Privacy-Preserving Electronic Ticket Scheme with Attribute-Based Credentials


**Published in:**
IEEE Transactions on Dependable and Secure Computing

**Document Version:**
Peer reviewed version
Privacy-Preserving Electronic Ticket Scheme with Attribute-Based Credentials

Jinguang Han, Senior Member, IEEE, Liqun Chen, Senior Member, IEEE, Steve Schneider, Helen Treharne and Stephan Wesemeyer

Abstract—Users accessing services are often required to provide personal information, for example, age, profession and location, in order to satisfy access policies. This personal information is evident in the application of e-ticketing where discounted access is granted to visitor attractions or transport services if users satisfy policies related to their age or disability or other defined over attributes. We propose a privacy-preserving electronic ticket scheme using attribute-based credentials to protect users’ privacy. The benefit of our scheme is that the attributes of a user are certified by a trusted third party so that the scheme can provide assurances to a seller that a user’s attributes are valid. The scheme makes the following contributions: (1) users can buy different tickets from ticket sellers without releasing their exact attributes; (2) two tickets of the same user cannot be linked; (3) a ticket cannot be transferred to another user; (4) a ticket cannot be double spent. The novelty of our scheme is to enable users to convince ticket sellers that their attributes satisfy the ticket policies and buy discounted tickets anonymously. This is a step towards identifying an e-ticketing scheme that captures user privacy requirements in transport services. The security of our scheme is proved and reduced to a well-known complexity assumption. The scheme is also implemented and its performance is empirically evaluated.

Index Terms—Anonymity, Attribute-based Credentials, Privacy-enhanced Authentication, Electronic Ticket

1 INTRODUCTION

Due to their flexibility and portability, electronic ticket (e-ticket) systems have been extensively investigated by both industry [1], [2] and the academic research communities [3], [4], [5]. E-tickets are attractive to transport operators as well as customers because they can reduce paper costs (tickets can be stored on a hand-held device) and improve customer experience (tickets can be purchased and delivered any time and anywhere). However, the use of e-tickets also raises many questions regarding the privacy of its users due to the possibility of linking different e-ticket transactions to a particular user—in contrast to anonymous paper-based tickets—and thus potentially revealing private information, e.g., working patterns, likely places of work, etc.

Customers are increasingly aware of privacy issues especially in the light of the newly introduced general data protection regulations (GDPR) [6]. One way to address this is through anonymous authentication which enables users to authenticate without revealing their identities and this approach has been used to protect a user’s privacy in many privacy-preserving e-ticket schemes [3], [7], [8], [9], [10], [11]. However, many of these schemes were not formally proven to be secure. Notable exceptions are those proposed by Arfaoui et al. [8] and Rupp et al. [12]. Arfaoui et al. [8] formally defined their security models for e-ticket schemes, including unforgeability, unlinkability and non-repudiation, but the authors only provided a very high-level proof. Rupp et al. [12] formalised their security models of privacy-preserving pre-payments with refunds schemes including transportation authority security and user privacy but the security proof of their scheme was again at a high level. Another requirement of a realistic e-ticket systems is the support for different tickets based on a user’s attributes (e.g., age, location, disability, profession, etc.), i.e. to offer discounts for, say, students or disabled passengers. However, if not implemented carefully, there is a risk that such a ticket system reveals more information about a user than necessary when purchasing or validating tickets. For example, a student buying a discounted student ticket may end up revealing the university at which she is enrolled and, depending on the student card, even her birthday neither of which is relevant to obtaining the student discount. The minimum proof required is that she can demonstrate that she is a legitimate student. Similarly, a disabled passenger might need to reveal more details about his disability to the ticket seller or verifier than necessary for purchasing or verifying a ticket. Gudymenko [10] and Kerschbaum et al. [11] addressed this issue, but their schemes were not proven formally.

Transport operators are naturally concerned about fraudulent use of e-tickets due to the ease with which they can be copied. Double spend or more generally overspend detection, i.e. the process of determining whether a ticket has been used too many times, is therefore also an important feature that an e-ticket scheme should support.

To address the above requirements, this paper proposes a new privacy-preserving e-ticket scheme using attribute-
based credentials which supports issuing different tickets depending on a user’s attributes. Our scheme protects an honest user’s privacy while allowing for the de-anonymisation of users who try to use their tickets more than once (double spend detection). It is also a general e-ticket system and can be used in various application scenarios including:

- mobility as a service transport tickets (e.g. rail, bus, etc.) where age, disability, profession, affiliation, etc. might determine the prices of tickets;
- one-off token for Internet services (e.g. print service, download service for multimedia, etc.) where age, affiliation, membership might determine the service/access level;
- e-voting where age, nationality, voting district, etc. might determine the voting ballot that should be issued;
- event tickets (e.g. concert, tourist attractions, conferences, etc.) where age, affiliation, disability, etc. might determine the ticket price/access rights.

1.1 Contributions
In this paper, we propose a new attribute-based e-ticket scheme. The main contributions of our scheme are: (1) **Attribute-based Ticketing**: users can buy different tickets depending on their certified attributes without releasing their exact details; (2) **Unlinkability**: two tickets of the same user cannot been linked; (3) **Untransferability**: a ticket can only be used by the ticket holder and cannot be transferred to another user; (4) **Double Spend Detection**: a ticket cannot be double spent and the identities of users who try to double spend tickets can be revealed.

The novelty of our scheme is to enable users to convince ticket sellers that their attributes satisfy the ticket policies and buy discounted tickets anonymously. Our scheme thus offers a natural as well as flexible way of representing user attributes, e.g. to obtain an age based discount, a user would expect to prove that her age is in a certain range, while for a disability discount, she would want to demonstrate her impairment is contained within the set of recognised disabilities. Furthermore, a user’s attributes are additionally certified by a trusted third party thereby allowing a user’s claimed attributes to also be verified.

The theoretical contribution is that the security of the proposed scheme is formally proven and reduced to a well-known complexity assumption. The scheme is also implemented and performance timings are given.

