J Supercomput (2014) 70:31–53
DOI 10.1007/s11227-013-1051-3

An anonymous e-rental protocol based on ID-based cryptography and NFC

Jia Ning Luo · Ming Hour Yang

Published online: 12 December 2013

Abstract Most e-rental services require customers to register sensitive information, which gives malicious service providers a good opportunity to launch social engineering attacks, or to use data mining techniques collecting and analyzing customers' information or rental preferences. Therefore, we propose an anonymous e-rental protocol based on ID-based cryptography and near field communication technology, with particular focus on vehicle rentals. Our contributions include: (1) Anonymity. Users' real identity is hidden from the rental service providers. (2) Unlinkability. Rental service providers cannot find the relation between two rental records. (3) Traceability. As full anonymity is not always desirable, traceability allows disclosure of a malicious user's identity, whereas other users' privacy remains unviolated. Rental service providers can request TTP to reveal users' identity with a legal warrant. (4) Flexibility. Users choose their preferred service providers and vehicles. (5) Anonymous payment. Rental service providers cannot associate users' identity with the financial transactions.

Keywords Bilinear pairing · ID-based cryptography · NFC · Anonymous car rental

J. N. Luo
Department of Information and Telecommunications Engineering, Ming Chuan University,
5 De-Ming Road, Gui-Shan District, Taoyuan 33348, Taiwan
e-mail: deer@mail.mcu.edu.tw

M. H. Yang (✉)
Department of Information & Computer Science, Chung Yuan Christian University,
200 Chung Pei Road, Chung Li City, Taoyuan 32023, Taiwan
e-mail: mhyang@cycu.edu.tw

1 Introduction

More and more subscription and rental services have gone online in recent years, such as Apple's iTunes video and magazine services, Amazon's Kindle Lending Library, even car rental services, etc. Most of the time these service providers require customers to register their personal information. For example, in current e-car rental systems [1,2], customers have to provide ID certificates, telephone numbers, residential address, and driving license. Unfortunately, if a malicious provider collects customers' rental records or personal information, e.g. hobbies, lifestyles, credit numbers and frequent locations, customers' privacy may be compromised. Moreover, if the cars are equipped with GPS devices, users' travel routes can be completely tracked.

As this has attracted more and more privacy concerns, people are getting aware of the importance of renting a car without disclosing their own identity. They hope that a secured anonymous rental system must achieve anonymity, unlinkability, and traceability [3]. To guarantee anonymity, users provide their personal info to a trusted third party (TTP) instead of to a rental service provider. As for unlinkability, it means a rental service provider cannot find the relation between a user's rental records and his identity. But full anonymity is not always desirable [3]; traceability allows disclosure of a malicious user's identity, whereas other users' privacy remains unviolated. A rental service provider can even request a TTP to reveal users' identity if there are consumer disputes or accidents.

More and more studies in this area have been published to protect users' privacy [3–5]. Wang et al. proposed an e-cash system with anonymity revocation [4]. In Wang et al.'s protocol, users have to acquire an anonymous certificate from a TTP for their withdrawal and payment. When double spending occurs, banks will request the TTP to reveal a user's identity. But during Wang et al.'s withdrawal stage, banks can only verify whether users have the secrets of a certificate. They cannot confirm if the certificate is issued by the TTP. A malicious user can take such an opportunity to generate a random certificate to the banks. Because the user knows the certificate's content, he can certainly pass the bank's verification. In such a case, banks may issue e-cash to adversaries. When banks find this e-cash is double spent, even the TTP cannot track the attackers through the randomly generated certificate.

Several online subscription systems have been proposed in recent years. In 2008 Blanton [6] proposed a method that allows an anonymous user to have unlimited times of access to the subscription service during each time interval, and no links can be established between a user's identity and his rental records. In 2010 Chen et al. [5] proposed a new e-cash system with anonymity revocation and mutual authentication. They improved Wang et al.'s scheme by enhancing its certificate validation and by performing mutual authentication, but their scheme does not guarantee forward secrecy. If an attacker knows the secret key of a TTP, a bank or an agency, he may eavesdrop on their communications. The attacker can compute their session key, decrypt their messages, and use counterfeit e-cash for payment. In 2010, Slamanig and Rass [3] proposed an e-service system that allows anonymous transactions. They use tamper-resistant smart cards to store anonymous certificates. Despite high portability of smart cards, their convenience is restricted because they cannot work without a card reader. Their scheme is able to identify malicious users behind transactions, but it cannot provide a fine-grained access control of the resources in their subscription service. In 2012 Vasco et al. [7] proposed an anonymous subscription scheme to protect user's anonymity and to provide fraud detection. It allows a set of users to buy access to a limited set of services, but cannot disclose a fraudster's identity. In 2013, Lee et al. [8] proposed an anonymous subscription system. The scheme is based on a single-login platform and does not require users to provide their real identity for registration, so as to protect their privacy.

Because ID-based encryption takes users' personal information such as names, phone numbers, ID numbers, or emails to generate public keys, some schemes are based on bilinear pairing ID-based encryption [9,10]. The key-generation center (KGC) uses public keys to generate private keys with RSA algorithms [11]. In 1985, both Koblitz and Miller [12,13] proposed new encryption schemes based on elliptic curves. In 2001, Boneh and Franklin [14] proposed a bilinear pairing ID-based encryption scheme. Compared with RSA, their method requires shorter keys but has better efficiency in computation.

In this paper, we propose an anonymous e-rental protocol based on ID-based cryptography and NFC technology, with particular focus on vehicle rentals. Our protocol uses an NFC-enabled phone for authentication and the vehicle's access control. In our mechanism, even though a malicious user can compromise a mobile phone, e.g. modifying the rental applications, he cannot clone the secrets inside because they are stored in the secure element.

Near field communication is based on radio frequency identification (RFID) standards and complies with ISO/IEC 18092 [15], ISO/IEC 21481, ECMA 340, and ETSI TS 102 190. It is also compatible with existing contactless smart card standards ISO/IEC 14443. NFC operates at 13.56 MHz, and its transmission is limited to 10–12 cm at the rate of 106, 212, or 424 kbps. As illustrated in Fig. 1, NFC-enabled phones consist of three NFC components: NFC controller, secure element (SE), and RF antenna.

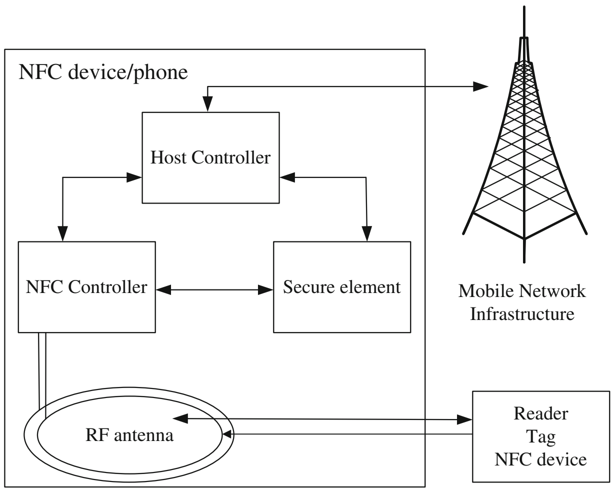

Fig. 1 Components of an NFC-enabled phone [16]

- Secure element (SE): A secure element is a tamper-resistant device that contains a secure memory area that stores confidential and cryptographic data, and a micro-controller that runs securely hosting applications.
- NFC controller: It serves as a bridge for host controller (the mobile phone OS), SE, and RF antenna. NFC controller has three operation modes: card emulation, P2P mode, and reader/writer mode.
- RF antenna: It is responsible for transmission and reception of radio signals.

Our protocol uses NFC phones as user's mobile devices, because NFC phones can serve as a smart card and a card reader. Besides, the secure element (SE) in an NFC phone is tamper proofed and stores user's private data. The cryptosystem in this paper is based on bilinear pairing identity-based certificateless signature schemes.

The steps of our car-rental protocol are as follows: (1) Users provide their personal information to a TTP, such as ID number, driving license, credit card number, and traffic offense records. The TTP is responsible for identity authentication and payment. (2) Users take their NFC phones to request a temporary anonymous license from the TTP. They can browse the web to choose a car to rent. They have to send the temporary license and the car's info to the company. After the company confirms that the license has been generated by the TTP, it issues a ticket for a specific vehicle and sends it to users' NFC phones. (3) Users receive the ticket and go to the pickup place. They hold their NFC phones near the car's onboard unit (OBU) for authentication. If the ticket is verified, users can drive the car. (4) When the car is returned, the company collects its payments through the TTP. (5) If there are consumer disputes or accidents, the rental service provider can request the TTP to reveal the driver's identity.

The main contributions of our scheme include: (1) high security and convenience of a car-rental system that combines NFC phones with ID-based cryptography; (2) a privacy-based rental system that allows anonymous car hiring without sacrificing the rental service providers' interests; (3) free choice of rental service providers; (4) anonymous payment.

The next section reviews the related studies in this area. The third section provides the preliminaries and detailed steps of our protocol. The fourth section deals with the security analysis of our method. In the fifth section, we use GNY proof to prove that our protocol is a viable scheme. A conclusion is drawn in Sect. 6.

2 Related work

Section 2.1 deals with bilinear pairing and its application. Section 2.2 analyzes related studies.

