Modelling Identity-Based Authentication and Key Exchange Protocol Using the Tamarin Prover

Srijanee Mookherji, Vanga Odelu, Rajendra Prasath, Alavalapati Goutham Reddy, and Basker Palaniswamy

Abstract In real-time applications, authentication plays a vital role in enabling secure communications. The authentication protocols need to be formally verified under a defined threat model. Unless the protocols are verified for the intended security, the purpose of employing such protocols may eventually fail. There are multiple ways to formally verify the security of the authentication protocols including the use of automatic verification tools like the Tamarin Prover. The Tamarin Prover tool supports equational theories along with built-in functions. However, this tool does not support some mathematical operations such as elliptic curve point addition. It is necessary to have point addition in Identity-Based Encryption (IBE)-based authentication protocols. Chen–Kudla modelled the point addition operation in the Tamarin Prover using a technique based on concatenation. However, this technique is not applicable to all identity-based protocols including IBE-based authentication protocols. In this paper, we present a modelling technique known as normalised precomputation for point addition using a hash function. We analyse the security of a simple identity-based encryption-based key exchange protocol under extended Canetti and Krawczyk's (eCK) adversary model. Our analysis shows that the proposed technique is secure and retains the properties of point addition. Therefore, the technique can be applied to different IBE-based authentication protocols where point addition operation is necessary.

S. Mookherji (✉) · V. Odelu · R. Prasath
Computer Science and Engineering Group, Indian Institute of Information Technology Sri City, Chittoor, 630 Gnan Marg, Sri City 517646, Andhra Pradesh, India
e-mail: srijanee.mookherji@iiits.in

V. Odelu
e-mail: odelu.vanga@iiits.in

R. Prasath
e-mail: rajendra.prasath@iiits.in

A. G. Reddy
Department of Mathematics and Computer Science, Fontbonne University, St. Louis, MO 63105, USA

B. Palaniswamy
VIT-AP University, Amravati 522237, Andhra Pradesh, India

S. J. Patel et al. (eds.), *Information Security, Privacy and Digital Forensics*,
Lecture Notes in Electrical Engineering 1075,
https://doi.org/10.1007/978-981-99-5091-1_9

Keywords The Tamarin Prover · Authentication protocol · Key exchange · eCK-Adversary model · Elliptic curve point addition

1 Introduction

A secure communication is a problem that the cryptography research community has been working on for a long time. Authentication and Key Exchange (AKE) is an essential part of a secure communication. The problem dealt with in an authenticated communication is that of an adversary $\mathcal{A}$ which has the power to modify, delete, delay and introduce false messages or impersonate a participant in the communication. Key exchange in an authenticated communication allows two parties to generate a shared secret. Authentication protocols use various key exchange techniques like Diffie–Hellman Key Exchange (DHKE) protocol [5] to establish a secret session key [6, 17]. When it comes to a multi-server environment, such authentication protocols may have a major limitations such as the clients may need to store public keys of every single server [31]. To overcome the limitations, identity-based key exchange protocols were introduced [4, 15, 22]. The idea has been applied to design many key exchange protocols [3, 18]. In an identity-based cryptosystem, user identities are used as public keys. A trusted third-party generates a private key for the user using the user identity and a master key. The public key is the user identity, thus users do not need to store multiple public keys.

The extended Canetti–Krawczyk (eCK) [10] adversary model is a widely accepted adversary model. It is used to verify the various required security properties for AKE protocols. A protocol is considered as secure, if an adversary $\mathcal{A}$, who is in control of communication between two parties, is unable to distinguish session key from a random value. It can do so, only if it calls certain queries that reveal various secret information that are part of the protocol communication. In the eCK adversary model, the adversary is able to call *Ephemeral Key Reveal Query*, *Long-Term Key Reveal Query* and *Session Key Reveal Query*. The *Ephemeral Key Reveal Query* allows an $\mathcal{A}$ to capture all the session-specific temporary secret information. The *Long-Term Key Reveal Query* reveals the long-term secret keys of a party to the adversary and *The Session Key Reveal Query* reveals the current session key between two parties. However, the adversary is allowed to call the queries one at a time.

The Tamarin Prover is an automatic formal security analysis tools which supports features like Diffie–Hellman, hashing, bilinear pairing and so on. The shortfall of the tool is that it does not support elliptic curve point addition [21]. The developers provide a modelling example for the Chen–Kudla protocol [3] where they use ordered concatenation in place of point addition. However, the same approach cannot be implemented for all AKE protocols using point addition operations. We introduce a generalised ID-based Authentication and Key Exchange (ID-AKE) protocol in this paper that uses point addition operations. We model the same in the Tamarin Prover tool using a different modelling technique and analyse it under the eCK adversary model.

In the upcoming sections, we define the required mathematical preliminaries in Sect. 2. Next, we describe the literature review on modelling AKE protocols using the Tamarin Prover in Sect. 3. We discuss the problem of replacing point addition operation with an ordered concatenation in Sect. 4. The contributions of the paper are presented in Sect. 5. In Sect. 6, the summary of a generalised ID-AKE protocol and its Tamarin Prover model is given. We demonstrate that the proposed modelling technique ensures that ID-AKE protocol is secure under the eCK adversary model and it retains the properties of point addition. Finally, Sect. 8 concludes the paper.

