 transition
 1.State=1/\RCV(IDa.Ta'.Sa')/\Sa'=H(Union(Ta, Kb))=|>
   State':=3/\Rb':=new()
              /\Tb' :=Pred(Pred(Rb, Db), P)
              /\Kb':=Pred(Db, Pa)
              /\Sb':=H(Union(Tb, Kb))
              /\Sk' :=H(Union(Ta, Union(Tb, Union(Sa, Union(Sb, Pred(Pred(Rb, Db), Ta))))))
              /\SND(IDb.Tb.Sb)
              /\secret(Db, bobid, {b, a})
              /\request(B, A, x2, Db)
 end role
```

Fig. 5 Role specification of the Responder (entity B) of the proposed 2PAKA protocol in HLPSL

Fig. 6 Simulation result of the proposed 2PAKA protocol in OFMC model checker

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/avispa/web-interface-
computation/./tempdir/workfilekB3vvi.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.04s
  visitedNodes: 7 nodes
  depth: 3 plies
```

6 Further Security Analysis of the Proposed Protocol

An informal security analysis of the proposed protocol is given in this section. First of all, it supports both the implicit and explicit key authentication properties. Because the proposed protocol instead of two short-term secrets a and b, exchanges $T_A = aQ_A$ and $T_B = aQ_B$ with their corresponding signatures R_A and R_B of the entities A and B through a public channel. Since the adversary $\mathcal{A}$ cannot forge the signatures R_A and R_B without the knowledge of static private keys d_A or d_B, so $\mathcal{A}$ cannot impersonate to

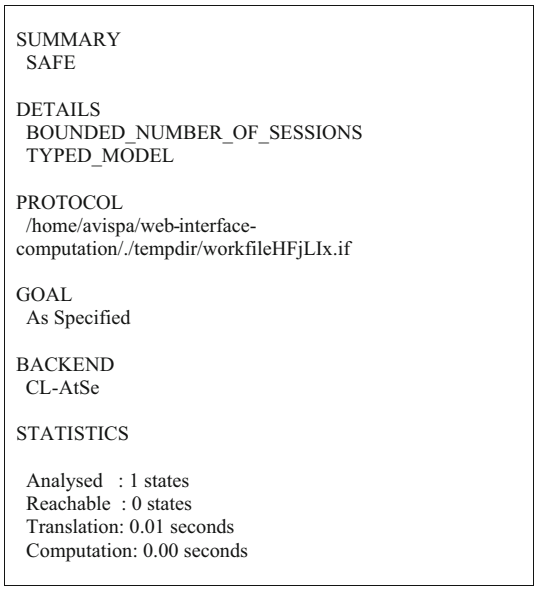

Fig. 7 Simulation result of the proposed 2PAKA protocol in CL-AtSe model checker

any entity and would not be able to compute the final session key, and thus confirms the correct session key of the participants. So the implicit key authentication property is provided by the proposed protocol. Also it supports explicit key authentication property as according to Blake-Wilson et al. [4] that a key agreement protocol supports explicit key authentication if it has both session key confirmation and implicit key agreement properties. The proposed protocol also satisfies other security attributes, for instances, man-in-the-middle attack, known-key secrecy, key-compromise impersonation resilience, unknown key-share resilience, perfect forward secrecy, known session-specific temporary information attack, no key control, etc., the details of which can be found in [4]. The detailed analysis of the proposed protocol to support the above security attributes are given below.

6.1 Man-in-the-Middle Attack

As stated, the proposed protocol exchanges $T_A = aQ_A$ and $T_B = bQ_B$ along with the signatures R_A and R_B, and generates the session key $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$ using two static private keys d_A and d_B, and two short-term keys a and b of the participants. Since the users can authenticate T_A and T_B using R_A and R_B very easily, a valid session key SK is generated. Note that MIMA is only possible in our scheme if $\mathcal{A}$ can forge R_A and R_B and/or compute $d_A d_B P$ from the pair $(Q_A, Q_B) = (d_A P, d_B P)$ which is not possible as it is a hard computational Diffie-Hellman problem (CDHP). Thus, the proposed protocol protects MIMA.

6.2 Known-Key Attack

The 2PAKA protocol satisfies the known-key security if the knowledge of previously computed session key does not allow an adversary to compromise the past or future session keys. Assume that a previous session key generated by the proposed protocol is disclosed to an adversary $\mathcal{A}$. However, $\mathcal{A}$ is unable to derive all past and future session keys from the

knowledge of the disclosed session key. To derive a past session key $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$, $\mathcal{A}$ has to compute the partial session key $K = K_A = K_B = abd_Ad_BP$ of that session, which depends on two ephemeral secrets a and b, and the private keys d_A and d_B of the participating entities. Since T_A and T_B of each session are known to $\mathcal{A}$, and until (d_A, d_B) and (a, b) are computed from (Q_A and Q_B) and (T_A and T_B), respectively, no past or future session keys are compromised. Because these computations involve solving ECDLP in polynomial-time, which is not possible as no such algorithm exists in reality. Therefore, the known-key security (K-KS) property is preserved in the proposed protocol.