2.1 ID-based encryption from bilinear pairings

Shamir [10] proposes an ID-based encryption scheme taking users' identity as public keys for better key management and distribution. Boneh and Franklin [14] propose an ID-based encryption from bilinear pairings. It achieves efficient computation of keys. Since then, many encryption schemes of this kind have been proposed [17].

In bilinear pairing ID-based encryption, the key generation center (KGC) is responsible for parameter setting and key distribution. It generates $\{G_1, G_2, P, q, e\}$ as system parameters, where G_1 denotes a cyclic additive group of order q, and P a generator of this group; G_2 denotes a cyclic multiplicative group of the same order q in a finite field; e denotes a mapping function, i.e., $e : G_1 \times G_1 \rightarrow G_2$. It also defines a map-to-point hash function $H_1 : \{0, 1\}^* \rightarrow G_1$to compute public keys and to map users' identity to a point over an elliptic curve. Meanwhile, KGC generates a random number $s \in Z_q^*$ as a private key and then uses it to compute its public key $P_{pub} = sP$. When a user sends his/her identity to KGC, it uses it to generate a key pair and return a public key $Q_{ID} = H_1(ID)$ and a private key $S_{ID} = sQ_{ID}$.

The following paragraph introduces the application of bilinear pairing ID-based encryption.

- En/decryption in Boneh and Franklin's [14] scheme:
  - Encryption: If Alice wants to send a message $M \in \{0, 1\}^n$ to Bob, Alice has to use Bob's public key Q_B to encrypt M. The steps are: (1) Alice computes Bob's public key $Q_B = H_1(ID_B)$; (2) generates a random number $\sigma \in \{0, 1\}^n$; (3) computes $r = H_3(\sigma, M)$; (4) encrypts the message as C= $\{rP, \sigma\oplus \mathrm{H}_2(\mathrm{e}(\mathrm{Q}_B,\mathrm{P}_{pub})^r), M\oplus \mathrm{H}_4(\sigma)\} = \{U, V, W\}$, and sends the encrypted message to Bob.
  - Decryption: After Bob receives the message $C = \{U, V, W\}$, he uses his private key S_B to decrypt C. The steps are: (1) Bob computes $V \oplus H_2(e(S_B,\mathrm{U})) = \sigma$; (2)$W \oplus H_4(\sigma) = M$; (3)$r = H_3(\sigma,\mathrm{M})$, and verifies rP ?= U; (4) if it is verified, Bob confirms the reception of message M.
- Key agreement protocol (KAP): Zhang et al. [18] propose a secured and efficient session key agreement protocol. (1) Both Alice and Bob generate a random value $a, b \in Z_q^*$, compute $T_A = aP$ and $T_B = bP$ respectively, and then send T_A and T_B to each other. (2) Alice and Bob compute their shared keys: $K_{AB} = e(aP_{pub} + S_A, T_B + Q_B)\ \ e(S_A, Q_B)$ and $K_{BA} = e(T_A + Q_A, bP_{pub} + S_B)\ \ e(Q_A, S_B)$, where $K_{AB} = K_{BA} = e(P, P)^{abs}\ \ e(P, Q_B)^{as}\ \ e(Q_A, P)^{bs}\ \ e(Q_A, Q_B)^{2s}$. And their shared session key is $SK_{AB} = H(A, B, h, K_{AB})$, where $H()$ denotes a one-way hash function, and $h = aT_B = bT_A$.

  Function 1:

$$\begin{aligned} K_{AB} &= e(aP_{pub} + S_A, T_B + Q_B)e(S_A, Q_B) \\ &= e(aP_{pub}, bP + Q_B)e(S_A, bP + Q_B)e(S_A, Q_B) \\ &= e(aP_{pub}, bP)e(aP_{pub}, Q_B)e(S_A, bP)e(S_A, Q_B)e(S_A, Q_B) \\ &= e(P, P)^{abs}e(P, Q_B)^{as}e(Q_A, P)^{bs}e(Q_A, Q_B)^{2s} \end{aligned}$$

  Function 2:

$$\begin{aligned} K_{BA} &= e(T_A + Q_A, bP_{pub} + S_B)e(Q_A, S_B) \\ &= e(aP, bP_{pub} + S_B)e(Q_A, bP_{pub} + S_B)e(Q_A, S_B) \\ &= e(aP, bP_{pub})e(aP, S_B)e(Q_A, bP_{pub})e(Q_A, S_B)e(Q_A, S_B) \\ &= e(P, P)^{abs}e(P, Q_B)^{as}e(Q_A, P)^{bs}e(Q_A, Q_B)^{2s} = K_{AB} \end{aligned}$$

2.2 Privacy protection and related studies

In 2008, Wang et al. [4] proposed an e-cash system that can revoke anonymity. In 2010, Chen et al. proposed a new one. Both of their schemes are based on bilinear pairing ID-based cryptography. They work with a TTP to issue an anonymous certificate for the user and store it on a server. This certificate will be used in e-cash withdrawal and payment. If double spending occurs, the banks can request the TTP to reveal the users' identity.

In Wang et al.'s scheme, when users try to withdraw e-cash, they have to prove that they know the secret value x of X, where $X = xP$ and is one of users' certificates; $P \in G_1$; and $x \in Z_q^*$. The users send X to the bank and the bank responds with a random number $w \in Z_q^*$. The users use their private keys to sign w: pick a random number $k \in Z_q^*$; compute $c = H_1(w, P, xP, kP)$, $s = k - cx$; and return $\{c, s\}$ to the bank. The bank verifies the signature by computing $c' = H_1(w, P, X, sP - cxP)? = c$. However, the bank is unable to verify whether X comes from the TTP or not. It means malicious users can take $X' \in Z_q^*$ to compute $X' = x'P$. They can prove to the bank that they know the secret value x', and then use a fake certificate X' to pass the bank's authentication. At last, they withdraw the e-cash.

In Chen et al.'s protocol, users need to ask the TTP for a license *License* = {LST, LVT} in the license-issuing stage. Next, they compute a session key shared with the bank $K_{CB} = H_2(e(S_C, kQ_B))$; encrypt the authentication message $Auth_{CB} = \{ID_C, ID_B, k, b^2 H_3(\text{CNO}), b^{-1}\text{LST}, \text{LVT}, T_S\}$; and return $EK_{CB}(Auth_{CB})$ and kQ_C to the bank. After receiving the message, the bank decrypts it with the shared session key $K_{BC} = H_2(e(S_B, kQ_C))$. It then verifies $e(V, P) = e(W, P_{pub})e(H_3(b^{-1}\text{LST}, W)Q_T, P_{pub})$. If it is verified that the license is issued by the TTP, the bank performs a blind signature on the e-cash and returns it to the users. The users receive and unblind it and sends the anonymous e-cash *e-cash* = {CNO, LST, (R, S)} to a merchant. The merchant has to verify the e-cash by examining its bank-issued signature and sends it to the bank to claim payment. If the bank finds double spending in the deal, it can send a request to the TTP to trace the user's real identity. Unfortunately, we have found that the session keys' forward secrecy is not guaranteed in each session of the scheme. We take the withdrawal stage as an example. If malicious users know the bank's secret key S_B, they may eavesdrop on $\{EK_{CB}(Auth_{CB}), \text{B}kQ_C\}$. Then they are able to compute the user-bank shared session key $K_{CB} = H_2(e(S_C, kQ_B)) = H_2(e(S_B, kQ_C)) = K_{BC}$. If they have S_B and kQ_C, they can act as a counterfeit bank to compute the session key $K_{CB} = K_{BC} = K_{attack} = H_2(e(S_B, kQ_C))$; and to decrypt $EK_{CB}(Auth_{CB})$.

3 An anonymous e-rental protocol based on ID-based cryptography and NFC

As shown in Fig. 2, our anonymous e-rental protocol consists of five elements: TTP, rental service provider, user (NFC phone), vehicle (OBU), and bank. It includes four stages: anonymous license issuing, vehicle ticket issuing, vehicle ticket verification, and vehicle control.

The elements are described as follows:

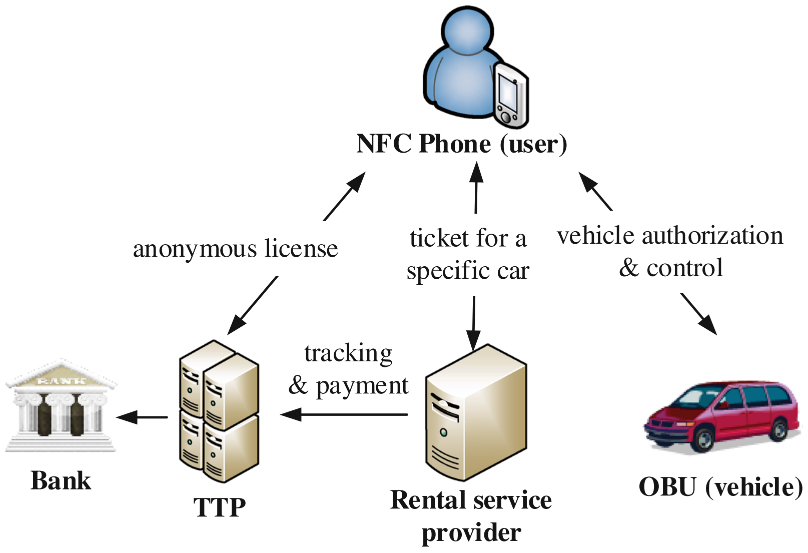

Fig. 2 Infrastructure of our anonymous e-rental protocol

- TTP: It performs mutual authentication with users and is responsible for issuing anonymous licenses. A rental service provider should verify an anonymous license with TTP and claim for payments.
- User (NFC phone): The user should use an NFC-enabled mobile phone that contains a broker application. To launch the application, the user's PIN number [19] is required to access the secure element (SE) of the NFC. The broker application helps users browse vehicle information on the Internet, such as car type, location, price, and its rental service provider. The secure element is mainly responsible for the encryption/decryption of the communications and storage keys.
- Rental service provider: A rental service provider owns the vehicles waiting for rent. The provider should verify users' anonymous licenses with TTP. If a license is valid, the provider issues a vehicle ticket for a specific vehicle to the users. After the vehicle is returned, the provider collects user' payments through the TTP.
- Vehicle (OBU): Each vehicle is equipped with an OBU with an NFC reader. The OBU checks the validity of users' vehicle tickets. If the tickets are verified, users are authorized to control the vehicle.
- Bank: The bank has to cooperate with the TTP and is responsible for the payments.