1.2 Related Work
Mut-Puigserver et al. [4] surveyed numerous e-ticket systems and summarised their various functional requirements (e.g. expiry date, portability, flexibility, etc.) and security requirements (e.g. integrity, authentication, fairness, non-overspending, anonymity, transferability, unlinkability, etc.). E-ticket schemes are classified into different types: transferable tickets [5], untransferable tickets [3], single-use tickets [6], [7], [8], [14]. Our scheme falls into the untransferable, single-use tickets categories while providing anonymity, unlinkability, non-overspending and flexibility.

We now compare our scheme with a number of other schemes. In these schemes, blind signatures [15], group signatures [16], anonymous credentials [17] and pseudonyms [16], [18] were used to protect user privacy.

**E-Ticket Schemes from Blind Signatures.** In a blind signature scheme, a user can obtain a signature on a message without the signer knowing the content. Based on the blind signature scheme proposed by Chaum [15], Fan and Lei [19] proposed an e-ticket system for voting in which each voter can vote in different elections using only one ticket. Song and Korba [9] proposed an e-ticket scheme to protect users’ privacy and provide non-repudiation in pay-TV systems. Quercia and Hailes [20] proposed an e-ticket scheme for mobile transactions using Chaum’s blind signature scheme [15] to generate both limited-use and unlimited-use tickets. Rupp et al. [12], [13] proposed privacy-preserving pre-payments with refunds schemes derived from Chaum’s scheme [22] and Boneh et al.’s short signature scheme [23].

In their scheme, trip authorisation tokens were generated using Chaum’s blind signatures, while Boneh et al.’s short signature scheme was used to implement the privacy-preserving aggregation of refunds. Milutinovic et al. [3] proposed an e-ticket scheme which combines the partial blind signature scheme proposed by Abe et al. [24], Pedersen’s secret sharing commitment scheme [25] and Camenisch et al.’s anonymous credential scheme [26] to protect user privacy. All these schemes can protect user privacy and provide ticket unlinkability, but, unlike our scheme, they do not support de-anonymisation after double spending nor ticket untransferability.

**E-Ticket Schemes from Group Signatures.** A group signature enables a user to sign a message on behalf of the group without exposing his identity, while the group manager can re-lease the identity of the real signer. Nakanishi et al. [27] proposed an electronic coupon (e-coupon) scheme where the group signature scheme [28] was used to provide anonymity and unlinkability. Vives-Guasch [29] proposed an automatic fare collection (AFC) system in which the group signature scheme proposed by Boneh et al. [30] was used to provide unlinkability and revocable anonymity. These schemes can implement anonymity, de-anonymity, ticket unlinkability and ticket untransferability, but, unlike our scheme, they do not support privacy-preserving attribute-based ticketing.

While Gudymenko in [10] addressed user privacy as well as differently priced tickets in his e-ticket scheme and used group signatures to make tickets unlinkable, no formal security models and security proofs were presented.

**E-Ticket Schemes from Anonymous Credentials.** In an anonymous credential scheme, a user can prove to a verifier that she has obtained a credential without releasing any other information. Heydt-Benjamin et al. [7] used anonymous credentials, e-cash and proxy re-encryption schemes to enhance the security and privacy of their public transport e-ticket systems. Arfaoui et al. [8] modified the signature scheme proposed by Boneh et al. in [31] to eliminate expensive pairing operations in the verification phase, and then proposed a privacy-preserving near field communication (NFC) mobile ticket (m-ticket) system by combining their modified signature with the anonymous credential scheme proposed by Camenisch et al. [32]. In their scheme, a user can anonymously use an m-ticket at most k times, otherwise the user
is revoked by the revocation authority. These schemes can implement anonymity, ticket unlinkability as well as ticket untransferability, but, unlike our scheme, do not support privacy-preserving attribute-based ticketing. Additionally, the security of these schemes was not formally proven.

**E-Ticket Schemes from Pseudonyms.** Pseudonyms allow a user to interact with multiple organisations anonymously and potentially without linkability. Fujimura and Nakajima [33] proposed a general-purpose e-ticket framework where anonymity was achieved by using pseudonym schemes [34, 35]. Jorns et al. [36] proposed a pseudonym scheme which could be implemented on constrained devices, and then used it to protect users’ privacy in e-ticket systems. Kunzle and Schmidt [37] proposed a scheme to generate pseudonym tickets by using the identities embedded in attestation identity keys (AIKs) certified by the privacy certificate authority (PCA). Vives-Guasch et al. [38] proposed a light-weight e-ticket scheme using pseudonyms which also addressed culpability (i.e. a service provider cannot falsely accuse a user of having overspent her ticket, and the user is able to demonstrate that she has already validated the ticket before using it) and reusability (i.e. a ticket can be used a predefined number of times). In [38], pseudonyms were used to provide unlinkability of users’ transactions. Kerschbaum et al. [11] considered the privacy-preserving billing issue in e-ticket schemes and applied pseudonyms to provide unlinkability of user transactions. These schemes can implement anonymity, ticket unlinkability as well as ticket untransferability, but, unlike our scheme, they do not support privacy-preserving attribute-based ticketing. Furthermore, the security of these schemes was not formally proven.

**E-Tickets from Special Devices.** There are other e-ticket schemes designed around special devices, including personal trusted device (PTD) [39], trusted platform module (TPM) [37], mobile handsets [40], etc. Unlike our scheme, these schemes require special devices and do not enable de-anonymisation after double spending a ticket nor do they support privacy-preserving attribute-based ticketing.

We compare our scheme with related schemes in Table 1 in terms of unlinkability, untransferability, double spend detection, de-anonymisation, attribute-based ticketing and security proof, where --- indicates that security was not considered by the authors of the respective schemes.