6.3 Key-Compromise Impersonation Attack

In K-CI attack, an adversary $\mathcal{A}$ having the knowledge of long-term private key of a participant, say d_A of the participant A, may impersonate other participant B to A and try to obtain the correct session key established between the participants. An authenticated key agreement protocol should resist this attack. Suppose that $\mathcal{A}$ who knows d_A can compute d_Ad_BP using B's public key $Q_B = d_BP$ and sends ($ID_B, T_B = bQ_B, R_B = H_1(T_B \parallel d_AQ_B)$) to A, where $b \in_R Z_q^*$ is selected by $\mathcal{A}$. Similarly A computes (ID_A, T_A, R_A) and sends them to $\mathcal{A}$. To derive the session key $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$, $\mathcal{A}$ must obtain $K = abd_Ad_BP$, which requires the knowledge of the private key d_B of B. Since due to the difficult of ECDLP, it is not possible to derive d_B from B's public key $Q_B = d_BP$, the correct session key SK cannot be computed by $\mathcal{A}$. Thus, the proposed protocol resists the K-CI attack.

6.4 Unknown Key-Share Attack

After successful completion of a key agreement, an entity, say A out of two entities A and B, believes that a correct session key with entity B has been established but, the same may not be true to the entity B and he mistakenly believes that the session key instead of A has been established with an adversary $\mathcal{A}$. Note that this cannot exist in the proposed protocol, because both of them compute the common session key from the authentication tokens T_A and T_B validated by their signatures R_A and R_B, and due to ECDLP, the long-term private keys never be obtained from the public keys of the entities. Thus, the proposed protocol is immune from unknown key-share (U-KS) attack.

6.5 Perfect Forward Secrecy

A key agreement protocol satisfies forward secrecy (FS) if the secrecy of a previously generated session key is not compromised even if the private key of one or more entities but, not all are known to an adversary. And a protocol has perfect forward secrecy (PFS) property if an adversary having the knowledge of the private keys of all entities is unable to acquire any previously generated session key. Now if the long-term private keys d_A and d_B of A and B in the proposed protocol are disclosed to an adversary $\mathcal{A}$, the session key cannot be computed because $\mathcal{A}$ needs to derive the session ephemeral secrets a and b from T_A and T_B by solving ECDLP. Since it is not solvable with a polynomial-time bounded algorithm, therefore, our protocol supports perfect forward secrecy.

6.6 Known Session-Specific Temporary Information Attack

In 2001, Canetti and Krawczyk [7] initially investigated the known session-specific temporary information attack (KSSTIA) and later on, it is further studied by Cheng et al. [12], where

it is pointed out that the security of the generated session keys should not be compromised even if the session ephemeral secrets a and b are leaked to an adversary $\mathcal{A}$. In the proposed protocol, both A and B compute their session key as $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$. Note that $\mathcal{A}$ can derive SK if he knows $K = abd_Ad_BP$. But $\mathcal{A}$ cannot derive K even if a and b are disclosed, because d_A and d_B are not known to $\mathcal{A}$. Also $\mathcal{A}$ cannot derive directly from the pair $(T_A, T_B) = (ad_AP, bd_BP)$ because it is required to solve the CDHP problem and it is not breakable by a polynomial-time bounded algorithm. Furthermore, $\mathcal{A}$ cannot use the pair (Q_A, Q_B) to derive the session key SK, because it needs to compute d_Ad_BP from $(Q_A, Q_B) = (d_AP, d_BP)$, which is difficult as to solve the CDHP. Thus, we may conclude that the proposed protocol sustains against known session-specific temporary information attack.