Details of the four stages are as follows:

1. Anonymous license issuing: Users perform mutual authentication with the TTP. If users' identity is confirmed, the TTP issues a temporary anonymous license to them. Users can use the license for car hiring within its validity period.
2. Vehicle ticket issuing: After the APP helps users choose their preferred vehicle, it sends the anonymous license and the chosen vehicle's information to the company to which the vehicle belongs. If the license is verified, the company issues a ticket to the users.

3. Vehicle ticket verification: Users hold their NFC phones close to the OBU to perform ticket verification. After the OBU confirms its own provider issues the ticket, and the ticket is still valid, it authorizes the users to control the vehicle.
4. Vehicle control: During car hiring, users are allowed to use their NFC phones to give commands to the car such as door open, door close, ignition, and return.

3.1 Notations

The notations we use in our car-rental system are listed in Table 1.

Table 1 Notations

q	A large prime number
P	Generator: $P \in G_1$
G_1	A cyclic additive group of order q with P as its generator
G_2	A cyclic multiplicative group of order q in a finite field
e	A mapping function$e$: $G_1 \times G_1 \rightarrow G_2$
H_1	A map-to-point hash function H_1: $\{0, 1\}^* \rightarrow G_1$
H_2	H_2: $G_2 \rightarrow \{0,1\}^n$
H_3	H_3: $\{0,1\}^n \times \{0,1\}^n \rightarrow Z_q^*$
H_4	H_4: $\{0,1\}^* \rightarrow \{0,1\}^n$
$f(x)$	Function $f(x)$: $x = x + 1$
s	A private key $s \in Z_q^*$
P_{pub}	A public key $P_{pub} = sP$
$Params$	Public system parameters $\{G_1, G_2, q, P, P_{pub}, e, n, H_1, H_2, H_3, H_4, f(x)\}$
ID_{SEi}	Identifier of secure element SE_i
ID_T	Identifier of TTP
ID_{Sj}	Identifier of rental service provider S_j
ID_{Ck}	Identifier of vehicle C_k's OBU
Q_{ID}	Public key based on each element's identifier; $Q_{ID} = H_1(ID)$
S_{ID}	Private key based on each element's identifier; $S_{ID} = sQ_{ID}$
TID_i	Anonymous identifier issued to SE_i by TTP
AuthList	A list for TTP to store a user's temporary anonymous identity
RentList	A list for TTP to store all of the use's' rental records.
CarList	A list for rental service provider to store the rented vehicle's information
$License_i$	Anonymous license issued to SE_i by TTP
$Ticket_i$	Vehicle ticket issued to SE_i by rental service provider S_j
$SK_{x,y}$	Session key between x and y
K_{LICi}	A key for TTP to encrypt anonymous license
K_{SC}	OBU-company shared key for ticket encryption
$validity$	Valid day(s) of anonymous license
$ExpTime_i$	Expiry date of anonymous license
$RentTime_i$	Expiry date of vehicle ticket

3.2 Preliminaries

In our preliminaries, we explain in detail how we initialize our car-rental system and clarify each role's function in our system, i.e. TTP, rental service provider, rental APP, SE, and OBU.

3.2.1 Initialization of our system

The system generates G_1, G_2, P, and e: $G_1 \times \quad G_1 \rightarrow G_2$; and it defines the following functions:

$$\begin{aligned} H_1 &: \{0,1\}^* \rightarrow G_1; \\ H_2 &: G_2 \rightarrow \{0,1\}^n ; \\ H_3 &: \{0,1\}^n \times \{0,1\}^n \rightarrow Z_q^*; \\ H_4 &: \{0,1\}^* \rightarrow \{0,1\}^n ; \\ f(x) &: x = x + \ 1. \end{aligned}$$

In elliptic curve cryptography [20–22], a 160-bit key can achieve RSA's 1,024-bit security. To make a trade-off between the security baseline and low computational loads for mobile devices, here we take 160 bits as the default value of our key length. Keys longer than 160 bits can of course have even higher security strength.

The detailed functions of these hashes are as follows:

H_1 is a map-to-point hash function, which converts an arbitrary string $\{0,1\}^*$ to a point on an elliptic curve G_1, e.g., the MapToPoint function in Boneh and Franklin's [14] identity-based scheme.

H_2 is a hash function that maps a point on an elliptic curve G_2 to a fixed-length string.

H_3 is a one-way hash function that converts two fixed-length string $\{0,1\}^n$ to an integer in the Z_q^* domain.

H_4 is a traditional hash function that maps an arbitrary string $\{0,1\}^*$ to a fixed-length string $\{0,1\}^n$ The system generates a random number s as its private key and computes $P_{pub} = sP$ as its public key. It also makes its system parameters public, i.e., *Params* = $\{G_1, G_2, q, P, P_{pub}, e, n, H_1, H_2, H_3, H_4, f(x)\}$.

During initialization, each role in our protocol has a unique identifier: ID_T for TTP, ID_{SEi} for SE, ID_{Sj} for rental service provider, and ID_{Ck} for OBU, respectively. According to each role's identifier, the system computes a public key for each of them, i.e., $Q_{ID} = H_1(ID)$, and generates their corresponding private key $S_{ID} = sQ_{ID}$. Each role receives a pair of keys from the system at the initial stage. For an NFC phone, it has a key pair $\{Q_{SEi}, S_{SEi}\}$ stored in the SE; for a TTP, it is $\{Q_T, S_T\}$; for a rental service provider, $\{Q_{Sj}, S_{Sj}\}$; for an OBU, $\{Q_{Ck}, S_{Ck}\}$, respectively.

As shown in Table 2, TTP's AuthList includes: user's identifier ID_{SEi}; user's temporary identifier TID_i; the expiry date, $ExpTime_i$, of an anonymous license; a session key, $SK_{i,T}$, between a TTP and its user; and a key K_{LICi} for TTP to encrypt an anonymous license. Initially, AuthList is an empty set, i.e. AuthList = ϕ.

Table 3 depicts the content of TTP's RentList: TID_i; the expiry date, $RentTime_i$, of a ticket; and a rental service provider's identifier ID_{Sj}. Initially, RentList = ϕ.

Table 2 Content of TTP's AuthList

ID_{SEi}	TID_i	$ExpTime_i$	$SK_{i,T}$	K_{LICi}

Table 3 Content of TTP's RentList

TID_i	$RentTime_i$	ID_{Sj}

Table 4 Content of rental service provider's CarList

TID_i	$RentTime_i$	$SK_{i,j}$	ID_{Ck}	$ReturnConfirm$

Table 5 List of public and private information

Roles	Private information	Public information
TTP	$\{Q_T, S_T\}$; AuthList; RentList	ID_T
Rental service provider	$\{Q_{Sj}, S_{Sj}\}$; K_{SC}; CarList	ID_{Sj}
SE	$\{Q_{SEi}, S_{SEi}\}$	ID_{SEi}
OBU	$\{Q_{Ck}, S_{Ck}\}$; K_{SC}	ID_{Ck}

Table 6 List of each role's functions

	TTP	Rental service provider	Car-rental APP	SE	OBU
Encrypt()	✓	✓		✓	✓
Decrypt()	✓	✓		✓	✓
IBE_Encrypt()	✓		✓	✓	
IBE_Decrypt()	✓		✓	✓	
ComputeSK()	✓			✓	

Table 4 shows the content of a rental service provider's CarList: TID_i; $RentTime_i$; $SK_{i,j}$; OBU's identifier ID_{Ck}; and a flag *ReturnConfirm*, which is used to confirm vehicle return.

After initialization, each role's public and private information is listed in Table 5.

3.2.2 Function

This subsection clarifies the functions of each role; see Table 6.