The European Telecommunications Standards Institute (ETSI) [41, 42] released two specifications on attribute-based encryption (ABE) which can be used to securely protect personal data and implement fine-grained access control. In an ABE scheme, a message is encrypted by using a set of attributes so that only the users whose attributes match those in the ciphertext can decrypt it and see the message. As specified in [42], ABE supports offline access control. However, in our scheme, a user can only authenticate to an online verifier. Furthermore, an issued credential enables a user to prove that she holds required attributes without revealing them.

### 1.3 Organisation

The remainder of this paper is organised as follows. In Section 2 the preliminaries used throughout this paper are described. The construction of our scheme is presented in Section 3. In Section 4 the performance of our scheme is evaluated. Section 5 presents the security proof of our scheme. Finally, Section 6 concludes this paper.

## 2 Preliminaries

In this section, the preliminaries used throughout this paper are introduced. The most important notation is summarised in Table 2. We define a function \( e(x) \) is negligible if for any \( k \in \mathbb{N} \), there exists an \( N_k \) such that \( e(y) \leq \frac{1}{y^k} \) for all \( y > N_k \).

### 2.1 Bilinear Groups

Let \( G_1, G_2 \) and \( G_r \) be cyclic group with prime order \( p \). A map \( e : G_1 \times G_2 \rightarrow G_r \) is a bilinear map if the following properties are satisfied [43]:

1. **Bilinearity.** For all \( g \in G_1 \), \( h \in G_2 \) and \( x, y \in \mathbb{Z}_p \), \( e(g^x, h^y) = e(g^x, h^y) = e(g, h)^{xy} \).
2. **Non-degeneration.** For all \( g \in G_1 \) and \( h \in G_2 \), \( e(g, h) \neq 1 \), where \( 1 \) is the identity element in \( G_r \).
3. **Computability.** For all \( g \in G_1 \) and \( h \in G_2 \), there exists an efficient algorithm to compute \( e(g, h) \).

Note that Galbraith, Paterson and Smart [44] classified pairings into three basic types: Type-I: \( G_1 = G_2 \); Type-II: \( G_1 \neq G_2 \) and there exists an efficient map \( \psi : G_1 \rightarrow G_1 \); Type-III: \( G_1 \neq G_2 \) but there is no efficient map between \( G_1 \) and \( G_2 \). Our scheme is based on the Type-I pairing which is used to construct the signature below.

In the case that \( G_1 = G_2 \), \( e \) is called symmetric bilinear map. Let \( BG(1^k) \rightarrow (e, p, G, G_r) \) be a symmetric bilinear group generator which takes as input a security parameter \( 1^k \) and outputs a bilinear group \( (e, p, G, G_r) \) with prime order \( p \) and \( e : G \times G \rightarrow G_r \).

### 2.2 Complexity Assumptions

**Definition 1.** (q-Strong Diffie-Hellman (SDH) Assumption) [31] Let \( BG(1^k) \rightarrow (e, p, G, G_r) \), \( g \) be a generator of \( G \) and \( x \in \mathbb{Z}_p \). We say that the \( q \)-strong Diffie-Hellman assumption holds on \( G \) if for all probabilistic polynomial time (PPT) adversary \( A \) given \( g, g^x, \ldots, g^{x^q} \) can output a pair \( (e, g^{x^q}) \) with negligible probability, namely \( Adv^{q-SDH}_A = \Pr[\mathcal{A}(g, g^x, \ldots, g^{x^q}) \rightarrow (e, g^{x^q})] \leq \epsilon(k) \), where \( e \in \mathbb{Z}_p \).

The security of the following two signatures used in our scheme and thus our overall security can be shown to reduce to this complexity assumption.

### 2.3 Zero-Knowledge Proof

In this paper, we use zero-knowledge proof of knowledge protocols to prove knowledge of statements about discrete logarithms [45], including discrete logarithm, equality, product, disjunction and conjunction. We follow the notation proposed in [28] and formalised in [46]. By \( \text{PoK} \{ (\alpha, \beta, \gamma) : A = g^\alpha h^\beta \land \tilde{A} = \tilde{g}^\alpha \tilde{h}^\beta \} \), we denote a zero-knowledge proof of knowledge of \( \alpha, \beta \) and \( \gamma \) such that \( A = g^\alpha h^\beta \) and \( \tilde{A} = \tilde{g}^\alpha \tilde{h}^\beta \) hold in groups \( G \) and \( \tilde{G} \) simultaneously, where \( G = \langle g \rangle = \langle h \rangle \) and \( \tilde{G} = \langle \tilde{g} \rangle = \langle \tilde{h} \rangle \). Conventionally, the values in the parenthesis \( (\alpha, \beta, \gamma) \) denote quantities of knowledge which is being proven, while the other values are public to the verifier.
TABLE 1
The Comparison Between Our Scheme and Related Schemes

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Unlinkability</th>
<th>Untransferability</th>
<th>Double Spend Detection</th>
<th>De-anonymisation</th>
<th>Attribute-based Ticketing</th>
<th>Security Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>[16]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[37]</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>Sketch</td>
</tr>
<tr>
<td>[29]</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[19]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[38]</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[12, 31]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[35]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[33]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[36]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>[8]</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Sketch</td>
</tr>
<tr>
<td>Our Scheme</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>Reduction</td>
</tr>
</tbody>
</table>