2 Mathematical Background

In this section, we discuss the required mathematical preliminaries used to design the ID-AKE protocol.

2.1 Bilinear Pairings

Bilinear pairings can be defined by assuming that G_1 is an additive cyclic group of prime order q, G_2 is a multiplicative cyclic group of prime order q. Let, P be the generator of G_1, the bilinear pairing equation $e : G_1 \times G_1 \rightarrow G_2$ satisfies the following properties [29]:

- Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$, for all $P, Q \in G_1$ and for all $a, b \in Z_q^*$.
- Computability: For all $P, Q \in G_1$, $e(P, Q)$ can be efficiently computed.
- Non-degeneracy: There exists $P, Q \in G_1$ with $e(P, Q) \neq 1$, where 1 is the multiplicative identity of G_2.

2.2 Hash Function

A one-way hash function is a function $h : \{0, 1\}^* \rightarrow \{0, 1\}^n$ satisfying the following conditions [13, 16]:

- The input $x\{0, 1\}^*$ is of arbitrary length binary string and the output $h(x) \in \{0, 1\}^n$ is a binary string of fixed length with n bits.
- **One-wayness**: Given a $y = h(x) \in \{0, 1\}^n$, it is hard to compute x in $\{0, 1\}^*$.
- **Collision-Resistant**: Given $x \in \{0, 1\}^*$, finding $y \in \{0, 1\}^*$ where $x \neq x$ such that $h(x) = h(y)$ is infeasible to compute.

2.3 Message Authentication Code

Let $M \in \{0,1\}^*$ be a message of variable length, $K \in \mathcal{K}$, where $\mathcal{K}$ is the key space, be a secret key shared between two parties. We define a message authentication code, say, $MAC : \mathcal{K} \times \{0,1\}^* \rightarrow \{0,1\}^n$, and the function $C = MAC(K, M)$, where $C \in \{0,1\}^n$ is a fixed length binary string. The MAC satisfies the following properties [24]:

- Without key K, it is hard to verify the message authentication code M.
- For a given C, it is hard to compute the M and K due to one-wayness of the MAC.

3 Related Works

Many AKE protocols have been designed till now using the concepts of the DHKE protocol [1, 7]. Shamir [22] introduced the identity-based key exchange protocol to overcome the problem of storing multiple public keys. Using the same concept, many ID-AKE protocols were proposed [3, 9, 14, 30]. The identity of the parties was used as their public keys. The DHKE protocol is a simple and essential protocol that is still being used widely to design AKE protocols. To study the modelling techniques used in the Tamarin Prover, studying the DHKE protocol model is of utmost importance. The DHKE protocol model is coded in [12] and its vulnerability is tested against the Man-In-The-Middle (MITM) attack under the eCK adversary model. In the MITM attack, an adversary $\mathcal{A}$ is able to impersonate one or both the participants without their knowledge and finally establish a shared secret key with the participants [8].

In various Tamarin Prover documentations [11, 20, 21], the authors have described about a few AKE protocols like Joux protocol [9], Chen–Kudla protocol [3], RYY protocol [19], etc. that use point addition. The protocols are modelled using different modelling techniques and the codes are present in the 'The Tamarin Prover Github Repository'[25]. For formalising the Joux protocol, they used the multiset of identities of the participants. A study of the technique is presented in Sect. 3.2. Next, as stated in [21], Chen–Kudla KA protocol is modelled using an ordered concatenation instead of point addition. The Chen–Kudla protocol's modelling technique is thoroughly explained in Sect. 3.3. An exhaustive study of three protocols [3, 5, 9] is presented in order to look through the different modelling techniques incorporated to formalise them. Also, the paper focusses on their potential of being used in a generalised case. A comparative study of the protocols is presented in Table 1. The study summarises the protocols based on the cryptographic primitives used, the protocol specifications, the modelling techniques and various security properties used for verification.

```
lemma MITM:
"All #i1 skey. (Session_created(skey)@i1)
==> (not(Ex #i2. K(skey)@i2))"
```

Fig. 1 A simple MITM lemma

3.1 Diffie–Hellman Key Exchange Protocol

The DHKE protocol is reviewed and the Tamarin Prover model is presented in this section. We consider two parties: *Alice* and *Bob*. Alice and Bob choose a secret $a \in (1 \leq a \leq p-1)$, $b \in (1 \leq b \leq p-1)$, respectively, over a finite field $GF(p)$ with prime p. $A = g^a \mod p$ and $B = g^b \mod p$ are computed by Alice and Bob, respectively. A is sent to Bob by Alice and B is sent to Alice by Bob. They compute the shared session key $SessK = SessKA = SesskB = g^{ab} \mod p$.

Man-In-The-Middle Attack in the DHKE Protocol: The DHKE protocol is vulnerable to the MITM attack [8]. Let us assume that an adversary $\mathcal{A}$ intercepts the message $M1 = \langle A \rangle$ from Alice and replaces it with $X = g^x$ finally sending $M2 = \langle X \rangle$ to Bob. Bob sends $M3 = \langle B \rangle$ which is again intercepted by the $\mathcal{A}$ and passed onto Alice without any changes. At the end, $\mathcal{A}$ establishes a session with Bob using $SessK = g^{xb}$, and therefore impersonating Alice. The DHKE protocol is modelled and tested using the Tamarin Prover in [12]. The model is tested for MITM vulnerability using the simple Lemma 1 which is shown and elaborated in Fig. 1.