- Encrypt() uses a key K to encrypt *message*.
  Input: K, *message*
  Output: C
  $C = E_K(message)$
  return C

- Decrypt() uses K to decrypt ciphertext $E_K(message)$.
  Input: K, $E_K(message)$
  Output: *message*

- IBE_Encrypt() uses an ID-based cryptosystem [14] to encrypt message M with the public key Q_{ID} and returns C

```
Input: $M, ID$
Output: $C$
$\sigma \quad = \{0,1\}^n$
$r = H_3(\sigma, M)$
$C = \{rP, \sigma \oplus H_2(e(Q_{ID}, P_{pub})^r), M \oplus H_4(\sigma)\} = \{U, V, W\}$
return $C$
```

- IBE_Decrypt() uses an ID-based cryptosystem to decrypt the ciphertext C with the private key S_{ID}. Then it verifies the user's identity U and returns M or Reject.

```
Input: $C, S_{ID}$
Output: $M$ or Reject
$V \oplus H_2(e(S_{ID}, U)) = \sigma$
$W \oplus H_4(\sigma) = M$
Set $r = H_3(\sigma, M)$
if $U = rP$
return $M$
else
return Reject
```

- ComputeSK() uses ID_A's private key S_A and $\{ID_B, T_B\}$ to compute a session key SK_{AB}, whose key agreement is based on Zhang et al.'s [18] scheme. k denotes a random number generated by ID_A.

```
Input: $k, S_A, ID_B, T_B$
Output: $SK_{AB}$
$Q_B = H_1(ID_B)$
$K_A = e(T_B + Q_B, kP_{pub} + S_A) \quad e(Q_B, S_A)$
$SK_{AB} = H_4(ID_A \,||\, ID_B \,||\, K_A \,||\, kT_B)$
return $SK_{AB}$.
```

3.3 Proposed anonymous e-rental protocol

Our anonymous e-rental system consists of four parts: (1) issuing of anonymous license; (2) issuing of ticket; (3) verification of ticket; (4) vehicle control.

3.3.1 Anonymous license issuing protocol

The steps of our license issuing protocol are demonstrated in Fig. 3.

Step 1 User's car-rental APP requests a random number T_{SEi} from SE.

Step 2 SE generates a random number $a \in Z_q^*$, computes $T_{SEi} = aP$, and returns the random value to the APP.

Step 3 The APP generates a nonce $Nonce_{SEi} \in \{0,1\}^n$; uses TTP's public key Q_T to encrypt $\{ID_{SEi}||T_{SEi}||\, Nonce_{SEi}\}$ as ciphertext C and sends $\{ID_{SEi}, C\}$ to TTP.

Step 4 TTP perform the following actions:

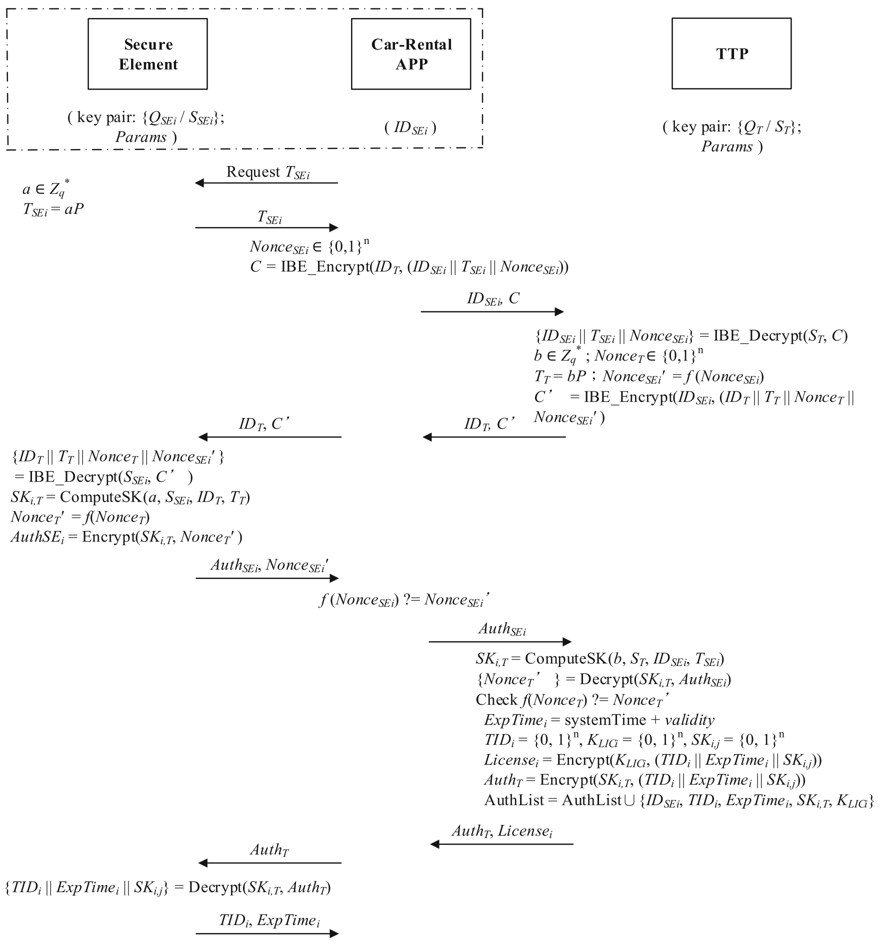
Secure Element
Car-Rental APP
TTP
(key pair: {Q_SEi / S_SEi}; Params)
(ID_SEi)
(key pair: {Q_T / S_T}; Params)
Request T_SEi
a ∈ Z_q^*
T_SEi = aP
T_SEi
Nonce_SEi ∈ {0,1}^n
C = IBE_Encrypt(ID_T, (ID_SEi || T_SEi || Nonce_SEi))
ID_SEi, C
{ID_SEi || T_SEi || Nonce_SEi} = IBE_Decrypt(S_T, C)
b ∈ Z_q^* ; Nonce_T ∈ {0,1}^n
T_T = bP ; Nonce_SEi' = f(Nonce_SEi)
C' = IBE_Encrypt(ID_SEi, (ID_T || T_T || Nonce_T || Nonce_SEi')
ID_T, C'
ID_T, C'
{ID_T || T_T || Nonce_T || Nonce_SEi' } = IBE_Decrypt(S_SEi, C')
SK_i,T = ComputeSK(a, S_SEi, ID_T, T_T)
Nonce_T' = f(Nonce_T)
AuthSE_i = Encrypt(SK_i,T, Nonce_T')
Auth_SEi, Nonce_SEi'
f(Nonce_SEi) ?= Nonce_SEi'
Auth_SEi
SK_i,T = ComputeSK(b, S_T, ID_SEi, T_SEi)
{Nonce_T'} = Decrypt(SK_i,T, Auth_SEi)
Check f(Nonce_T) ?= Nonce_T'
ExpTime_i = systemTime + validity
TID_i = {0, 1}^n, K_LIG = {0, 1}^n, SK_i,j = {0, 1}^n
License_i = Encrypt(K_LIG, (TID_i || ExpTime_i || SK_i,j))
Auth_T = Encrypt(SK_i,T, (TID_i || ExpTime_i || SK_i,j))
AuthList = AuthList ∪ {ID_SEi, TID_i, ExpTime_i, SK_i,T, K_LIG}
Auth_T, License_i
Auth_T
{TID_i || ExpTime_i || SK_i,j} = Decrypt(SK_i,T, Auth_T)
TID_i, ExpTime_i

Fig. 3 Anonymous license issuing protocol

(a) TTP uses his own private key S_T to decrypt C to derive $\{ID_{SEi}||T_{SEi}||Nonce_{SEi}\}$ and verifies whether the message is sent by ID_{SEi}.
(b) TTP generates a random number $b \in Z_q^*$ and a nonce $Nonce_T \in \{0,1\}^n$. It also requires that $T_T = bP$ and $Nonce'_{SEi} = f(Nonce_{SEi})$.
(c) TTP uses user's public key Q_{SEi} to encrypt $\{ID_T||T_T||Nonce_T||Nonce'_{SEi}\}$ as ciphertext C' and then returns $\{ID_T, C'\}$ to the user.

Step 5 APP forwards the message to SE, and SE's actions include:

(a) SE uses S_{SEi} to decrypt C', obtains $\{ID_T||T_T||Nonce_T||Nonce'_{SEi}\}$, and verifies whether the message is sent by TTP.
(b) SE uses $\{a, S_{SEi}, ID_T, T_T\}$ to compute the session key $SK_{i,T}$, shared with TTP.
(c) SE requires that $Nonce'_T = f(Nonce_T)$; uses $SK_{i,T}$ to encrypt $Nonce'_T$ as $Auth_{SEi}$ and returns $\{Auth_{SEi}, Nonce'_{SEi}\}$ to TTP.

Step 6 After APP verifies $Nonce'_{SEi}$, it sends $Auth_{SEi}$ to TTP.

Step 7 TTP's actions include:

(a) TTP uses $\{b, S_T, ID_{SEi}, T_{SEi}\}$ to compute the session key $SK_{i,T}$ using to decrypt $Auth_{SEi}$. Then it verifies $Nonce'_T$.

(b) TTP issues a temporary anonymous license:

(i) It computes the expiry date of the license $ExpTime_i$ = systemTime + *validity*.

(ii) It generates an anonymous identifier $TID_i \in \{0,1\}^n$; a key $K_{LICi} \in \{0,1\}^n$; and a session key $SK_{i,j} \in \{0,1\}^n$ for the user and company.

(iii) It uses K_{LICi} to encrypt $\{TID_i || ExpTime_i || SK_{i,j}\}$ as an anonymous license $License_i$; also, it uses $SK_{i,T}$ to encrypt $\{TID_i || ExpTime_i || SK_{i,j}\}$ as $Auth_T$.

(c) TTP stores $\{ID_{SEi}, TID_i, ExpTime_i, SK_{i,T}, K_{LICi}\}$ into the list AuthList and returns $\{Auth_T, License_i\}$ to the user.

Step 8 APP forwards $Auth_T$ to SE. SE uses $SK_{i,T}$ to decrypt the message, obtains $\{TID_i || ExpTime_i || SK_{i,j}\}$, stores $SK_{i,j}$, and returns $\{TID_i, ExpTime_i\}$ to APP.