TABLE 2
Notation

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ℓ</td>
<td>A security number</td>
</tr>
<tr>
<td>e(ℓ)</td>
<td>A negligible function in ℓ</td>
</tr>
<tr>
<td>CA</td>
<td>A central authority</td>
</tr>
<tr>
<td>S</td>
<td>A ticket seller</td>
</tr>
<tr>
<td>U</td>
<td>A user</td>
</tr>
<tr>
<td>V</td>
<td>A ticket verifier</td>
</tr>
<tr>
<td>H</td>
<td>A cryptographic hash function</td>
</tr>
<tr>
<td>p</td>
<td>A universal set of ticket policies</td>
</tr>
<tr>
<td>p_U</td>
<td>The policies satisfied by U</td>
</tr>
<tr>
<td>e_p_j</td>
<td>The j-th range policy</td>
</tr>
<tr>
<td>z_i</td>
<td>The i-th set policy</td>
</tr>
<tr>
<td>I_j</td>
<td>The j-th item in z_i</td>
</tr>
<tr>
<td>σ_S</td>
<td>A credential of S</td>
</tr>
<tr>
<td>σ_U</td>
<td>A credential of U</td>
</tr>
<tr>
<td>A_U</td>
<td>The attributes of U</td>
</tr>
<tr>
<td>ID_U</td>
<td>The identity of U</td>
</tr>
<tr>
<td>ID_S</td>
<td>The identity of S</td>
</tr>
<tr>
<td>PoK</td>
<td>Proof of knowledge</td>
</tr>
<tr>
<td>Serv</td>
<td>The services requested by U</td>
</tr>
<tr>
<td>VP_X</td>
<td>A validity period for X</td>
</tr>
<tr>
<td>MSK</td>
<td>The master secret key of the system</td>
</tr>
<tr>
<td>params</td>
<td>The public parameters of the system</td>
</tr>
<tr>
<td>Ticket_U</td>
<td>A ticket of U</td>
</tr>
<tr>
<td>Trans_U</td>
<td>A proof transcript of the ticket Ticket_U</td>
</tr>
<tr>
<td>KG(1^ℓ)</td>
<td>A secret-public key pair generation algorithm</td>
</tr>
<tr>
<td>BG(1^ℓ)</td>
<td>A bilinear group generator</td>
</tr>
<tr>
<td>x ~ U</td>
<td>x is randomly selected from the set U</td>
</tr>
<tr>
<td>A(x) → y</td>
<td>y is obtained by running the algorithm A(·) with input x</td>
</tr>
<tr>
<td>A_U</td>
<td>A_U satisfies the item I_j</td>
</tr>
<tr>
<td>(SK_S, PK_S)</td>
<td>A secret-public key pair of S</td>
</tr>
<tr>
<td>(SK_U, PK_U)</td>
<td>A secret-public key pair of U</td>
</tr>
</tbody>
</table>

2.4 Boneh-Boyen (BB) Signature

In 2004, Boneh and Boyen [31] proposed a short signature scheme. This scheme was used to construct efficient set-membership proof and range proof [47]. In this paper, we use this signature scheme to generate tags for the ticket policies. The scheme works as follows:

KeyGen. Let \( BG(1^ℓ) \rightarrow (e, p, G, G_T) \) and \( g_1, g_2 \) be generators of \( G \). The signer generates a secret-public key pair \((x, Y)\) where \( x \overset{R}{\in} \mathbb{Z}_p \) and \( Y = g_2^x \).

Signing. To sign on a message \( m \in \mathbb{Z}_p \), the signer computes the signature as \( σ = g_1^m \).

Verifying. To verify whether \( σ \) is a signature on the message \( m \), the verifier checks \( e(σ, Yg_2^m) = e(g_1, g_2) \).

**Theorem 1.** (Boneh and Boyen [31]) This signature scheme is \((q_S, e(ℓ))\)-secure against existentially forgery under the weak chosen message attacks if the \((q, e(ℓ))\)-SDH assumption holds on \((e, p, G, G_T)\), where \( q_S \) is the number of signing queries made by the adversary \( A \), \( q > q_S \) and \( e(ℓ) = e(ℓ) \).

2.5 Signature with Efficient Proof Protocol

Au et al. [48] proposed a signature with an efficient proof protocol scheme and referred to it as BBS+ signature. In this paper, we use their signature scheme to issue credentials to users and ticket sellers and to generate tickets for users. This scheme works as follows:

KeyGen. Let \( BG(1^ℓ) \rightarrow (e, p, G, G_T) \) and \((h, g_0, g_1, \ldots, g_{n+1})\) be generators of \( G \). The signer generates a secret-public key pair \((x, Y)\) where \( x \overset{R}{\in} \mathbb{Z}_p \) and \( Y = h^x \).

Signing. To sign on \((m_1, m_2, \ldots, m_n) \in \mathbb{Z}_p^n\), the signer selects \( w, s \overset{R}{\in} \mathbb{Z}_p \) and computes \( σ = (g_0g_1^{m_1} \cdots g_{n+1}) \). The signature on \((m_1, m_2, \ldots, m_n)\) is \((w, s, σ)\).

Verifying. To verify whether \((w, s, σ)\) is a valid signature on \((m_1, m_2, \ldots, m_n)\), the verifier checks \( e(σ, Yh^w) = e(g_0g_1^{m_1} \cdots g_{n+1}h, h) \).

Proof of the Signature. To prove \((w, s, σ)\) is a signature on \((m_1, m_2, \ldots, m_n)\), the prover selects \( r_1, r_2 \overset{R}{\in} \mathbb{Z}_p \) and computes \( A_1 = σg_2^{r_1} \) and \( A_2 = g_1^{r_2} \). Let \( t_1 = wr_1 \) and \( t_2 = wr_2 \). We utilise Au et al.'s [48] honest-verifier zero-knowledge proof of knowledge protocol, II, as follows:

\[
\begin{align*}
\text{PoK} \left\{ (r_1, r_2, t_1, t_2, w, s, σ, m_1, \ldots, m_n) : A_2 = g_1^{r_1}g_2^{r_2} \land \right. \\
A_2 = g_1^{r_1}g_2^{r_2} \land e(A_2, Y) = e(g_1, h)^s \cdot e(A_1, Y)^{-w}, \\
e(g_2, h)^{r_1w} \cdot e(g_2, Y)^{r_2w} \cdot \prod_{i=2}^{m-1} e(g_i, h)^{m-1} \right\}
\end{align*}
\]

**Theorem 2.** (Au et al. [48]) This signature with an efficient proof protocol is \((q_S, e(ℓ))\)-existentially unforgeable under the adaptively chosen message attacks if the
\( (q, \epsilon'(<\ell>))\)-SDH assumption holds on \((e, p, G, G_2)\), where \(q_S\) is the number of signing queries made by the adversary \(A\), \(q > q_S\) and \(\epsilon'(<\ell>) > q \cdot \epsilon(<\ell>)\).