Lemma 1 *For all cases, session keys that are created at an instance* $i1$*, the adversary,* $\mathcal{K}$ *must not be able to compute a session key at an instance* $i2$*.*

The tool produced an analysis stating that the protocol is vulnerable to the MITM attack. The Tamarin Prover tool traces for the possibility of the MITM attack and is able to find a counterexample where an adversary, $\mathcal{K}$, is able to compute the session key, thus turning the lemma red. Here, the adversary sends a fake g^a. Therefore, we can conclude that an adversary is able to perform an MITM attack.

3.2 The Joux Protocol Using Signatures (SIGJOUX)

In this section, we review the three-party authentication protocol proposed by Joux [9], which is a variation of the Diffie–Hellman protocol. This uses bilinear pairing.

- Three parties: *Alice*, *Bob* and *Carol* participate in one round tripartite DHKE.
- Each of them select random values a, b and c, respectively. They also choose long-term keys ka, kb, kc, respectively, $\ni 1 \leq (a, b, c) \leq p-1$ over the finite field GF(p) with prime p. Finally, they compute $A = aP$, $B = bP$ and $C = cP$. Here, $e : G_1 \times G_1 \rightarrow G_2 \ni e(xP, yP) = e(P, P)_{xy}$.

– Alice, Bob and Carol simultaneously sign a message with their chosen long-term keys and send the following to two other parties : $SigA : sign_{ka}(ID_B, ID_C, A)$, $SigB : sign_{kb}(ID_A, ID_C, B)$ and $SigC : sign_{kc}(ID_A, ID_B, C)$.
– On receiving the same, each party is able to compute their own shared secret keys, SessKA: $h(e(B, C)^a, ID_A, ID_B, ID_C)$, SessKB: $h(e(A, C)^b, ID_A, ID_B, ID_C)$ and SessKC: $h(e(A, B)^c, ID_A, ID_B, ID_C)$.
– Finally, the shared secret is $SessK = SessKA = SessKB = SessKC = e(P, P)^{abc}$ where $e(P, P)^{abc} \in G_2$.

The entire code $Joux.spthy$ is present in the Tamarin Prover Github repository [27]. The ephemeral key, ekA, is denoted with $\sim ekA$ which denotes that it is a fresh value and the rule Proto1 will generate a fresh ekA for every session. The !Ltk() ensures that the fact will remain constant at all times. While modelling the Session Key generation we can see that, for each party, the IDs are denoted by $ which means that they are public. For each party, the IDs of the other two are added using the multiset operator '+' as per the multiset rewriting rules for formalising a protocol.

The Tamarin Prover modelling formalises the protocol by using multiset rewriting rules. Alice, A, chooses her ephemeral key ekA. The other two parties are Bob, B, and Carol, C. The signing key $ltkA$ is used to sign his own public identity $\$A$, the multiset of the public identities of the two parties $\$B, \$C$ along with the public ephemeral key $[ekA]P$. Pstate$(x, A, B + C)$, which is the protocol state fact, denotes that a session is executed. In the second rule, A checks the signatures of the other two parties, extracts their XB and XC which are the public ephemeral keys of B and C, respectively, and computes the shared key as $e(XB, XC)^{ekA}$. The protocol succeeds in providing Perfect Forward Secrecy with *Long-Term Key Reveal* model if A accepts the session key generated with B and C. However, it fails to provide the same if there is an *Ephemeral Key Reveal* modelled for the protocol.

3.3 Chen–Kudla Key Agreement Protocol

We study the Chen–Kudla Key Agreement Protocol in this section. It is an ID-based key exchange protocol that uses the concepts of bilinear pairing and point addition. A Key Generation Centre (KGC) is there that is responsible for the registration of users $\mathcal{U}$. The key exchange protocol is a two-party communication.

– In the KGC setup phase, the KGC randomly selects a secret key, Key, which acts as the long-term key and computes public key $Pub = KeyP \ni P \in G_1$. Here, P is a generator of G_1 and $Key \in Z_q^*$.
– In the key exchange phase, Alice (A) and Bob (B) are considered as two users. KGC computes $H_A = h(ID_A)$, $S_A = KeyH_A$ for Alice, $H_B = h(ID_B)$, $S_B = KeyH_B$ for Bob. KGC sends S_A and S_B to Alice and Bob, respectively. Here $H_A, H_B \in G_1$. $h()$ is the hash function $\ni h\{0, 1\}^* \rightarrow G_1$.

– A computes $A = aP$ and B computes $B = bP$, where a and b are randomly selected ephemeral secrets. A is sent to Bob by Alice. B is sent to Alice by Bob.
– Alice then generates $SessKA = e(S_A, B)e(aH_B, Pub)$ and Bob generates $SessKB = e(S_B, A)e(bH_A, Pub)$ which results in the computation of $SessKey = SessKA = SessKB = e(bH_A + aH_B, Pub)$.