6.7 Key Off-set Attack/Key Replicating Attack

The key off-set/key replicating attack (KOA/KRA) is a variation of MIMA, where an active adversary intercepts and modifies the messages exchanged between two entities in a session, and enforces the entities to agree upon a wrong session key, although this attack does not allow the adversary to gain any knowledge of the agreed session key. It is also a violation of the key integrity property, which indicates that any accepted session key should depend on the inputs, exchanged using the protocol. Blake-Wilson et al. [4] pointed out that a two-flow authenticated key agreement protocol without key conformation is vulnerable to KOA/KRA. The entities A and B in the proposed protocol exchange (ID_A, T_A, R_A) and (ID_B, T_B, R_B) between each other and an active adversary $\mathcal{A}$ can easily offset some of these values, say T_A and T_B by an unknown exponent ϵ and produces ϵT_A and ϵT_B. Nevertheless $\mathcal{A}$ cannot compute $R_A^* = H_1(\epsilon T_A \parallel d_AQ_B)$ and $R_B^* = H_1(\epsilon T_B \parallel d_BQ_A)$, because the derivation of d_AQ_B or d_BQ_A requires the knowledge of the long-term static private keys d_A or d_B of the entities. Therefore, A and B easily detect this attack using R_A and R_B, and hence, the KOA/KRA attack is infeasible in the proposed protocol.

6.8 No Key Control

As stated earlier, both the entities in the proposed protocol generate a common secret session key $SK = H_2(ID_A \parallel ID_B \parallel Trans \parallel K)$. Since $K = abd_Ad_BP$ and the values of a, b in K are chosen by the entities A and B randomly, so neither an entity nor an adversary can enforce the session key to be a predetermined value and/or lie within a set having small number of elements. Hence, we claim that the proposed protocol provides no key control (NKC) attribute.

6.9 Reflection attack

The proposed protocol is free from reflection attack (RA) and U-KS attack, because according to Boyed and Choo [6], the identities of the participating entities are included in the hash function H_2, which is used to derive the common session key. Also the proposed protocol provides freshness and data origin authentication as the transcript *Trans* is included in the H_2 function.

- *Cryptanalysis of Tsaur's two-party key agreement protocol* Based on SC-PKC setting, Tsaur [59] proposed a key agreement protocol, called time-variant session key agreement protocol. In this protocol, A and B exchanged their contributions $(ID_A, T_A = aP, Q_A)$ and $(ID_B, T_B = bP, Q_B)$ between them, where $(d_i, Q_i = d_iP)$ is the private/public key

pair of the entity $ID_i, i \in \{A, B\}$ and $a, b \in [2, n-2]$ are selected by A and B respectively. A computes the session key as $SK = a(Q_B) + d_A(T_B) = aQ_B + bQ_A$ and B computes the session key as $SK = b(Q_A) + d_B(T_A) = bQ_A + aQ_B$. However, Tsaur's protocol has some common security loopholes, which are briefly described here:

1. We claim that Tsaur's protocol has security flaws against perfect forward secrecy (Sect. 6.5). Suppose that the private keys d_A and d_B of A and B are leaked to an adversary $\mathcal{A}$, who then captures the messages $(ID_A, T_A = aP, Q_A)$, $(ID_B, T_B = bP, Q_B)$ and computes the session key $SK = d_A T_B + d_B T_A = d_A bP + d_B aP = bd_A P + ad_B P = bQ_A + aQ_B$ of that session. Accordingly, the protocol is also vulnerable to PKG-FS attack.
2. According to the discussion made in Sect. 6.6, it is to be noted that both of A and B computes the session key $SK = bQ_A + aQ_B$. Therefore, if a and b are known to an adversary $\mathcal{A}$, then he can easily calculate $SK = b(Q_A) + a(Q_B)$. Thus, Tsaur's protocol is vulnerable to known session-specific temporary information attack.
3. Tsaur's protocol is also weak against the KOA/KRA as given in Sect. 6.7. In this protocol, A and B exchange their messages $(ID_A, T_A = aP, Q_A)$ and $(ID_B, T_B = bP, Q_B)$ between them through open channel. Suppose that, an adversary $\mathcal{A}$ modifies these messages as $(ID_A, T_A^* = cT_A = caP, Q_A)$ and $(ID_B, T_B^* = cT_B = cbP, Q_B)$, and forwards to A and B respectively. Then A computes the session key $SK_A = a(Q_B) + d_A(T_B) = aQ_B + d_A cbP = aQ_B + cbQ_A = cbQ_A + aQ_B$, B computes the session key $SK_B = b(Q_A) + d_B(T_A) = bQ_A + d_B acP = bQ_A + acQ_B$ and $SK_A \neq SK_B$. Therefore, Tsaur's protocol is not secure against key off-set attack/key replicating attack.