3 Formal Definitions and Security Models

In this section, we provide the formal definitions and security models of our scheme which will be used to verify its security.

3.1 Formal Definitions

Our scheme consists of the following four entities: central authority \(CA\), user \(U\), ticket seller \(S\) and ticket verifier \(V\). \(CA\) authenticates \(U\) and \(S\), and issues anonymous credentials to them; \(S\) registers to the \(CA\), obtains anonymous credentials from the \(CA\), and sells tickets to \(U\) in accordance with the ticket policies; \(U\) registers to the \(CA\), obtains anonymous credentials from the \(CA\), purchases tickets from \(S\), and proves the possession of tickets to \(V\); \(V\) validates the tickets provided by \(U\) and detects whether a ticket is double spent.

The interactions between the different entities in our scheme is shown in Fig. 1. The algorithms associated with these interactions are formally defined as follows:

\[
\text{Setup}(1^\ell) \rightarrow (\text{MSK}, \text{params}, \mathbb{P}). \quad CA \text{ inputs a security parameter } 1^\ell \text{, and outputs the master secret key } \text{MSK}, \text{ the public parameters } \text{params} \text{ and a universal set of ticket policies } \mathbb{P}.
\]

Registration. This algorithm consists of the following two sub-algorithms: \(S\)’s registration \(S\text{Registration}\) and \(U\)’s registration \(U\text{Registration}\).

1) \(S\text{Registration}(S(ID_S, SK_S, PK_S, \text{params}) \leftrightarrow CA(\text{MSK}, PK_S, \text{params})) \rightarrow (\sigma_S, (ID_S, PK_S)). \quad S\) runs the key generation algorithm \(KG(1^\ell) \rightarrow (SK_S, PK_S)\) to generate his secret-public key pair \((SK_S, PK_S)\), inputs his identity \(ID_S\), his secret-public key pair \((SK_S, PK_S)\) and the public parameters \(\text{params}\), and outputs a credential \(\sigma_S\). \(CA\) inputs his secret key \(\text{MSK}\), \(S\)’s public key \(PK_S\) and the public parameters \(\text{params}\), and outputs \((ID_S, PK_S)\).

2) \(U\text{Registration}(U(ID_U, AU, SK_U, PK_U, \text{params}) \leftrightarrow CA(\text{MSK}, AU, PK_U, \text{params})) \rightarrow (\sigma_U, (ID_U, PK_U)). \quad U\ runs the key generation algorithm \(KG(1^\ell) \rightarrow (SK_U, PK_U)\) to generate his secret-public key pair \((SK_U, PK_U)\), inputs his identity \(ID_U\), his attributes \(AU\), his secret-public key pair \((SK_U, PK_U)\) and the public parameters \(\text{params}\), and outputs a credential \(\sigma_U\). \(CA\) inputs the master secret key \(\text{MSK}\), \(U\)’s attributes \(AU\), \(U\)’s public key \(PK_U\) and the public parameters \(\text{params}\), and outputs \((ID_U, PK_U)\).

Ticket-Issuing\((U(SK_U, PK_U, AU, \sigma_U, PS_U, \mathbb{P}, VP, Serv, \text{params}) \leftrightarrow SSK_S, PK_S, PS_U, \mathbb{P}, Price, VP, Serv, \text{params})) \rightarrow (Ticket_U, (PS_U, Service))\). This is an interactive algorithm executed between \(U\) and \(S\). \(U\) inputs his secret-public key pair \((SK_U, PK_U)\), his attributes \(AU\), his secret key \(\sigma_U\), a pseudonym \(PS_U\), the ticket policies \(\mathbb{P}\), a valid period \(VP\), the selected services \(Serv\) and the public parameters \(\text{params}\), and outputs \((PS_U, Serv)\).

Ticket-Validating\((U(SK_U, PS_U, Ticket_U, VP, Serv, \text{params}) \leftrightarrow V(VP, Serv, \text{params})) \rightarrow (0/1, (Serv, Trans_T)). \quad This\ is\ an\ interactive\ algorithm\ executed\ between\ \(U\)\ and\ \(V\). \(U\) inputs his secret-public key pair \((SK_U, PK_U)\), his ticket \(Ticket_U\), the valid period \(VP\), the selected services \(Serv\) and the public parameters \(\text{params}\), and outputs \((PS_U, Serv)\).

Double-Spend-Detecting\((Trans_T, \text{params}) \rightarrow (PK_U, \perp). \quad V\ inputs\ a\ proof\ transcript\ \(Trans_T\)\ and\ the\ public\ parameters\ \(\text{params}\)\ and\ outputs\ \(U\)’s\ public\ key\ \(PK_U\)\ if\ \(U\)\ has\ used\ a\ ticket\ twice;\ otherwise\ it\ outputs\ \(\perp\)\ to\ indicate\ that\ no\ double\ spend\ is\ found.

\[\text{Definition 2.}\ A\ privacy-preserving\ electronic\ ticket\ scheme\ with\ attribute-based\ credential\ is\ correct\ if\]

\[
\begin{align*}
\text{Setup}(1^\ell) & \rightarrow (\text{msk}, \text{params}, \mathbb{P}); \\
\text{SRegistration}(S(ID_S, SK_S, PK_S, \text{params}) \leftrightarrow CA(\text{MSK}, PK_S, \text{params})) & \rightarrow (\sigma_S, (ID_S, PK_S)); \\
\text{URegistration}(U(ID_U, AU, SK_U, PK_U, \text{params}) \leftrightarrow CA(\text{MSK}, AU, PK_U, \text{params})) & \rightarrow (\sigma_U, (ID_U, PK_U)); \\
\text{Ticket-Issuing}(U(SK_U, PK_U, AU, \sigma_U, PS_U, \mathbb{P}, VP, Serv, \text{params}) \leftrightarrow SSK_S, PK_S, PS_U, \mathbb{P}, Price, VP, Serv, \text{params})) & \rightarrow (Ticket_U, (PS_U, Service)); \\
A_U & \equiv \mathbb{P}.
\end{align*}
\]