The Chen–Kudla protocol is modelled in the Tamarin Prover by replacing the point addition operation with an ordered concatenation. The complete code *Chen_Kudla.spthy* is available in the Tamarin Prover Github Repository [26]. The shared secret key $sessK = e(ex[hp(\$B)] + ey[hp(\$A)], P_s)$ is written as $(e(hp(\$B), mpk)^{ex})$ $(e(skA, Y))$ using the concepts of bilinearity [2] that states that $e(P + Q, Y) = e(PY)e(QY)$. The protocol model works aptly when the adversary $\mathcal{A}$ is restricted from revealing the ephemeral key of the test session and its significant matching session. This is true even if no *Long-Term Key Reveal Query* is called by $\mathcal{A}$. On removing the *Ephemeral Key Reveal Query* restriction, the protocol fails to provide key secrecy.

Comparative Study: A comparative study of the protocols is presented in Table 1. The protocols [3, 5, 9] and the generalised ID-AKE protocol are compared with respect to the cryptographic primitives used to design the protocols, the Tamarin Prover modelling technique used and the various security properties achieved by the protocols.

Table 1 Comparative study of modelling protocols in the Tamarin Prover

Features / Protocol	DHKE [5]	Joux [9, 20]	Chen–Kudla KE [3]	Proposed ID-AKE
Cryptographic primitives	Finite field [GF(p)]	Bilinear pairing	Bilinear pairing, ECC point addition	Bilinear pairing
Protocol specifications	Not applicable	ID based, signature	ID based	ID based
Modelling technique	Simple	ID of other two parties as multiset (+ operator)	Bilinear terms concatenated instead of point addition operator	Pre-computed keys
MITM	Not secured	Secured	Secured	Secured
Perfect forward secrecy	Not applicable	Secured	Not secured	Secured
Ephemeral key secrecy	Not applicable	Not secured	Not secured	Secured

4 Problem with Ordered Concatenation in ID-AKE

The Tamarin Prover does not provide the provision of performing point addition. Also, it does not support computation of equalities such as $(c)[(a)P + (b)]P = [(ca)P + (cb)P]$ [21]. Here, for example, the Tamarin Prover model for Chen–Kudla Key Agreement Protocol (*Chen_Kudla.spthy*) present in the repository [26], bilinear terms having point addition are replaced with an ordered concatenation [21] as discussed in Sect. 3.3. There are many ID-AKE protocols that are designed using the point addition operation [14, 23, 28]. The same approach cannot be used in such cases where point addition is used to secure the master key of a Trusted Key Distribution Centre (TKGC). The point addition operation is used to generate the authentication message of the participants in the communication by using the public key of the TKGC. Using the concept of concatenation would not help in achieving security of the master key. This is because, for performing the concatenation operation, the master key needs to be used directly.

We present a generalised ID-based authentication and key exchange (ID-AKE) protocol and model it using the Tamarin Prover in this paper. The detailed description is presented in Sect. 6. The protocol uses the concept of bilinear pairing and point addition. Subsequently, to model the generalised ID-AKE protocol, we embrace the technique of normalisation and define a unary public function $hf/1$ that works similarly to a hash function in Tamarin Prover. Along with this, some precomputations need to be performed in order to ensure the KGC's secret key security. The technique is illustrated in detail in Sect. 6.2.

5 Contributions of the Paper

The research contributions of the paper are as follows:

- We present a comparative study of the Tamarin Prover modelling techniques used to model authentication protocols that use point addition operations.
- We discuss a generalised ID-AKE protocol that uses point addition operation and present a technique using normalisation and precomputation to model the same.
- Under the eCK adversary model, we test the security properties using the proposed technique. The result shows that the proposed technique is able to achieve the properties of point addition without compromising security of the original protocol.

6 A Generalised ID-AKE Protocol

We provide a summary of the generalised ID-AKE protocol in this section. We discuss about the modelling strategy used and the normalisation and precomputations needed to successfully model the protocol.

Table 2 ID-AKE—registration phase

KGC	
Chooses a Secret Key, Key Computes Public Key, $Pub = Key \cdot P$	
KGC	User
Compute: $K_U = \frac{1}{h(ID_U)+Key} P$ $\xrightarrow{\langle ID_U, K_U \rangle}$	

6.1 Authentication and Key Exchange Protocol

We begin with the Key Generation Centre setup phase as shown in Table 2. The Key Generation Centre (KGC) chooses a master private key, Key and generates public key $Pub = KeyP$.

A user requests for registration in the user registration phase (Table 2). The KGC computes $K_U = \frac{1}{h(ID_U)+Key} P$ and sends $\langle ID_U, K_U \rangle$, which the user keeps safe. In the proposed protocol, we assume that two users Alice, A and Bob, B register with the KGC. The KGC sends $\langle ID_A, K_A \rangle$ to Alice and $\langle ID_B, K_B \rangle$ to Bob. Here, $K_A = \frac{1}{h(ID_A)+Key} P$ and $K_B = \frac{1}{h(ID_B)+Key} P$.

In the authentication and key exchange phase (Table 3), Alice chooses secret a and computes $A = a(h(ID_B)P + Pub)$. Alice then sends $\langle M1 = A \rangle$ to Bob. Similarly, Bob chooses secret b, computes $B = b(h(ID_A)P + Pub)$ and $SessKB = e(A, K_B)^y$. Bob then sends $\langle M2 = MAC(SessKB, A, B) \rangle$ to Alice. Thus, Alice authenticates Bob.