The proposed protocol in terms of security attacks as mentioned above has been compared with a number of protocols [8,9,13,18,19,21,24,29,32,33,40,55–59] proposed recently and the outputs are given in Table 1. As shown, none of the protocols except our proposed one can protect all the attacks. However, the protocols [9,13,21,56–58] are comparatively efficient as they cannot defend a single attack. Note that the KOA/KRA are only protected by the two protocols namely our proposed technique and Choie et al. [13]. The reason of the failure of the remaining protocols is that they exchange messages in the form aP or aQ_A without their signatures as according to Blake-Wilson et al. [4], and McCullagh-Barrato [24], until the two-flow authenticated key agreement protocols support key conformation property, the KOA/KRA is not protected. Furthermore, the protocols [8,13,55,59] are insecure against the known session-specific temporary information attack, as the session key of these protocols depends on the session ephemeral secrets a and b only. Thus, the secrecy of the session key will be compromised to an adversary with the disclosure of a and b.

7 Performance Evaluation of the Proposed Protocol

The performance of any key agreement protocol mainly depends on the execution of the following two decisive factors, and a brief introduction of them is given below:

- *Computation cost* It is the execution cost required to perform different operations and their repetition involved in generating a common session key between the entities. Since different key agreement techniques follow different operations and accordingly the operations like elliptic curve point addition, simple hash operation, etc., require comparatively lesser computation cost than the operations such as bilinear pairing, elliptic curve scalar point

Table 1 Security comparisons

Protocols	PKG-FS	K-CI	PFS	MIMA	K-KS	RA	KSSTIA	KOA	NKC	IA	U-KS
Chen-Kudla [9]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Choie-I [13]	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Choie-II [13]	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Choie-III [13]	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Hölbol-Welzer [18]	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Hsieh et al. [19]	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
McCullagh-Barreto [24]	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Kudla-Paterson [21]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Ryu et al. [29]	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
Shim [32]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes
Smart [33]	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Xie [40]	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes
Cao et al. [8]	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Cao et al. [55]	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Zu-hua [56]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Ni et al. [57]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Wang et al. [58]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Tsaur [59]	No	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Proposed	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

PKG-FS PKG Forward secrecy; *IA* Impersonation attack; *Yes* Protect the attack; *No* Do not protect the attack

multiplication, modular exponentiation, etc. The computation cost of 2PAKA protocols can be reduced if a fewer numbers of expensive operations are used in their implementation. Such protocols have an advantage that they are the most suitable for resource-limited (e.g., power, storage, bandwidth, etc.) environments such as in smartcards, mobile networks, etc.

- *Communication cost* The communication cost is another important factor for measuring the performance of a 2PAKA protocol. It includes the number of rounds, the number of steps per round and message-length used by the entities for establishing the authenticated session key between them. The higher is the amount of communication costs required means to spend more time by participating entities to establish the session key. The increase in communication cost leads to more communication latency and thus involves more delay in the transmit - response phase of the users. For these reasons a 2PAKA protocol with high communication cost is unsuitable for telecommunication system such as online pay-TV, online money transaction, online e-voting, etc., that needs a quick response for any required service.

In order to calculate the computation cost, the type and number of operations involved in some relevant protocols are given below.

Hölbl and Welzer protocol-I [18] requires each entity to perform 4 modular exponentiations, 2 modular multiplications and for Hölbl and Welzer protocol-II [18] requires 3 modular exponentiations, 2 modular multiplications. Chen and Kudla's protocol [9] executes two elliptic curve scalar point multiplications, one elliptic curve point addition and two pairing operation per entity. Smart's 2PAKA protocol [33] established the session key

with executing two scalar point multiplications, two pairing operations (one of which can be partially pre-computed, thus consider its cost equivalent to one scalar point multiplication), one pairing operation and one pairing-based exponentiation operation per entity. McCullagh and Barreto's protocol [24] requires one pairing operation, two elliptic curve scalar point multiplications and one pairing-based exponentiation operation per entity. Wang et al.'s protocol (improved Ryu's protocol) [38] employs two elliptic curve scalar point multiplications and one pairing operation. Choie et al. protocol-I [13] executes two elliptic curve scalar point multiplications and three elliptic curve point additions per entity and Choie et al.'s protocol-II (modified Smart's protocol) [13] requires two elliptic curve scalar point multiplications and two elliptic curve point additions per entity. Cao et al.'s protocol [8] requires five elliptic curve scalar point multiplications and two elliptic curve point additions per entity. Wang et al.'s protocol [39] is implemented using modified Weil pairing or Tate pairing and executes one elliptic curve scalar point multiplication, one pairing operation, and one pairing-based exponentiation operation per entity. In Kudla-Paterson's protocol [21], each entity executes three pairing based exponentiation on the underlying multiplicative group whereas other bilinear pairing operations are executed in off-line mode. Wang et al.'s protocol [58] needs one elliptic curve scalar point multiplication and three pairing operations per entity to compute a session key whereas Cao et al.'s protocol [55] needs only five elliptic curve scalar point multiplications per entity for the same. Zu-zhu's protocol computes two scalar point multiplications, three pairing operations and two pairing-based exponentiation operations per entity. Zhu et al.'s protocol [43] computes the session key by performing five scalar point multiplications per entity and Tsaur's protocol [59] also requires the same amount of time. In Ni et al.'s protocol [57], four scalar point multiplications, one inverse operation and one pairing operation need to executed by an entity.