Step 9 APP stores $\{TID_i, ExpTime_i, License_i\}$.

3.3.2 Ticket issuing protocol

The steps of the ticket issuing protocol are illustrated in Fig. 4.

Step 1 Users use the car-rental APP to search for available vehicles near their location. Users select their preferred vehicle and choose their planned rental period. The APP forwards the selected vehicle's information $\{TID_i, RentTime_i, ID_{Sj}, ID_{Ck}, Price_{Ck}\}$ to SE for encryption.

Step 2 SE uses $SK_{i,T}$ to encrypt the vehicle information and then returns the ciphertext $Auth_{SEi}$ to APP.

Step 3 APP sends $\{TID_i, ExpTime'_i, License_i, Auth_{SEi}\}$ to the rental service provider.

Step 4 The rental service provider sends $\{ID_{Sj}, TID_i, ExpTime'_i, Auth_{SEi}\}$ to TTP.

Step 5 TTP's actions include:

(a) TTP verifies whether the license has expired by comparing $ExpTime'_i$ with $ExpTime_i$; checks the validity of TID_i; and checks if the license has been used somewhere. Any failure in the verification or checking will abort the session.

(b) TTP uses $SK_{i,T}$ to decrypt $Auth_{SEi}$; obtains $\{TID_i || RentTime_i || ID_{Sj} || ID_{Ck} || Price_{Ck}\}$; and verifies $\{TID_i, ID_{Sj}\}$ so as to confirm the user's reservation with the service provider. This prevents malicious rental service providers' abuse of users' personal data in fake deals.

(c) TTP returns $\{K_{LICi}, RentTime_i, ID_{Ck}, Price_{Ck}\}$ to the rental service provider.

Step 6 The rental service provider's actions include the following:

(a) The service provider uses K_{LICi} to decrypt $License_i$; obtains $\{TID_i || ExpTime_i || SK_{i,j}\}$; and verifies $\{TID_i, ExpTime_i, Price_{Ck}\}$.

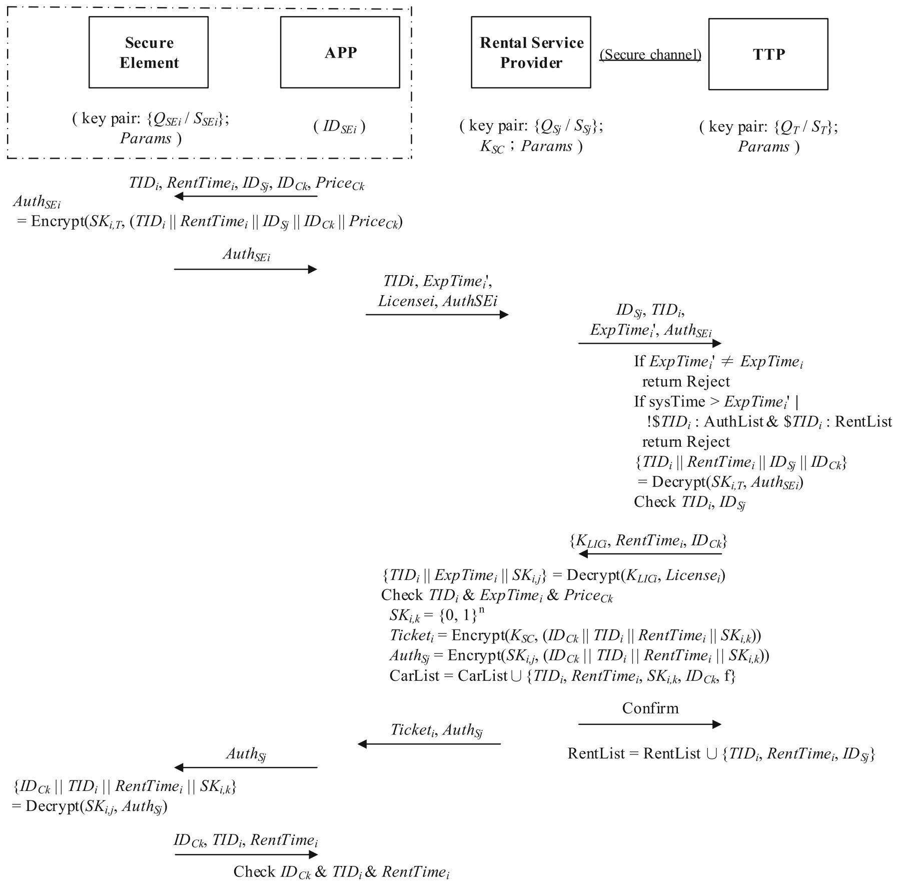
Secure Element
APP
Rental Service Provider
(Secure channel)
TTP
(key pair: {Q_{SEi} / S_{SEi}}; Params)
(ID_{SEi})
(key pair: {Q_{Sj} / S_{Sj}}; K_{SC} ; Params)
(key pair: {Q_T / S_T}; Params)
TID_i, $RentTime_i$, ID_{Sj}, ID_{Ck}, $Price_{Ck}$
$Auth_{SEi}$ = Encrypt($SK_{i,T}$, (TID_i || $RentTime_i$ || ID_{Sj} || ID_{Ck} || $Price_{Ck}$)
$Auth_{SEi}$
TIDi, $ExpTime_i'$, Licensei, AuthSEi
ID_{Sj}, TID_i, $ExpTime_i'$, $Auth_{SEi}$
If $ExpTime_i' \neq ExpTime_i$
return Reject
If sysTime > $ExpTime_i'$ |
!\$$TID_i$: AuthList & \$$TID_i$: RentList
return Reject
{TID_i || $RentTime_i$ || ID_{Sj} || ID_{Ck}}
= Decrypt($SK_{i,T}$, $Auth_{SEi}$)
Check TID_i, ID_{Sj}
{K_{LICi}, $RentTime_i$, ID_{Ck}}
{TID_i || $ExpTime_i$ || $SK_{i,j}$} = Decrypt(K_{LICi}, $License_i$)
Check TID_i & $ExpTime_i$ & $Price_{Ck}$
$SK_{i,k}$ = {0, 1}^{n}
$Ticket_i$ = Encrypt(K_{SC}, (ID_{Ck} || TID_i || $RentTime_i$ || $SK_{i,k}$))
$Auth_{Sj}$ = Encrypt($SK_{i,j}$, (ID_{Ck} || TID_i || $RentTime_i$ || $SK_{i,k}$))
CarList = CarList ∪ {TID_i, $RentTime_i$, $SK_{i,k}$, ID_{Ck}, f}
Confirm
$Ticket_i$, $Auth_{Sj}$
RentList = RentList ∪ {TID_i, $RentTime_i$, ID_{Sj}}
$Auth_{Sj}$
{ID_{Ck} || TID_i || $RentTime_i$ || $SK_{i,k}$}
= Decrypt($SK_{i,j}$, $Auth_{Sj}$)
ID_{Ck}, TID_i, $RentTime_i$
Check ID_{Ck} & TID_i & $RentTime_i$

Fig. 4 Protocol of ticket issuing

(b) The service provider issues a temporary ticket for a specific vehicle.
  (i) It generates a session key $SK_{i,k} \in \{0,1\}^n$ for OBU and the user.
  (ii) It uses K_{SC} to encrypt $\{ID_{Ck} || TID_i || RentTime_i || SK_{i,k}\}$ as $Ticket_i$ and uses $SK_{i,j}$ to encrypt the same message as $Auth_{Sj}$.

(c) The service provider stores $\{TID_i, RentTime_i, SK_{i,k}, ID_{Ck}, \phi\}$ into CarList and runs following steps:
  (i) It returns a confirmation message to TTP and the TTP stores $\{TID_i, RentTime_i, ID_{Sj}, Price_{Ck}\}$ into RentList.
  (ii) It returns $\{Ticket_i, Auth_{Sj}\}$ to the user.

Step 7 The APP forwards $Auth_{Sj}$ to SE. SE decrypts it with $SK_{i,j}$; obtains $\{ID_{Ck} || TID_i || RentTime_i || SK_{i,k}\}$; stores $SK_{i,k}$; and returns $\{ID_{Ck}, TID_i, RentTime_i\}$ to APP.

Step 8 After APP verifies $\{ID_{Ck}, TID_i, RentTime_i\}$, it stores $\{ID_{Ck}, TID_i, RentTime_i, Ticket_i\}$.