Alice further computes $SessKA = e(B, K_A)^x$ and sends $\langle M3 = MAC(SessKA, B, A) \rangle$ back to Bob. Hence, authenticating herself to Bob and establishing a secret session key $SessK = SessKA = SessKB = e(P, P)^{ab}$.

6.2 Modelling ID-AKE Using the Tamarin Prover

In this section, we describe about the normalisation and precomputations that are required in our modelling technique. We discuss the Tamarin Prover model and verify the protocol security under the eCK adversary model using the Tamarin Prover.

Normalisation and precomputation: In the designed ID-AKE, to model the point addition operation, normalisation needs to be performed. The public key for the participants needs to be pre-computed by the KGC. The importance of point addition in this protocol is that the operation is used to construct the ID-based public key without revealing the private key of the KGC. This is achieved by point adding the public key of the KGC. The point addition operation provides the required hardness to secure the private key.

For the normalisation of the initial point addition operation, we introduce a public unary function $hf/1$ that is against multiple inputs the function provides a single

Table 3 Alice and Bob authentication and key exchange phase

Alice	Bob
Choose secret a	
$A = a(h(ID_B)P + Pub)$	
$\xrightarrow{M_1=\langle ID_A, A\rangle}$	
	Choose secret b
	$B = b(h(ID_A)P + Pub)$
	Compute : $SessKB = e(A, K_B)^b$
	$Auth = MAC(SessKB, ID_A, ID_B, A, B)$
	$\xleftarrow{M2=\langle Auth, B\rangle}$
Compute : $SessKA = e(B, K_A)^a$	
$Conf = MAC(SessKA, B, A, ID_B, ID_A)$	
Check: $Auth \stackrel{?}{=} Conf$	
$\xrightarrow{M3=\langle Conf\rangle}$	
	Check : $Conf \stackrel{?}{=} Auth$
Shared Secret: $SessK = SessKA = SessKB = e(P, P)^{ab}$	

output. We use inv denoting field inverse and $pmult$ denoting point multiplication. The normalisation in the protocol is performed as follows:

- For $K_A = \frac{1}{h(ID_A)+Key}P$ the point addition part is normalised as $TempKa = hf(ID_A, Key)$. Next, K_A is computed as $K_A = pmult(inv(hf(ID_A, Key)),' P')$.
- For $K_B = \frac{1}{h(ID_B)+Key}P$ the point addition part is normalised as $TempKb = hf(ID_B, Key)$. Next, K_B is computed as $K_B = pmult(inv(hf(ID_B, Key)),' P')$.

In the AKE phase, $A = a(h(ID_B)P + Q)$ and $B = b(h(ID_A)P + Q)$ are the ephemeral public key that needs to be computed at Alice and Bob's end, respectively. In order to use the normalisations $TempKa$ and $TempKb$ for computation of A and B, Alice and Bob need to have the knowledge of 'Key' which is the long-term key of KGC. It is highly undesirable from protocol security point of view. Thus, we precompute the values $ap = pmult(TempKa,' P')$ and $bp = pmult(TempKb,' P')$ at the KGC's end and send it to Alice and Bob as public keys. With ap Bob computes $B = pmult(\ b, ap)$ and with bp Alice computes $A = pmult(\ a, bp)$ which are the ephemeral public key used for authentication.

The Tamarin Prover Code: The Tamarin Prover model for the generalised ID-AKE is explained below: The program *IDAKE.spthy* starts with the header *'theory IDbasedAKE'* which is the theory name. The next line has 'begin' which means start of the protocol modelling. The third line calls the bulitins that are required for the modelling. The fourth line describes the public functions hf/1 and mac/2 used to model the protocol. The code is shown in Fig. 2.

The rule 'TrustedKGC' is depicted in Fig. 3. It is defined to compute the public key and long-term key (ltk) of the KGC (key generation centre). For every session,

```
theory IDbasedAKE
begin
builtins: diffie-hellman, bilinear-pairing, hashing
functions: hf/1, mac/2
```

Fig. 2 ID-AKE—Tamarin Prover model—the Tamarin Prover code header

```
rule TrustedKgc:
let Pub = pmult(~ Key,'P')
in [Fr(~ Key)]--[]->[!Ltk($TKGC,~ Key),!PubK($TKGC,Pub)]
```

Fig. 3 ID-AKE—Tamarin Prover model—rule for KGC setup

```
rule AliceReg:
let TempKa = hf(h($IDA),Key)
 ap = pmult(TempKa,'P')
 Ka = pmult(inv(TempKa),'P')
in [!Ltk($TKGC,Key)]--[]->[!PubA($IDA,ap),!Ltk($IDA,Ka)]

rule BobReg:
let TempKb = hf(h($IDB),Key)
 bp = pmult(TempKb,'P')
 Kb = pmult(inv(TempKb),'P')
in [!Ltk($TKGC,Key)]--[]->[!PubB($IDB,bp),!Ltk($IDB,Kb)]
```

Fig. 4 ID-AKE—Tamarin Prover model—rule for user registration

the rule will generate a fresh persistent long-term key ~Key as registered with !Ltk() which acts as the master key. Persistent fact !PubK() is used to compute the public key.