It may be mentioned that in a pairing based 2PAKA, either Tate pairing or Weil pairing is used to evaluate bilinear pairing operation $\hat{e} : G_1 \times G_1 \rightarrow G_2$, where G_1 is an additive group on elliptic curve E/F_p defined over F_p and its order is q, a 160-bit prime number and G_2, in order to provide an equivalent level of security to that of 1024-bit RSA with a 512-bit prime p [28], is a q-order (160 bit) subgroup of the multiplicative group of the finite field $F^*_{p^2}$. As an estimation of pairing cost, one bilinear pairing, according to [2,3], requires two to three times more multiplications than an elliptic curve scalar point multiplication, and also it has been seen from Cao et al.'s experimental result [8] that the execution of one pairing-based exponentiation is approximately equal to one-half of the time needed to execute one pairing operation.

In order to estimate the communication cost, we assume that the length of the identity is 16 bits and the output of the hash function (e.g., SHA-1) is 160 bits, and according to Cao et al.'s protocol [8], the longest message, which contains two points in elliptic curve group and one identity, requires $(16 + 2 \times 160)/8 = 42$ bytes bandwidth for communication. We define different time complexity notations and their conversions in terms of T_{ML} [2,3,8,15, 24,49,54,55] as shown in Table 2. The time complexity, number of rounds and bandwidth requirements of different relevant protocols have been calculated and shown in Table 3.

8 Conclusions

We proposed a 2PAKA protocol based on ECC and self-certified public keys of the participants, where the security of the session key lies on the Computational Diffie-Hellman assumption in the random oracle model. Furthermore, the formal security of the proposed

Table 2 Definition and conversion of various operation units

Notations	Definition and conversion
T_{ML}	Time complexity for executing the modular multiplication
T_{EX}	Time complexity for executing the modular exponentiation, $T_{EX} \approx 240T_{ML}$
T_{EM}	Time complexity for executing the elliptic curve scalar point multiplication, $T_{EM} \approx 29T_{ML}$
T_{BP}	Time complexity for executing the bilinear pairing operation, $T_{BP} \approx 3T_{EM} \approx 87T_{ML}$
T_{PX}	Time complexity for executing pairing-based exponentiation operation, $T_{PX} \approx 43.5T_{ML}$
T_{EA}	Time complexity for executing the addition of two elliptic curve points, $T_{EA} \approx 0.12T_{ML}$
T_{IN}	Time complexity for executing the modular inversion operation, $T_{IN} \approx 11.6T_{ML}$
T_H	Time complexity for executing the simple hash function, which is negligible

Table 3 Performance comparisons

Protocol	No. of rounds	Bandwidth (bytes)	Computation cost
Cao et al. [8]	2	$(16 + 2 \times 160)/8 = 42$	$10T_{EM} \approx 290T_{ML}$
Cao et al. [55]	3	$(16 + 2 \times 160)/8 = 42$	$10T_{EM} \approx 290T_{ML}$
Chen-Kudla [9]	2	$(16 + 512)/8 = 66$	$4T_{BP} + 4T_{EM} \approx 464T_{ML}$
Choie et al.-I [13]	2	$(16 + 512)/8 = 66$	$4T_{BP} + 6T_{EM} \approx 522T_{ML}$
Choie et al.-II [13]	2	$(16 + 512)/8 = 66$	$4T_{BP} + 8T_{EM} \approx 580T_{ML}$
Hölbl-Walzer-I [18]	2	$(16 + 2 \times 1024)/8 = 258$	$8T_{PX} \approx 348T_{ML}$
Hölbl-Walzer-II [18]	2	$(16 + 2 \times 1024)/8 = 258$	$6T_{PX} \approx 261T_{ML}$
Kudla-Paterson [21]	2	$(16 + 4 \times 1024)/8 = 514$	$6T_{PX} \approx 261T_{ML}$
McCullagh-Barreto [24]	2	$(16 + 512)/8 = 66$	$2T_{BP} + 4T_{EM} + 2T_{PX} \approx 377T_{ML}$
Smart [33]	2	$(16 + 2 \times 512)/8 = 132$	$2T_{BP} + 2T_{EM} + 2T_{PX} \approx 319T_{ML}$
Wang et al. [38]	2	$(16 + 512)/8 = 66$	$2T_{BP} + 4T_{EM} \approx 290T_{ML}$
Wang et al. [39]	2	$(16 + 2 \times 512)/8 = 132$	$2T_{BP} + 2T_{EM} + 2T_{EX} \approx 319T_{ML}$
Zhu et al. [43]	3	$(16 + 4 \times 160)/8 = 82$	$12T_{EM} \approx 348T_{ML}$
Zu-hua [56]	2	$(16 + 512)/8 = 66$	$4T_{EM} + 4T_{PX} + 6T_{BP} \approx 802T_{ML}$
Ni et al. [57]	2	$(16 + 512)/8 = 66$	$8T_{EM} + 2T_{IN} + 2T_{BP} \approx 429T_{ML}$
Wang et al. [58]	2	$(16 + 2 \times 512)/8 = 132$	$2T_{EM} + 6T_{BP} \approx 319T_{ML}$
Tsaur [59]	2	$(16 + 2 \times 160)/8 = 42$	$10T_{EM} \approx 290T_{ML}$
Proposed	2	$(16 + 2 \times 160)/8 = 42$	$6T_{EM} \approx 174T_{ML}$