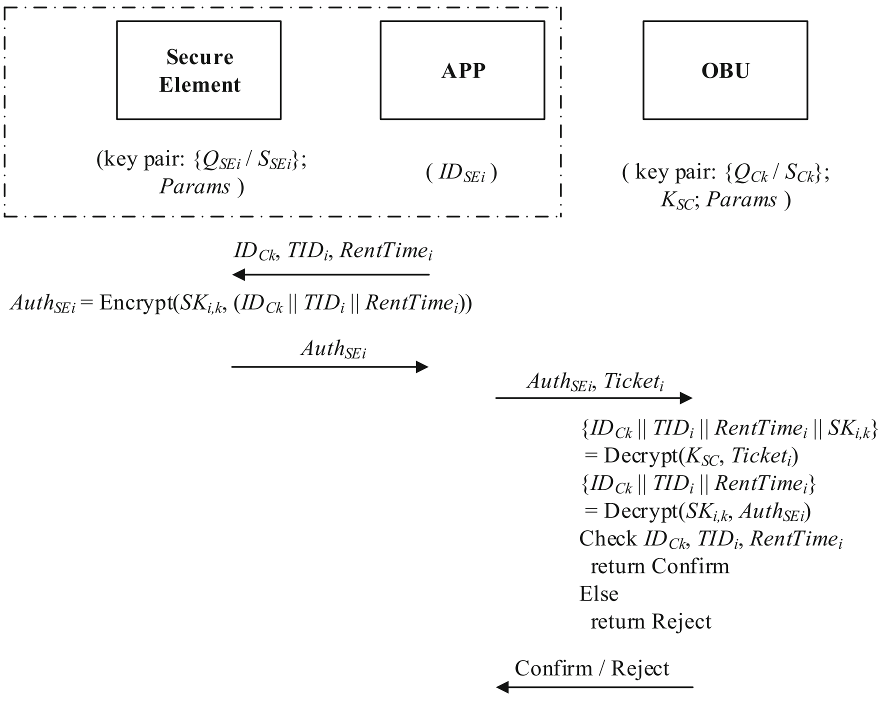
Secure Element
APP
OBU
(key pair: {Q_SEi / S_SEi}; Params)
(ID_SEi)
(key pair: {Q_Ck / S_Ck}; K_SC; Params)
ID_Ck, TID_i, RentTime_i
Auth_SEi = Encrypt(SK_i,k, (ID_Ck || TID_i || RentTime_i))
Auth_SEi
Auth_SEi, Ticket_i
{ID_Ck || TID_i || RentTime_i || SK_i,k} = Decrypt(K_SC, Ticket_i)
{ID_Ck || TID_i || RentTime_i} = Decrypt(SK_i,k, Auth_SEi)
Check ID_Ck, TID_i, RentTime_i
return Confirm
Else
return Reject
Confirm / Reject

Fig. 5 Protocol of ticket verification

3.3.3 Vehicle ticket verification protocol

Figure 5 shows the detailed steps of our ticket verification.

Step 1 The user sends $\{ID_{Ck}, TID_i, RentTime_i\}$ to SE through APP.
Step 2 SE uses $SK_{i,T}$ to encrypt the vehicle info and then returns the ciphertext $Auth_{SEi}$ to APP.
Step 3 APP forwards $\{Auth_{SEi}, Ticket_i\}$ to OBU.
Step 4 OBU's actions include the following:
(a) OBU uses K_{SC} to decrypt $Ticket_i$; obtains $\{ID_{Ck}||TID_i||RentTime_i||SK_{i,k}\}$; and uses $SK_{i,k}$ to decrypt $Auth_{SEi}$. Then it obtains $\{ID_{Ck}||TID_i||RentTime_i\}$ and verifies it. If the verification fails, OBU ends the session.
(b) OBU stores $\{TID_i, RentTime_i, SK_{i,k}\}$ and authorizes the user to control the vehicle.

3.3.4 Vehicle control protocol

Figure 6 shows the detailed steps of vehicle control.

Step 1 APP sends user's command to SE.
Step 2 SE generates a nonce $Nonce \in \{0,1\}^n$, uses $SK_{i,k}$ to encrypt $\{Command||Nonce\}$ as CMD and returns CMD to APP.

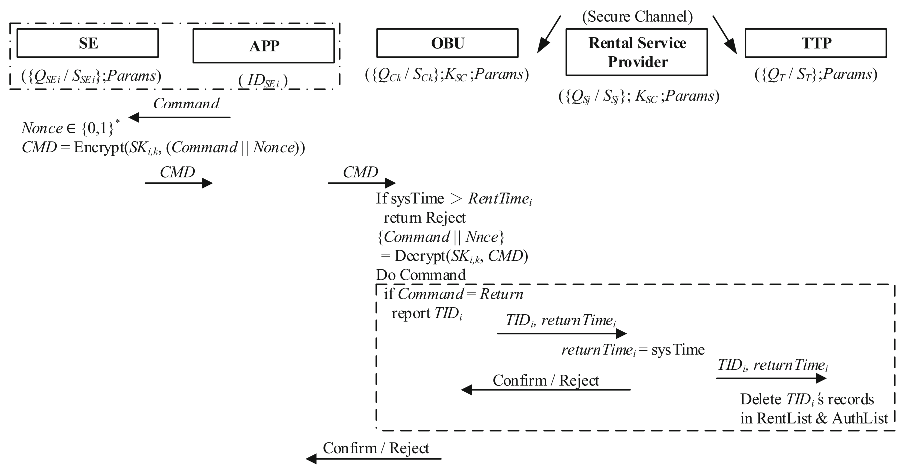

Fig. 6 Protocol of vehicle control and return

Step 3 APP forwards *CMD* to OBU.

Step 4 OBU's actions include the following:

(a) OBU checks users' rental period. If the period is reached, OBU rejects user's command.

(b) OBU uses $SK_{i,k}$ to decrypt *CMD* and then executes *Command.*

(c) If the command is *Return*, OBU sends TID_i and $returnTime_i$ to its rental service provider.

Step 5 The rental service provider stores current system time as the return time and sends $returnTime_i$ to TTP.

Step 6 TTP clears its AuthList and RentList, and the user's rental session is finalized.

4 Performance evaluation and security analysis

4.1 Performance evaluation and comparison

This section evaluates the performance of our proposed protocol. Detailed analysis of our computational loads are described as follows. We use E to denote a symmetric key cryptographic function; H a hash operation; nGx n multi-exponentiations in group Gx; Ra pseudo random number. The computational loads are listed in Table 7. As the table shows, most of the computational loads are required for our license issuing stage and the rest stages only run symmetric encryption and generate random numbers.

On the NFC interface, the significant increase of computational loads occurs at the ticket verification stage, in which both $Auth_{SEi}$ and $Ticket_i$ are transmitted over the air. OBU uses K_{SC} to decrypt $Ticket_i$; $Ticket_i$ contains encrypted $\{ID_{Ck} || TID_i || RentTime_i || SK_{i,k}\}$; and $Auth_{SEi}$ contains encrypted $\{ID_{Ck} || TID_i || RentTime_i\}$. If we

Table 7 Operation costs

Protocol	SE	APP	Service provider	OBU	TTP
License issuing	R+$2E$+$6H$+$3P$ +$(r+k)G_1$	R+$2H$+$2P$+rG_1			$2R$+$5H$+$3P$+$2rG_1$
Ticket issuing	$2E$		$3E+R$		E
Ticket verification	E			$2E$	
Vehicle control	E		E	$2E$	E

Table 8 Comparison with other e-rental schemes

Schemes	Functions					
	Anonymity	Unlinkability	Traceability	Mutual Auth.	Flexibility	Anonymous payment
Slammanig and Rass [3]	✓	✗	✓	✓	✓	✗
Wang et al. [4]	✓	✓	✗	✓	✓	✓
Chen et al. [5]	✓	✓	✓	✓	✓	✓
Blanton [6]	✓	✓	✗	✓	✓	✗
Vasco et al. [7]	✓	✓	Partial	✓	✓	✓
Lee et al. [8]	✓	✓	✗	✓	✓	✗
Our protocol	✓	✓	✓	✓	✓	✓

assume the session key is 128 bits and others are all 32-bit integers, the payload in air transmission is 40 bytes plus packet headers.

Tables 8 compares the functionality of our paper and related schemes. As shown in the table, all the protocols meet the requirement of anonymity, flexibility, and mutual authentication. As for unlinkability, Slamanig and Rass's protocol [3] allows a service provider to issue multiple tokens to a user. Despite their user's anonymity, the service provider can still find the relation between the user's tokens. As for traceability, only Slamanig and Rass's [3], Chen et al.'s [5] and our methods allow TTP to disclose users' identity. The TTP in Vasco et al.'s [7] protocol can only store partial information of a user's identity. Consequently, their service provider cannot request a user's real identity from the TTP. As for anonymous payment, Slamanig and Rass's [3], Blanton's [6], and Lee et al.'s [8] methods do not implement this function.

The main difference between our e-rental car services and other online rental services lies in the legal issues and the value of the hired products. As driving requires a license and a car is a real object which is much more expensive than most things for rent online, an anonymous e-rental protocol has to take traffic regulations and rental agents' risk evaluation into consideration. Although we guarantee a customer's anonymity, we still have to allow the rental service providers to trace the identity of their renters under some special circumstances.

4.2 Security analysis

Anonymity

- At the stage of anonymous license issuing, only the TTP knows users' real identity ID_{SEi}.
- At the stage of vehicle ticket issuing, if users want to rent vehicles, they only need to send the anonymous license with a specific vehicle information to a rental service provider. The service provider does not know the user's real identity.
- At the stage of vehicle ticket verification, the OBU in the vehicle does not know the user's identity because the content of the ticket does not include the users' real identity.
- Throughout the rental process, only TTP has the user's real identity and anonymity is guaranteed in our protocol.

Confidentiality
Messages among user, TTP, rental service provider, and OBU are all protected by session keys SK_{SETi}, SK_{SESi}, SK_{SECi}, respectively. Consequently, adversaries cannot decrypt the ciphertexts to access the information.

Replay attack
At the stage of anonymous license issuing, the messages between user and TTP are encrypted with nonces such as $\{T_{SEi}, Nonce_{SEi}\}$ and $\{T_T, Nonce_T\}$. Particularly, SE needs T_{SEi} and T_T to generate session key $SK_{i,T}$ Therefore if attackers replay earlier messages, they cannot pass our verification.