The code presented in Fig. 4 shows the Rules 'AliceReg' and 'BobReg' which are used to model the long-term key generation for Alice and Bob using the master key of the KGC. According to the protocol, the key of the user is computed by using the concept of normalisation. Point addition is replaced by the singular public function hf/1 and TempKa and TempKb is computed accordingly. With the normalised value, the long-term key for Alice and Bob is computed and registered using !Ltk(). For the precomputation, the public key is registered using !PubA() and !PubB() which contains the normalised value.

Rules 'Alice' and 'Bob' present the computation of values of A and B. The computations are done using the Ephemeral keys a and b. The code is presented in Fig. 5. A and B (as per the protocol) are computed using the KGC's public key. Thus, the long-term key of the KGC remains a secret. For modelling the same the normalised pre-computed values ap and bp have been used and ephemeral keys ~a and ~b have been registered using !Ephk().

Rules 'AliceSession' and 'BobSession' as shown in Fig. 6 are used to generate the session keys sesska and sesskb using the persistent fact !SessionKey(), MAC(),

```
rule Alice:
let A = pmult(~ a,bp)
in [!PubB($IDB,bp),Fr(~ a)]--[]->[Alice(A),!Ephk( ~ a,~ a )]

rule Bob:
let B = pmult(~ b,ap)
in [!PubA($IDA,ap),Fr(~ b)]--[]->[Bob(B), !Ephk( ~ b,~ b )]
```

Fig. 5 ID-AKE—Tamarin Prover model—rule for generation of ephemeral public key

```
rule AliceSession:
let bilp = em(B,Ka)
 sesska = bilp ^ ~ a
 macmsgalice = mac(sesska,(<B,A,$IDA>))
in [Alice(~ a),!Ltk($IDA,Ka), In(<B, macmsgbob>),Alice(A)]
 --[Session_created_A(sesska), Accept( ~ a, $IDA, $IDB, sesska )
 , SessionID( ~ a, <'Alice',$IDA,$IDB,A,B> )
 , MatchingSession(~ a, <'Bob',$IDB,$IDA,A,B>)
 , Eq(macmsgalice,macmsgbob)]->[!SessionKey(sesska),Out(<A,macmsgalice>)]

rule BobSession:
let bilp = em(A,Kb)
 sesskb = bilp^~ b
 macmsgbob = mac(sesskb,(<$IDA,A,B>))
in [Bob(~ b),!Ltk($IDB,Kb),Bob(B),In(<A,macmsgalice>)]
 --[Session_created_B(sesskb), Accept( ~ b, $IDB, $IDA, sesskb )
 , SessionID( ~ b, <'Bob',$IDB,$IDA,A,B> )
 , MatchingSession( ~ b, <'Alice',$IDA,$IDB,A,B> )
 , Eq(macmsgalice,macmsgbob)]->[!SessionKey(sesskb),Out(<B, macmsgbob>)]
```

Fig. 6 ID-AKE—Tamarin Prover model—rule for session key generation

which is used to authenticate each other is also computed. An equality check is done for the MAC() values that are exchanged using the equality restrictions (presented in Fig. 7). A SessionID() and a MatchingSession() is associated for every session created by the above rules. Session_Created() denotes that the rule ran and a session is created and Accept() fact states that the shared session key sesska and sesskb has been accepted [11].

Security Properties: To analyse the modelled protocol under the eCK adversary model, we design Lemmas 10 and 3. The lemma codes are presented in Figs. 8 and 9.

Lemma 2 *MITM : The lemma states that for all sessions created and the adversary, $\mathcal{K}$, has not called the Long-Term Key Reveal Query or the Session Key Reveal Query, it implies that $\mathcal{K}$ is not able to compute the shared secret session key.*

Lemma 3 *Session Key Secrecy: The lemma states that there does not exist an accepted test session and the adversary, $\mathcal{K}$, does not have the shared session key. Also, the adversary has not called a Session Reveal Query. If the adversary has found a matching session it implies that the following queries have not been called:*

```
/*Restrictions*/
restriction Equality:
"All x y #i. Eq(x,y) @#i ==> x = y"

/* Key Reveals */
rule ltk_reveal:
 [ !Ltk($TKGC, ~ Key) ]--[ LtkReveal($TKGC) ]-> [ Out(~ Key) ]
rule Sessionk_reveal:
 [ !SessionKey(skey) ] --[ SesskeyReveal(skey)]-> [ Out(skey)]
rule Ephk_reveal:
 [ !Ephk(~ s, ~ ek) ]--[ EphkeyReveal(~ s) ]-> [ Out(~ ek) ]
```

Fig. 7 ID-AKE—Tamarin Prover model—restrictions and key reveal models

```
lemma MITM:
"(All #i1 #i2 skey .
(Session_created_A(skey) @ i1 & Session_created_B(skey)
@i2 & not ( (Ex A #ia . LtkReveal( A ) @ ia )
| (Ex B #ib . SesskeyReveal( B ) @ ib )))
==> not (Ex #i3. K( skey ) @ i3 ))"
```