and

\[
\begin{align*}
\text{Ticket-Validating}(U(SK_U, PS_U, Ticket_U, VP, Serv, \text{params}) \leftrightarrow V(VP, Serv, \text{params})) & \rightarrow (0/1, (Serv, Trans_T)); \\
\text{Double-Spend-Detecting}(Trans_T, \text{params}) & \rightarrow (PK_U, \perp); \\
\text{Ticket-Issuing}(U(SK_U, PK_U, Ticket_U, VP, Serv, \text{params}) \leftrightarrow V(VP, Serv, \text{params})) & \rightarrow (0/1, (Serv, Trans_T)); \\
\text{Double-Spend-Detecting}(Trans_T, \text{params}) & \rightarrow (PK_U, \perp).
\end{align*}
\]

3.2 Security Model

While Universally Composable (UC) security models \([49]\) can offer strong security, it is very difficult to construct
a scheme which can be shown to provide UC security. To the best of our knowledge, none of the existing smart ticketing schemes was proven in UC security model. Consequently, the security of our scheme is defined by using the simulation-based definition as introduced in [50], [51], [52], [53]. The simulation-based model is defined by the indistinguishability between the following real world and ideal world experiment.

The Real-World Experiment. We first present how our scheme works with the central authority CA, the ticket seller S, the user U and the ticket verifier V are honest. The real-world adversary A can control S, U and V, but cannot control CA. The entities controlled by A can deviate arbitrarily from their behaviour described below. CA runs Setup(1\(^f\)) \rightarrow (MSK, params, \mathcal{P}) to generate the master secret key msk, system public parameters params and the universal set \mathcal{P} of ticket polices, and sends params and \mathcal{P} to U, S and V.

When receiving a registration message (registration, ID\(_S\)) from CA, S executes the seller registration algorithm SRegistration with CA. S runs KG(1\(^f\)) \rightarrow (SK\(_S\), PK\(_S\)) as input his identity ID\(_S\), the secret-public key pair (SK\(_S\), PK\(_S\)) and the public parameters params, outputs a credential \(\sigma\). CA takes inputs his master secret key MSK, S's public key PK\(_S\) and the public parameters params, and outputs S’s identity ID\(_S\) and public key PK\(_S\). S sends a bit \(b \in \{0,1\}\) to CA to show whether the SRegistration algorithm succeed (\(b = 1\)) or failed (\(b = 0\)).

When receiving a registration message (registration, ID\(_U\), AU) from CA, U executes the user registration algorithm URegistration with CA. U runs KG(1\(^f\)) \rightarrow (SK\(_U\), PK\(_U\)) as input his identity ID\(_U\), attributes AU, secret-public key pair (SK\(_U\), PK\(_U\)) and the public parameters params, and outputs a credential \(\sigma\). CA takes inputs his master secret key MSK, U's public key PK\(_U\) and the public parameters params, and outputs U’s identity ID\(_U\), attributes AU and public key PK\(_U\). U sends a bit \(\overline{b} \in \{0,1\}\) to CA to show whether the URegistration algorithm succeed (\(\overline{b} = 1\)) or failed (\(\overline{b} = 0\)).

When receiving a ticket issuing message (ticket \(_\text{issuing}, AU, VP, Service\)) from CA, U first checks whether he has got a credential for AU. If so, U executes the ticket issuing algorithm Ticket-Issuing with S. U takes as inputs his secret-public key pair (SK\(_U\), PK\(_U\)), attributes AU, a pseudonym PS\(_U\), his credential \(\sigma\), the valid period \(VP\), the service Serv and the public parameters params. S takes as input his secret-public key pair (SK\(_S\), PK\(_S\)), the valid period \(VP\), the service Serv and the public parameters params. Finally, U obtains a ticket \(T_U\) to \(\overline{b} = 1\) to show failure. S outputs U’s pseudonym PS\(_U\) and the service Serv. If the ticket issuance is successful, U sends a bit \(\overline{b} \in \{0,1\}\) to CA to show the Ticket-Issuing algorithm succeed (\(\overline{b} = 1\)) or failed (\(\overline{b} = 0\)).

When receiving a ticket validation message (ticket \(_\text{validating}, T_U, VP, Serv, params\) from CA, U first checks whether he has the ticket \(T_U\) which includes the valid period \(VP\) and the service Serv. If so, U executes the ticket validating algorithm Ticket-Validating with V; otherwise U outputs \(\overline{b} = 1\) to show he does not have the ticket \(T_U\). If U has the ticket \(T_U\), he takes as input his secret-public key pair (SK\(_S\), PK\(_S\)), the ticket \(T_U\), the valid period \(VP\), the service Serv and the system public parameters params, and outputs a bit \(\overline{b} \in \{0,1\}\) to show whether the ticket is valid (\(\overline{b} = 1\)) or invalid (\(\overline{b} = 0\)). V takes input the valid period \(VP\), the service Serv and the public parameters params, and outputs the service Serv and the transcript Trans. Finally, if \(\overline{b} = 1\), U returns success; otherwise U returns fail.

When receiving a double spend detecting message (double_spend_detecting, Trans, params) from CA, V checks that whether there is a (Trans', params) with Trans = Trans'. If so, V returns a bit \(\overline{\overline{b}} = 1\) to indicate that it is a double spend ticket; otherwise \(\overline{\overline{b}} = 0\) is returned to show that the ticket has not been double spent.

The Ideal-World Experiment. In the ideal world experiment, there are the same entities as in real world experiment, including the central authority CA, ticket seller S, user U and ticket verifier V. All communications among these entities must go through a trusted party TP. The behaviour of TP is described as follows. TP maintains four lists which are
initially empty: a ticket seller credential list, a user credential list, a ticket list for each user and a ticket validating list.