Considering if an attacker replays an expired anonymous license at the stage of issuing a vehicle ticket, because the rental service provider will verify the license with TTP, TTP will find that the license does not exist in its AuthList. If a license is replayed to a rental service provider within its rental period, TTP will find TID_i is already stored in the RentList.

At the stage of ticket verification, if an attacker replays a vehicle ticket to an OBU within its rental period, the OBU will decline the request.

Man-in-the-Middle (MITM) attack
At the stage of anonymous license issuing, APP uses TTP's public key Q_T for encryption and then sends $\{ID_{SEi}, C\}$ to TTP. Even though adversaries intercept the message, they do not have TTP's private key to decrypt the ciphertext C. Thus, they cannot acquire $\{ID_{SEi}, T_{SEi}, Nonce_{SEi}\}$, and they cannot compute $Nonce'_{SEi}$ and the session key $SK_{i,T}$. In other words, since adversaries do not have users' private key either, they are unable to decrypt C', sent from TTP to users. Again, they cannot compute $Nonce'_T$ and $SK_{i,T}$. So, here adversaries are unable to launch MITM attacks by playing a user or a TTP.

At the stage of issuing a vehicle ticket, users' rental info is encrypted with $SK_{i,T}$. Because attackers do not have this key, they cannot decrypt the message and find out the service provider information and the vehicle information. Therefore, they cannot act as a middle man and launch an MITM attack.

At the stage of vehicle ticket verification, because $\{ID_{Ck} || TID_i || RentTime_i\}$ is encrypted with $SK_{i,k}$, attackers are unable to find which vehicle has been rented by the user without this key. They cannot launch an MITM attack under such a condition.

Backward secrecy
Our protocol guarantees backward secrecy.
At the stage of anonymous license issuing, even though we assume an attacker has S_{SEi}, S_T and s, he cannot compute a and b by using $T_{SEi} = aP$ and $T_T = bP$. Therefore, he cannot compute the session key $K_{SEi} = e(T_T + Q_T, aP_{pub} + S_{SEi}) \quad e(Q_T, S_{SEi}) = e(T_{SEi} + Q_{SEi}, bP_{pub} + S_T) \quad e(Q_{SEi}, S_T) = K_T$, where $K_{SEi} = K_T = e(P, P)^{abs} \quad e(P, Q_T)^{as} \quad e(Q_{SEi}, P)^{bs} \quad e(Q_{SEi}, Q_T)^{2s}$. Even though attackers may eavesdrop on $T_{SEi} = aP$ and $T_T = bP$, it is still nearly impossible to compute abP from aP and bP. Even if an attacker gets the key $SK_{i,T}$, he cannot deduce the session keys of previous sessions.

At the stages of ticket issuing and ticket verification, the session keys $SK_{i,j}$ and $SK_{i,k}$ are randomly generated by a TTP and a rental service provider, respectively. Even though attackers can obtain each role's private keys, they cannot break current session keys or decrypt messages of previous sessions.

Forward secrecy
Our protocol guarantees forward secrecy.

At the stage of anonymous license issuing, if an attacker obtains S_{SEi}, S_T and s, he cannot predict the random numbers a and b to compute the following user TTP session keys.

At the stages of vehicle ticket verification, an attacker cannot deduce future session keys from current ones and each role's private key, because $SK_{i,j}$ and $SK_{i,k}$ are randomly generated by a TTP and a rental service provider, respectively.

5 Proof

We use GNY proof [23] to examine the security of our protocol. Our security proof consists of four parts: (1) notations used in the proof; (2) initial assumptions; (3) goals at each stage; (4) detailed steps of our security proof. Table 9 details each step of our proof.

Table 9 GNY proof of our protocol

Notations	
TTP	Trusted third party
U_i	User (including secure element SE_i and a car-rental application APP)
S_j	Rental service provider
C_k	OBU (vehicle)
$\{X\}_K$	Message X is encrypted with symmetric key K
$P \triangleleft X$	P receives message X
$P \ni X$	P owns X
$P \mid\equiv \#(X)$	P believes the freshness of X

Table 9 continued

$P\mid\equiv\phi(X)$	P believes the content of X is recognizable
$P\mid\equiv P \overset{S}{\longleftrightarrow} Q$	s is a shared secret and appears between P and Q only
Initial assumption	
TTP	U_i (user)
$TTP \ni ID_T, b, Nonce_T, TID_i, SK_{i,j}, K_{LICi}, ExpTime_i$	$U_i \ni ID_{SEi}, a, Nonce_{SEi}, Nonce, K_{SEi}$ $U_i\mid\equiv\mid\xrightarrow{K_T} TTP$
$TTP\mid\equiv\mid\xrightarrow{K_{SEi}} U_i$	$U_i\mid\equiv\mid TTP\mid\Rightarrow SE_i \overset{SK_{i,j}}{\longleftrightarrow} S_j$
S_j (rental service provider)	C_k (OBU/vehicle)
$S_j \ni ID_{Sj}, SK_{i,k}$	$C_k \ni ID_{Ck}$
$S_j\mid\equiv\#(SK_{i,k})$	$C_k\mid\equiv C_k \overset{K_{SC}}{\longleftrightarrow} S_j$
$S_j\mid\equiv S_j \overset{K_{SC}}{\longleftrightarrow} C_k$	$C_k\mid\equiv S_j\mid\Rightarrow C_k \overset{SK_{i,k}}{\longleftrightarrow} U_i$
Goal	
Anonymous license issuing protocol	
$TTP\mid\equiv\#\left(ID_{SEi}, \{ID_{SEi}, T_{SEi}, Nonce_{SEi}\}_{+K_T}\right)$ $U_i\mid\equiv\#\left(ID_T, \{ID_T, T_T, Nonce_T, f(Nonce_{SEi})\}_{+K_{SEi}}\right)$ $U_i\mid\equiv U_i \overset{SK_{i,T}}{\longleftrightarrow} TTP$ $U_i\mid\equiv\#(TID_i, ExpTime_i, SK_{i,j})$	Mutual authentication between TTP and the user. User obtains the session key $SK_{i,T}$, and the anonymous license contains TID_i, $ExpTime_i$ and $SK_{i,j}$
Vehicle ticket issuing protocol	
$S_j\mid\equiv TTP\mid\sim\left\{TID_i, ExpTime_i, SK_{i,j}\right\}_{K_{LICi}}$ $S_j\mid\equiv S_j \overset{SK_{i,j}}{\longleftrightarrow} U_i$ $U_i\mid\equiv U_i \overset{SK_{i,j}}{\longleftrightarrow} S_j$ $U_i\equiv\#\left\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\right\}_{K_{SC}}$	Mutual authentication between the rental service provider and the user. The user obtains the session key $SK_{i,}$ and vehicle ticket ID_{Ck}, TID_i, $RentTime_i$, and $SK_{i,k}$
Vehicle ticket verification protocol	
$C_k\mid\equiv S_j\mid\sim\left\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\right\}_{K_{SC}}$	OBU verifies that the ticket is issued by its own rental service provider and then obtains the session key $SK_{i,k}$
$U_i\mid\equiv U_i \overset{SK_{i,k}}{\longleftrightarrow} C_k$	OBU shares a session key with the user
Vehicle control protocol	
1. $C_k\mid\equiv U_i\mid\sim\{CMD\}_{SK_{i,k}}$	OBU verifies that the commands come from the user
Proof	
Anonymous license issuing protocol	
$Msg1.1: TTP \triangleleft *ID_{SEi}, *\{ID_{SEi}, T_{SEi}, Nonce_{SEi}\}_{K_T}$ $TTP \ni *ID_{SEi}, *\{ID_{SEi}, T_{SEi}, Nonce_{SEi}\}$ $TTP \ni ID_{SEi}, T_{SEi}, Nonce_{SEi}$ $TTP\mid\equiv\#\left(ID_{SEi}, \{ID_{SEi}, T_{SEi}, Nonce_{SEi}\}_{K_T}\right)$	We believe the freshness of nonce $Nonce_{SEi}$ can guarantee that ID_{SEi} and T_{SEi} are not replayed
$Msg1.3: TTP \triangleleft *\{f(Nonce_T)\}_{SK_{i,T}}$ $TTP \triangleleft *\{f(Nonce_T)\}$ $TTP \ni f(Nonce_T)$ $TTP\mid\equiv\#(f(Nonce_T))$ $TTP\mid\equiv U_i\mid\equiv TTP \overset{SK_{i,T}}{\longleftrightarrow} U_i$	TTP checks the validity of user-TTP shared session key $SK_{i,T}$