Fig. 8 ID-AKE—Tamarin Prover model—MITM lemma

```
lemma key_secrecy:
"not (Ex #i1 #i2 s A B k . Accept(s, A, B, k) @ i1 & K( k ) @ i2
& not(Ex #i4. SesskeyReveal(s) @ i4 ) & (All ss #i4 #i5 ms.
(SessionID (ss, ms) @ i4 & MatchingSession(s, ms) @ i5)
==> (not(Ex #i6. SesskeyReveal(ss) @ i6)
& not(Ex #i6 #i7. LtkReveal (A) @ i6  & EphkeyReveal (s)@i7)
& not(Ex #i6 #i7. LtkReveal (B) @ i6  & EphkeyReveal (ss)@i7)
& not(Ex #i6 #i7. LtkReveal (A) @ i6 & LtkReveal  (B)@i7)
& not(Ex #i6 #i7. EphkeyReveal (s) @ i6
& EphkeyReveal(ss)@i7))) & ((not(Ex ss #i4 #i5 ms.
SessionID (ss, ms) @ i4 & MatchingSession(s, ms) @ i5))
==> (not(Ex #i6. EphkeyReveal (s) @ i6 )
& not(Ex #i6. LtkReveal (B) @ i6 & i6 < i1 ))))"
```

Fig. 9 ID-AKE—Tamarin Prover model—session key secrecy lemma

- *A Session Key Reveal Query for the obtained matching session.*
- *A Long-Term Key Reveal Query for Alice and Ephemeral Key Reveal Query for Alice's matching session.*
- *A Long-Term Key Reveal Query for Bob and Ephemeral Key Reveal Query for Bob's session ID.*
- *A Long-Term Key Reveal Query for both Alice and Bob.*
- *An Ephemeral Key Reveal Query for obtained matching session and the parties' session ID.*

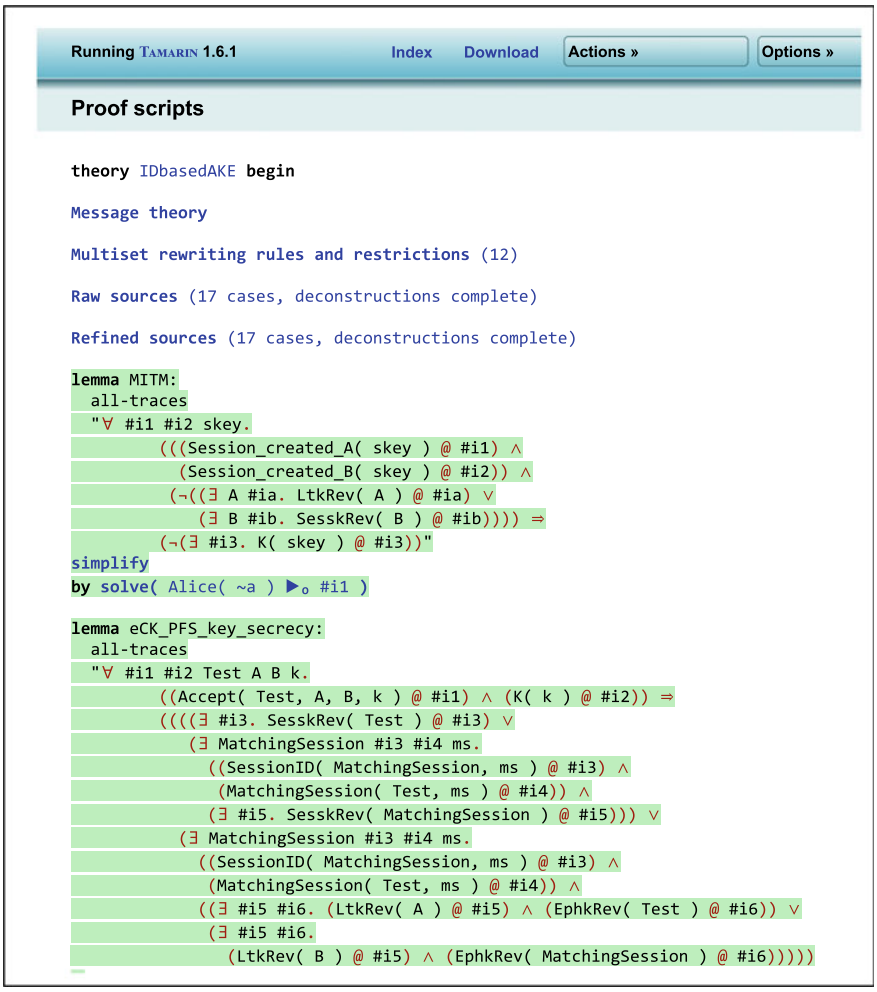

Fig. 10 The Tamarin Prover model visualisation for ID-AKE protocol

Finally, if the adversary did not find a matching session, it implies that there does not exist an Ephemeral Key Reveal Query for matching session. Also there does not exist a Long-Term Key Reveal call for Bob, thus stating that Bob has not been compromised.

Model Visualisation: Once Lemmas 2 and 3 are solved in the Tamarin Prover, the colour of the proof turns green as shown in Fig. 10. It is an indication that there were no traces found for any adversary computing the secret session key, k. Thus, we suggest that the designed ID-AKE protocol using the technique of precomputation and normalisation resists MITM attack and provides session key secrecy under the eCK adversary model.