When receiving a registration message \((\text{registration}, ID_U, A_U)\) from \(U\), \(TP\) sends \((\text{registration}, ID_U, A_U)\) to \(CA\) and obtains a bit \(\nu = 1\) from \(CA\). If \(\nu = 1\), \(TP\) adds \((U, A_U)\) into the user ticket list and sends \(\nu\) to \(U\); otherwise, \(TP\) sends \(\nu = 0\) to \(S\) to indicate failure.

When receiving a registration message \((\text{registration}, ID_U, A_U)\) from \(U\), \(TP\) sends \((\text{registration}, ID_U, A_U)\) to \(CA\) and obtains a bit \(\nu = \{0, 1\}\) from \(CA\). If \(\nu = 1\), \(TP\) adds \((U, A_U)\) into the user ticket list and sends \(\nu\) to \(U\); otherwise, \(TP\) sends \(\nu = 0\) to \(S\) to indicate failure.

When receiving a ticket issuing message \((\text{ticket issuing}, Ps_U, Price, VP, Serv)\) from \(U\), \(TP\) sends \((\text{ticket issuing}, Ps_U, Price, VP, Serv)\) to \(S\) and obtains a bit \(\nu = \{0, 1\}\) from \(S\). If \(\nu = 1\), \(TP\) adds \((U, A_U, Price, VP, Serv)\) into the user ticket list, and sends \(\nu\) to \(V\); otherwise, \(TP\) sends \(\nu = 0\) to \(U\) to indicate failure.

When receiving a ticket validating message \((\text{ticket validating}, T_U)\) from \(V\), \(TP\) checks whether \(T_U\) is in the user ticket list. If so, \(TP\) sends a bit \(\nu = 1\) to \(U\) and puts \(T_U\) into the ticket validation list \(TVL\); otherwise, \(TP\) sends \(\nu = 0\) to indicate failure.

When receiving a double spend detecting message \((\text{double spend detecting}, T_U)\) from \(V\), \(TP\) checks whether \(T_U \in TVL\). If it is, \(TP\) returns \(\nu = 1\) to \(U\) to indicate it is double spend; otherwise, \(\nu = 0\) is returned to show it is not double spent.

The entities \(CA', S', U'\) and \(V'\) in ideal world simply relay the inputs and outputs between \(E\) and \(TP\).

\textbf{Definition 3.} Let \(\text{Real}_{E,A}(\ell)\) be the probability that the environment \(E\) outputs 1 when running in the real world with the adversary \(A\) and \(\text{Ideal}_{E,A}\) be the probability that \(E\) outputs 1 when running in the ideal world with the adversary \(A'\). A set of cryptographic protocols is said to securely implement privacy-preserving electronic ticket scheme with attribute-based credential if \(|\text{Real}_{E,A}(\ell) - \text{Ideal}_{E,A}(\ell)| \leq \ell(\ell).

\textbf{Security Properties.} We now look at the security properties of our scheme which the ideal-world experiment can provide.

\textbf{User’s Privacy.} \(S'\) does not know users’ identities and their exact attributes, namely \(S'\) only knows that a user buys a ticket for which she has the required attributes. Even if \(S'\) colludes with \(V'\) and potentially with other users, they can only try to know the attributes required by the ticket policies. Furthermore, two tickets for the same users cannot be linked. Since each user needs to prove that he knows the corresponding secret key included in a ticket when using the ticket, he cannot transfer his tickets to others.

\textbf{Seller’s Security.} \(U'\) cannot generate a ticket on behalf of the seller \(S'\). Even if \(U'\) colludes potentially with other users and \(V'\), they cannot forge a valid ticket. Since a double spend ticket can be detected and the real user can be identified, \(U'\) cannot double spend a ticket. Therefore, the seller’s security includes both unforgeability, double spend detection and de-anonymization.

\section{Construction of our scheme}

In this section, we describe the formal construction of our scheme. Our scheme uses a number of ideas and concepts from Au et al.’s signature with efficient protocol scheme \cite{Auetal03}, Camenisch et al.’s set-membership proof scheme and range proof scheme \cite{CaPi02}, Pedersen’s commitment scheme \cite{Pe88} and Au et al.’s e-cash \cite{Au01} scheme. In particular, we incorporate Au et al.’s signature scheme which enables a user to obtain a signature on a committed block of attributes and prove the knowledge of the signature in zero-knowledge. This is to issue credentials to users and ticket sellers and to generate tickets for users. We adapt Camenisch et al.’s set-membership proof and range proofs schemes to prove a user’s attributes. In our scheme, these attributes are additionally certified by a trusted third party as well. Pedersen’s commitment scheme is used in our scheme to hide the knowledge which a prover needs to prove. Lastly we incorporate Au et al.’s \cite{Au01} approach to detect and de-anonymise a double spend user.

\textit{Construction challenges:} The schemes described in \cite{Au01, CaPi02, Au03, Au01} form the basis of our construction, the challenge is to combine and adapt them such that the resulting scheme provides the following three additional features: (1) The attributes (e.g. age, disability, etc.) which a user needs to prove to a ticket seller must be certified by a trusted third party or otherwise users could simply buy discounted tickets using attributes which they do not possess. To address this, Au et al.’s signature scheme \cite{Au01} is used to certify a user’s attributes. (2) Tickets need to be untransferable and unlinkable while doublespend detection must be possible. Thus our tickets are generated using anonymous credentials (unlinkability) which include a user’s personal information (untransferability). To detect a double spend user, each ticket includes a serial number. If two tickets have the same serial number, the public trace technique proposed by Au et al. in \cite{Au01} is used to reveal the user’s identity (via her public key). (3) To provide a high degree of flexibility for setting ticket policies, both range policies and set policies must be available for use. Users can then use their certified attributes to demonstrate membership of multiple range and set policies, e.g. to get a young-persons discount, a frequent traveller bonus as well as a disability reduction.