scheme is validated through AVISPA software and the simulation results indicated that no active and passive attacks on the proposed scheme is possible. The detail informal security analysis of the proposed protocol against several known attacks is made and it has been found that all attacks are protected. Our protocol is also efficient in terms of bandwidth requirements and computation costs, which thus efficiently useable as an alternative scheme to the PKI- or identity-based 2PAKA protocol.

Acknowledgments The work is supported by the Outstanding Potential for Excellence in Research and Academics (OPERA) award, Birla Institute of Technology and Science (BITS) Pilani, Pilani Campus, Rajasthan, India.

References

1. Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, *22*, 644–654.
2. Barreto, P., Lynn, B., & Scott, M. (2004). On the selection of pairing-friendly groups. In *Proceedings of the selected areas in cryptography, LNCS*, Vol. 3006, Springer, 2004, pp. 17–25.
3. Barreto, P., Kim, H., Lynn, B., & Scott, M. (2002). Efficient algorithms for pairing-based cryptosystems. In *Proceedings of the 22nd annual international cryptology conference on advances in cryptology, LNCS*, Vol. 2442, Springer, pp. 354–368.
4. Blake-Wilson, S., Johnson, D., & Menezes, A. (1997). Key agreement protocols and their security analysis. In *Proceedings of the 6th IMA international conference on cryptography and coding, LNCS*, Vol. 1335, Springer, pp. 30–45.
5. Boneh, D., & Franklin, M. K. (2001). Identity-based encryption from the Weil pairing. In *Proceedings of the advances in cryptology, LNCS*, Vol. 2139, Springer, pp. 213–229.
6. Boyd, C., & Choo, K. K. R. (2005). Security of two-party identity-based key agreement. In *Proceedings of the progress in cryptology (MYCRYPT'05), LNCS*, Vol. 3715, Springer, pp. 229–243.
7. Canetti, R., & Krawczyk, H. (2001). Analysis of key exchange protocols and their use for building secure channels. In *Proceedings of the advances in cryptology (EUROCRYPT'01), LNCS*, Vol. 2045, Springer, pp. 453–474.
8. Cao, X., Kou, W., & Du, X. (2010). A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges. *Information Sciences*, *180*, 2895–2903.
9. Chen, L., & Kudla, C. (2002). Identity based key agreement protocols from pairings. In *Proceedings of the 16th IEEE computer security foundations workshop*, pp. 219–233.
10. Chen, T. H., Lee, W. B., & Chen, H. B. (2008). A round-and computation-efficient three-party authenticated key exchange protocol. *Journal of System and Software*, *81*(9), 1581–1590.
11. Chen, L., Cheng, Z., & Smart, N. P. (2007). Identity-based key agreement protocols from pairings. *International Journal of Information Security*, *6*, 213–241.
12. 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, 2005. Available at http://eprint.iacr.org/2005/129
13. Choie, Y., Jeong, E., & Lee, E. (2005). Efficient identity-based authenticated key agreement protocol from pairings. *Applied Mathematics and Computation*, *162*, 179–188.
14. Choo, K. K. R., Boyd, C., Hitchcock, Y., & Maitland, G. (2005). On session identifiers in provably secure protocols: The Bellare-Rogaway three-party key distribution protocol revisited. In *Proceedings of the information security and privacy, LNCS*, Vol. 3352, Springer, pp. 351–366.
15. Chung, Y. F., Huang, K. H., Lai, F., & Chen, T. S. (2005). ID-based digital signature scheme on the elliptic curve cryptosystem. *Computer Standards & Interfaces*, *29*(6), 601–604.
16. Girault, M. (1991). Self-certified public keys. In *Proceedings of the advances in cryptology (EUROCRYPT'91), LNCS*, Vol. 547, Springer, pp. 491–497.
17. Hankerson, D., Menezes, A., & Vanstone, S. (2004). *Guide to elliptic curve cryptography*. New York: Springer.
18. Hölbl, M., & Welzer, T. (2009). Two improved two-party identity-based authenticated key agreement protocols. *Computer Standards & Interfaces*, *31*, 1056–1060.
19. Hsieh, B. T., Sun, H. M., Hwang, T., & Lin, C. T. (2002) An improvement of Saeednia's identity based key exchange protocol. In *Proceedings of the information security conference*, pp. 41–43.
20. Koblitz, N. (1987). Elliptic curve cryptosystem. *Journal of Mathematics of Computation*, *48*, 203–209.
21. Kudla, C., & Paterson, K. G. (2005). Modular security proofs for key agreement protocols. In *Procdeedings of the advances in cryptolog (ASIACRYPT'05), LNCS*, Vol. 3788, Springer, pp. 549–565.
22. Li, S., Yuan, Q., & Li, J. (2005). Towards security two-part authenticated key agreement protocols. Cryptology ePrint Archive, Report, 2005/300, 2005. Available at http://eprint.iacr.org/2005/300.
23. Lu, R., & Cao, Z. (2007). Simple three-party key exchange protocol. *Computers & Security*, *26*(2007), 94–97.
24. McCullagh, N., & Barreto, P. S. L. M. (2005). A new two-party identity-based authenticated key agreement. In *Proceedings of the topics in cryptology (CT-RSA'05)*, pp. 262–274.
25. Miller, V. S. (1985). Use of elliptic curves in cryptography. In *Proceeding on advances in cryptology (CRYPTO'85), LNCS*, Vol. 218, Springer, pp. 417–426.
26. Phan, R. C. W., Yau, W. C., & Goi, B. M. (2008). Cryptanalysis of simple three-party key exchange protocol (S-3PAKE). *Information Science*, *178*, 2849–2856.