Table 9 continued

$Msg1.4: U_i \triangleleft *\{TID_i, ExpTime_i, SK_{i,j}\}_{SK_{i,T}},$ $*\{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$ $U_i \ni *\{TID_i, ExpTime_i, SK_{i,j}\},$ $*\{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$ $U_i \ni TID_i$ $U_i \ni ExpTime_i$ $U_i \ni SK_{i,j}$ $U_i \vert\equiv \#\{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$	TTP issues an anonymous license to user
Vehicle ticket issuing protocol	
$Msg2.1: S_j \triangleleft *TID_i, *ExpTime_i,$ $*\{TID_i, RentTime_i, ID_{Sj}, ID_{Ck}, \text{Pr}ice_{Ck}\}_{SK_{i,T}},$ $*\{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$ $S_j \ni TID_i$ $S_j \ni ExpTime_i$ $S_j \vert\equiv \#\{TID_i, RentTime_i, ID_{Sj}, ID_{Ck},$ $\text{Pr}ice_{Ck}\}_{SK_{i,T}}$ $S_j \vert\equiv \#\{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$	Rental service provider receives user's vehicle info and then sends it to TTP for verification
$Msg2.2: TTP \triangleleft *ID_{Sj}, TID_i, Exp\,Time_i,$ $*\{TID_i, Rent\,Time_i, ID_{Sj}, ID_{Ck}, \text{Pr}ice_{Ck}\}\, SK_{i,T}$ $TTP \ni ID_{Sj}, ID_{Ck}, TID_i, \text{Pr}ice_{Ck}, ExpTime_i,$ $RentTime_i$ $TTP \vert\equiv S_j \vert\equiv S_j \overset{TID_i, ID_{Sj}}{\longleftrightarrow} SE_i$ $TTP \vert\equiv S_j \overset{TID_i, ID_{Sj}}{\longleftrightarrow} SE_i$ $TTP \vert\equiv \#\{TID_i, ID_{Sj}\}$	TTP receives user's vehicle info and verifies that the user rents a car from the company
$Msg2.3: S_j \triangleleft *K_{LICi}, *RentTime_i, *ID_{Ck},$ $*\text{Pr}ice_{Ck}$ $S_j \ni K_{LICi}$ $S_j \ni RentTime_i$ $S_j \ni ID_{Ck}$ $S_j \ni \text{Pr}ice_{Ck}$ $S_j \triangleleft *\{TID_i, ExpTime_i, SK_{i,j}\}$ $S_j \vert\equiv TTP \vert\sim \{TID_i, ExpTime_i, SK_{i,j}\}_{K_{LICi}}$ $S_j \ni TID_i$ $S_j \ni ExpTime_i$ $S_j \ni SK_{i,j}$ $S_j \vert\equiv TTP \vert\equiv S_j \overset{SK_{i,j}}{\longleftrightarrow} U_i$ $S_j \vert\equiv S_j \overset{SK_{i,j}}{\longleftrightarrow} U_i$	Rental service provider verifies the validity of the anonymous license and then obtains session key $SK_{i,j}$
$Msg2.4: U_i \triangleleft *\{ID_{Ck}, TID_i, RentTime_i,$ $SK_{i,k}\}_{SK_{i,j}},$ $*\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\}_{K_{SC}}$ $U_i \vert\equiv TTP \vert\equiv U_i \overset{SK_{i,j}}{\longleftrightarrow} S_j$ $U_i \vert\equiv U_i \overset{SK_{i,j}}{\longleftrightarrow} S_j$ $U_i \triangleleft *\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\},$ $*\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\}_{K_{SC}}$ $U_i \ni ID_{Ck}$ $U_i \ni TID_i$ $U_i \ni RentTime_i$ $U_i \ni SK_{i,k}$ $U_i \vert\equiv \#\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\}_{K_{SC}}$	Rental service provider issues a vehicle ticket to the user. The ticket contains ID_{Ck}, TID_i, $RentTime_i$, and $SK_{i,k}$

Table 9 continued

Vehicle ticket verification protocol	
$Msg3.1: C_k \lhd *\{ID_{Ck}, TID_i, RentTime_i\}_{SK_{i,k}}, *\left\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\right\}_{K_{SC}}$ $C_k \lhd *\left\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\right\}, *\{ID_{Ck}, TID_i, RentTime_i\}_{SK_{i,k}}$ $C_k \ni ID_{Ck}, TID_i, RentTime_i, SK_{i,k}$ $C_k\mid\equiv \{ID_{Ck}, TID_i, RentTime_i\}_{SK_{i,k}}$ $C_k\mid\equiv S_j\mid\sim \left\{ID_{Ck}, TID_i, RentTime_i, SK_{i,k}\right\}_{K_{SC}}$ $C_k\mid\equiv S_j\mid\equiv C_k \overset{SK_{i,k}}{\longleftrightarrow} U_i$ $C_k\mid\equiv C_k \overset{SK_{i,k}}{\longleftrightarrow} U_i$	OBU verifies the validity of the ticket, obtains the session key $SK_{i,k}$, and authorizes the user to control the vehicle
Vehicle control protocol	
$Msg4.1: C_k \lhd *\{\{CMD\}, Nonce\}_{SK_{i,k}}$ $C_k \ni *\{\{CMD\}, Nonce\}$ $C_k \ni \{CMD\}$ $C_k \ni Nonce$ $C_k\mid\equiv U_i\mid\sim \{\{CMD\}, Nonce\}_{SK_{i,k}}$	OBU verifies that the user shares the session key $SK_{i,k}$, and that the commands are sent by the user
$Msg4.2: S_j \lhd *TID_i, *\left\{TID_i, ID_{Sj}\right\}_{SK_{i,T}}$ $S_j \lhd *TID_i, *\left\{TID_i, ID_{Sj}\right\}_{SK_{i,T}}$ $S_j \ni TID_i$ $S_j \ni SK_{i,Ti}$ $S_j\mid\equiv \#\left\{TID_i, ID_{Sj}\right\}_{SK_{i,T}}$	TTP verifies that the user has returned the car to its rental service provider

6 Conclusion

In this paper, we propose an anonymous e-rental protocol that features ID-based cryptography and NFC phone technology. A user can go through all the rental procedure on his NFC phone, and only TTP knows his real identity. Even his rental service provider and attackers cannot breach his privacy, abuse his personal data, or analyze his rental habits. But if there are consumer disputes or traffic accidents, the service provider is allowed to take a legal warrant to request the TTP to reveal the user's real identity.

With the integration of bilinear pairing ID-based cryptography, our protocol is secure and efficient in the mobile environment. All the secrets and users' sensitive information are stored in NFC phones' secure elements. Compared with related schemes, our protocol can prevent attacks such as replay attacks and MITM attacks, and guarantees anonymity, confidentiality and forward/backward secrecy.

References

1. Zipcar (2013) http://www.zipcar.com/ Retrieved date 16 July 2013
2. Car2go (2013) http://www.car2go.com/ Retrieved date 16 July 2013
3. Slamanig D, Rass S (2010) Anonymous but authorized transactions supporting selective traceability. In: 5th International conference on security and cryptography-SECRYPT, pp 132–141
4. Wang S, Chen Z, Wang X (2008) A new certificateless electronic cash scheme with multiple banks based on group signatures. In: IEEE international symposium on electronic commerce and security

5. Chen Y, Chou JS, Sun HM, Cho MH (2011) A novel electronic cash system with trustee-based anonymity revocation from pairing. Electron Commerc Res Appl 10(6):673–682
6. Blanton M (2008) Online subscriptions with anonymous access. In: Proceedings of the 2008 ACM symposium on information, computer and communications, security, pp 217–227
7. Vasco M, Heidarvand S, Villar JL (2012) Flexible Anonymous Subscription Schemes. e-Business Telecommun 222:203–219
8. Lee MZ, Dunn AM, Katz J, Waters B, Witchel E (2013) Anon-Pass: practical anonymous subscriptions. In: 2013 IEEE symposium on security and privacy, pp 319–333
9. Meffert D (2009) Bilinear Pairings in Cryptography. Radboud Universiteit Nijmegen, Master thesis
10. Shamir A (1984) Identity-based cryptosystems and signature schemes. Advances in cryptology-crypto' 84. In: Lecture notes in computer science, vol 196. Springer, Berlin, pp 47–53
11. Jonsson J, Kaliski B (2003) Public-key cryptography standards (PKCS) #1: RSA cryptography specifications version 2.1. RFC 3447 (Informational), Internet Engineering Task Force.
12. Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48:203–209
13. Miller V (1986) Use of elliptic curves in cryptography. In: Advances in cryptology-crypto '85, pp 417–426
14. Boneh D, Franklin M (2001) Identity-based encryption from the Weil pairing. SIAM J Comput 2139:586–615
15. International Organization for Standardization (2007) ISO/IEC 18092–4. Information technology-Telecommunications and information exchange between systems-Near Field Communication-Interface and Protocol (NFCIP-1)
16. Madlmayr G, Langer J, Scharinger J (2008) Near Field Communication based Mobile Payment System. Proc. Mobile und Ubiquitäre Informationssysteme-Technologien, Prozesse, Marktfähigkeit, pp 81–93
17. Dutta R, Barua R, Sarkar P (2004), Pairing-based cryptographic protocols: a survey. Cryptology ePrint Archive, Report 2004/064
18. Zhang J, Wu Z, Li Y (2011) An improved identity-based authenticated key agreement protocol using pairings. In: International conference on computer science and network technology
19. Damme GV, Wouters K, Karahan H, Preneel B (2009) Offline NFC payments with electronic vouchers. In: ACM workshop on networking, systems, and applications on mobile handhelds-MobiHeld, pp 25–30
20. Jao D (2010) Elliptic curve cryptography. In: Handbook of information and communication security, pp 35–57
21. Lauter K (2004) The advantages of elliptic curve cryptography for wireless security. IEEE Wirel Commun pp 62–67
22. Menezes A, Okamoto T, Vanston S (1993), Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans Inf Theory, pp 1639–1646
23. Gong L, Needham R, Yahalom R (1990) Reasoning about belief in cryptographic protocols. In: IEEE computer society symposium on research in security and privacy, pp 234–248