7 Conclusion

In this paper, we study the designing techniques of security models using Tamarin Prover for various authentication protocols. It is observed that the point addition operation modelled in the literature is not applicable to many of the IBE-based protocols. In this work, we present a generalised IBE-based key exchange protocol and modelled it using the proposed normalised precomputation technique with the help of hash function. The Tamarin Prover simulations showed that the proposed technique provides security under the eCK adversary model. In conclusion, the proposed model can be applied to IBE-based protocols where the point addition operation is used.

References

1. Bresson E, Chevassut O, Pointcheval D (2007) Provably secure authenticated group diffie-hellman key exchange. ACM Trans Inf Syst Secur (TISSEC) 10(3):10–es
2. Chatterjee S, Sarkar P (2011) Identity-based encryption. Springer Science & Business Media
3. Chen L, Kudla C (2003) Identity based authenticated key agreement protocols from pairings. In: Proceedings of the 16th IEEE computer security foundations workshop. IEEE, pp 219–233
4. Das ML, Saxena A, Gulati VP (2004) A dynamic id-based remote user authentication scheme. IEEE Trans Consum Electron 50(2):629–631
5. Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654
6. Hölbl M, Welzer T (2009) An improved authentication protocol based on one-way hash functions and diffie-hellman key exchange. In: 2009 International conference on availability, reliability and security. IEEE, pp 894–898
7. Huang LC, Chang TY, Hwang MS (2018) A conference key scheme based on the diffie-hellman key exchange. Int J Netw Secur 20(6):1221–1226
8. Johnston AM, Gemmell PS (2002) Authenticated key exchange provably secure against the man-in-the-middle attack. J Cryptol 15(2):139–148
9. Joux A (2000) A one round protocol for tripartite diffie–hellman. In: International algorithmic number theory symposium. Springer, pp 385–393
10. LaMacchia B, Lauter K, Mityagin A (2007) Stronger security of authenticated key exchange. In: International conference on provable security. Springer, pp 1–16
11. Meier S, Schmidt B, Cremers C, Basin D (2013) The tamarin prover for the symbolic analysis of security protocols. In: International conference on computer aided verification. Springer, pp 696–701
12. Mookherji S, Odelu V, Prasath R (2021) Modelling ibe-based key exchange protocol using tamarin prover. Cryptology ePrint Archive
13. Odelu V, Das AK, Goswami A (2015) A secure biometrics-based multi-server authentication protocol using smart cards. IEEE Trans Inf Forensics Secur 10(9):1953–1966
14. Odelu V, Das AK, Wazid M, Conti M (2018) Provably secure authenticated key agreement scheme for smart grid. IEEE Trans Smart Grid 9(3):1900–1910. https://doi.org/10.1109/TSG.2016.2602282
15. Okamoto E, Masumoto H (1990) Id-based authentication system for computer virus detection. Electron Lett 26(15):1169–1170
16. Preneel B (1993) Analysis and design of cryptographic hash functions. Ph.D. thesis, Katholieke Universiteit te Leuven

17. Pu Q (2010) An improved two-factor authentication protocol. In: 2010 Second international conference on multimedia and information technology, vol 2, pp 223–226. https://doi.org/10.1109/MMIT.2010.82
18. Ruan O, Zhang Y, Zhang M, Zhou J, Harn L (2017) After-the-fact leakage-resilient identity-based authenticated key exchange. IEEE Syst J 12(2)
19. Ryu EK, Yoon EJ, Yoo KY (2004) An efficient id-based authenticated key agreement protocol from pairings. In: International conference on research in networking. Springer, pp 1458–1463
20. Schmidt B (2012) Formal analysis of key exchange protocols and physical protocols. Ph.D. thesis, ETH Zurich
21. Schmidt B, Sasse R, Cremers C, Basin D (2014) Automated verification of group key agreement protocols. In: 2014 IEEE Symposium on security and privacy. IEEE, pp 179–194
22. Shamir A (1984) Identity-based cryptosystems and signature schemes. In: Workshop on the theory and application of cryptographic techniques. Springer, pp 47–53
23. Shim KA (2012) A round-optimal three-party id-based authenticated key agreement protocol. Inf Sci 186(1):239–248
24. Stallings W (2006) Cryptography and network security principles and practices, 4th edn
25. Tamarin prover github repository. https://github.com/tamarin-prover/tamarin-prover. Accessed 20 Oct 2022
26. Tamarin prover github repository—chen-kudla protocol. https://github.com/tamarin-prover/tamarin-prover/blob/develop/examples/ake/bilinear/Chen_Kudla.spthy. Accessed 20 Oct 2022
27. Tamarin prover github repository—joux protocol. https://github.com/tamarin-prover/tamarin-prover/blob/develop/examples/ake/bilinear/Joux.spthy. Accessed 20 Oct 2022
28. Tsai JL, Lo NW (2015) A privacy-aware authentication scheme for distributed mobile cloud computing services. IEEE Syst J 9(3):805–815
29. Tsai JL, Lo NW (2015) Secure anonymous key distribution scheme for smart grid. IEEE Trans smart grid 7(2):906–914
30. Tseng YM, Huang SS, Tsai TT, Tseng L (2015) A novel id-based authentication and key exchange protocol resistant to ephemeral-secret-leakage attacks for mobile devices. Int J Distrib Sens Netw 11(5):898716
31. Wang C, Xu G, Li W (2018) A secure and anonymous two-factor authentication protocol in multiserver environment. Secur Commun Netw 2018