27. Pu, Q., Zhao, X., & Ding, J. (2009). Cryptanalysis of a three-party authenticated key exchange protocol using elliptic curve cryptography. In *Proceedings of the international conference on research challenges in computer science*, pp. 7–10.
28. Ren, K., Lou, W., Zeng, K., & Moran, P. J. (2007). On broadcast authentication in wireless sensor networks. *IEEE Transaction on Wireless Communication*, *6*(11), 4136–4144.
29. Ryu, E., Yoon, E., & Yoo, K. (2004). An efficient ID-based authenticated key agreement protocol from pairings. In *Proceedings of the networking technologies, services, and protocols; performance of computer and communication networks; mobile and wireless communications (NETWORKING'04), LNCS*, Vol. 3042, pp. 1458–1463.
30. Saeednia, S. (2000). Improvement of Gunther's identity-based key exchange protocol. *Electronics Letters*, *36*(18), 1535–1536.
31. Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In *Proceedings of the advances in cryptology (CRYPTO'84), LNCS*, Vol. 196
32. Shim, K. (2003). Efficient ID-based authenticated key agreement protocol based on Weil pairing. *Electronics Letters*, *39*(8), 653–654.
33. Smart, N. P. (2002). An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters*, *38*, 630–632.
34. Sun, H., & Hsieh, B. (2003). Security analysis of Shim's authenticated key agreement protocols from pairings, Cryptology ePrint Archive 2003/113. Available at http://eprint.iacr.org/2003/113/
35. Tan, Z. (2010). An enhanced three-party authentication key exchange protocol for mobile commerce environments. *Journal of Communications*, *5*(5), 436–443.
36. Tseng, Y. M. (2007). An efficient two-party identity-based key exchange protocol. *Informatica*, *18*(1), 125–136.
37. Tseng, Y. M., Jan, J. K., & Wang, C. H. (2002). Cryptanalysis and improvement of an identity based key exchange protocol. *Journal of Computers*, *14*(3), 7–22.
38. Wang, S., Cao, Z., Choo, K. K. R., & Wang, L. (2009). An improved identity-based key agreement protocol and its security proof. *Information Sciences*, *179*, 307–318.
39. Wang, S., Cao, Z., Cheng, C., & Choo, K. K. R. (2009). Perfect forward secure identity-based authenticated key agreement protocol in the escrow mode. *Science in China series F: Information sciences*, *52*(8), 1358–1370.
40. Xie, G. (2004). Cryptanalysis of Noel McCullagh and Paulo S.L.M. Barreto's two-party identity-based key agreement, Cryptology ePrint Archive, Report 2004/ 308, 2004. Available at http://eprint.iacr.org/2004/308.
41. Yang, J. H., & Chang, C. C. (2009). An efficient three-party authenticated key exchange protocol using elliptic curve cryptography for mobile-commerce environments. *Journal of system and Software*, *82*(9), 1497–1502.
42. Zhang, S., Cheng, Q., & Wang, S. (2010) Impersonation attack on two identity-based authenticated key exchange protocols. In *Proceedings of the WASE international conference on information engineering*, pp. 113–116.
43. Zhu, R. W., Yang, G., & Wong, D. S. (2007). *Theoretical Computer Science*, *9*(378), 198–207.
44. Das, A. K. (2012). A secure and effective user authentication and privacy preserving protocol with smart cards for wireless communications. *Networking Science*,. doi:10.1007/s13119-012-0009-8.
45. Das, A. K., Massand, A., & Patil, S. (2013). A novel proxy signature scheme based on user hierarchical access control policy. *Journal of King Saud University-Computer and Information Sciences*,. doi:10.1016/j.jksuci.2012.12.001.
46. Farash, M. S., Attari, M. A., Atani, R. E., & Jami, M. (2012). A new efficient authenticated multiple-key exchange protocol from bilinear pairings. *Computers Electrical Engineering*, *39*(2), 530–541.
47. Basu, A., Sengupta, I., & Sing, J. K. (2012). Formal security verification of secured ECC based signcryption scheme. In *Proceedings of the advances in computer science, engineering & applications, LNCS*, Vol. 167, Springer, pp 713–725.
48. Islam, S. H., & Biswas, G. P. (2013). An efficient and secure strong designated verifier signature signature scheme without pairings. *Journal of Applied Mathematics & Informatics*, *31*(3), 425–441.
49. Islam, S. H., & Biswas, G. P. (2013). A provably secure identity-based strong designated verifier proxy signature scheme from bilinear pairings. *Journal of King Saud University-Computer and Information Sciences*,. doi:10.1016/j.jksuci.2013.03.004.
50. AVISPA Web tool: Automated validation of internet security protocols and applications. www.avispa-project.org/web-interface/. Accessed on Jan 2013.
51. AVISPA: The AVISPA user manual (2005). http://www.avispa-project.org/publications.html
52. Dolev, D., & Yao, A. C. (1983). On the security of public-key protocols. *IEEE Transactions on Information Theory*, *2*(29), 198–208.

53. Islam, S. H., & Biswas, G. P. (2013). Provably secure and pairing-free certificateless digital signature scheme using elliptic curve cryptography. *International Journal of Computer Mathematics*,. doi:10.1080/00207160.2013.776674.
54. 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.
55. Cao, X., Kou, W., Yu, Y., & Sun, R. (2008). Identity-based authentication key agreement protocols without bilinear pairings. *IEICE Transaction on Fundamentals.*, *E91–A*(12), 3833–3836.
56. Zu-hua, S. (2005). Efficient authenticated key agreement protocol using self-certified public keys from pairings. *Wuhan University Journal of Natural Sciences*, *10*(1), 267–270.
57. Ni, L., Chen, G. L., Li, J. H., & Hao, Y. Y. (2013). Strongly secure identity-based authenticated key agreement protocols in the escrow mode. *Science China Information Sciences*, *56*(8), 1–14.
58. Wang, S., Cao, Z., & Cao, F. (2008). Efficient identity-based authenticated key agreement protocol with PKG forward secrecy. *International Journal of Network Security*, *7*(2), 181–186.
59. Tsaur, W. J. (2005). Several security schemes constructed using ECC-based self-certified public key cryptosystems. *Applied Mathematics and Computation*, *168*, 447–464.

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. His research interest includes Cryptography and Information Security.

G. P. Biswas received B.Sc (Engg.) and M.Sc (Engg.) degrees in Electrical & Electronics Engineering and Computer Science & Engineering, respectively. He completed his Ph.D degree in Computer Science & Engineering from Indian Institute of Technology, Kharagpur, India. He is currently working as a Professor in the Department of Computer Science & Engineering, Indian school of Mines, Dhanbad, Jharkhand, India. He has around 20 years of teaching and research experiences, and published more than 100 research articles in Journals, Conferences and Seminar Proceedings. His main research interests include Cryptography, Computer Network and Security, Cellular Automata, VLSI Design.