\subsection{High-Level Overview}

In our e-ticket system, the type of tickets can be influenced by two kinds of policies: range and set. Range policies might include attributes like age, number of journeys made, salary, etc.; while set policies might consist of various other attributes, such as profession, disability, location, etc.

\textbf{Setup.} Figure \ref{fig:setup} shows how the scheme is initialised. The ticket price policies \(P\) is set to \(P = \{R_1, \ldots, R_{N_1}, S_1, \ldots, S_{N_2}\}\) where \(\{R_1, \ldots, R_{N_1}\}\) are the supported range policies and \(\{S_1, \ldots, S_{N_2}\}\) are the supported set policies. The CA selects the following secret keys \(MSK = (x, y, \mu_1, \ldots, \mu_{N_2})\) where \(x\) is used to generate credentials for users of the system, \(y\) is used to generate tags identifying the range policies and the \(\mu_i\) (\(i = 1, \ldots, N_2\)) are used to generate tags identifying the set policies. The CA then publishes the public parameters \(\text{params}\) which include the ticket price policy \(P\) together with the range and set policy tags.
CA publishes the ticket price policies $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_{N_1}, \mathcal{S}_1, \ldots, \mathcal{S}_{N_2}\}$ where $\mathcal{P}_l = [c_l, d_l]$ is a range policy (i.e. age, mileage) and $\mathcal{S}_i = \{I_{i_1}, I_{i_2}, \ldots, I_{i_T}\}$ is a set policy (i.e. location, profession, disibility) and consists of $\zeta$ items $I_{i_j}$ for $l = 1, 2, \ldots, N_1$ and $i = 1, 2, \ldots, N_2$.

CA runs $BG(1^l) \rightarrow (e, p, \mathcal{G}, \mathcal{G}_x)$. Suppose that the longest interval length in $\{\mathcal{P}_1, \ldots, \mathcal{P}_{N_1}\}$ is $[0, q^k]$ where $q \in \mathbb{Z}_p$ and $p > 2q^k + 1$. Let $g, g_0, g_1, g_2, g_3, \ldots, g_{N_1}, h, g, \eta, \xi, \rho, \theta, \eta_1, \eta_2, \ldots, \eta_{N_2}$ be generators of $\mathcal{G}$, $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p$ and $H' : \{0, 1\}^* \rightarrow \mathbb{G}$ be two cryptographic hash functions.

CA select $x, y, m_1, m_2, \ldots, m_{N_2} \in \mathbb{Z}_p$ and computes $\tilde{g} = g^y$, $\tilde{h} = h^y$, $h_0 = h^\frac{1}{q^k}$, $h_1 = h^y$, $\tilde{h}_{k-1} = h^\frac{1}{q^k}$, $\tilde{h}_1 = h^y$, $\tilde{h}_{k-1} = h^\frac{1}{q^k}$, $\tilde{\eta}_1 = \eta_1^y$, $\tilde{\eta}_2 = \eta_2^y$, $\ldots, \tilde{\eta}_{N_2} = \eta_{N_2}^y$ and

$$\left(\tilde{\eta}_1 = \eta_1^\frac{1}{y} \tilde{\eta}_1, \tilde{\eta}_2 = \eta_2^\frac{1}{y} \tilde{\eta}_2, \ldots, \tilde{\eta}_{N_2} = \eta_{N_2}^\frac{1}{y} \tilde{\eta}_{N_2}\right)_{i=1}^{N_2}.$$ 

The secret key of CA is $MSK = (x, y, m_1, m_2, \ldots, m_{N_2})$ and the public parameters are $params = (e, p, \mathcal{G}, \mathcal{G}_x, g, g_0, g_1, g_2, g_3, \ldots, g_{N_1}, h, g, \eta, \xi, \rho, \theta, \eta_1, \eta_2, \ldots, \eta_{N_2}, (\tilde{\eta}_1, \tilde{\eta}_2, \ldots, \tilde{\eta}_{N_2})_{i=1}^{N_2})$.

Fig. 2. Setup Algorithm

Registration. The steps involved in the registration process are shown in Figure 3. The registration of a seller $S$ requires $S$ to generate a secret-public key pair $(x_s, Y_S)$. He sends $Y_S$ to the CA as well as a proof of knowledge $\Pi_{X_S}$ to demonstrate he knows the secret key $x_s$. Using some out-of-band channel, $S$ authenticates himself to the CA and shows his evidence that he is allowed to operate as a seller. If $\Pi_{X_S}$ is successful, the CA generates a credential $\sigma_S$ which is a BBS+ signature including the public key $Y_S$ as well as a validity period $VP_S$. The secret part of the CA is $\Pi_{X_S}$, and so it uses a zero-knowledge proof of knowledge of $\Pi_{X_S}$'s validity period $VP_S$ for it. These details are then sent back to $S$ which verifies that the CA has allowed him as a seller by validating the credential $\sigma_S$.

In the case of a user registration, a user $U$ generates a secret-public key pair $(x_u, Y_U)$ and submits her public key $P_U$ together with a proof of knowledge $\Pi_{X_U}$ showing that she knows the secret key $x_u$. She also sends the CA the list of attributes $A_U$ (e.g. age, profession, location, etc.) which allow her to get discounted tickets. Again, using an out-of-band channel, she authenticates herself to the CA and shows her evidence that she is allowed to operate as a user. If $\Pi_{X_U}$ holds, the authentication is successful and the CA is satisfied with the provided evidence, it generates a credential $\sigma_U$ which is a BBS+ signature scheme including the public key $Y_U$, its validity period $VP_U$, as well as the corresponding attributes $A_U$. These details are then sent back to $U$ who uses them to verify that she is now a legitimate user of the system and that her attributes have been certified by the CA.

Ticket Issuing. Figure 4 shows the details of the ticket issuing phase. Let $P_U$ consists of the names of range policies and set policies satisfied by $U$. In order to prevent attackers from collecting users’ private information, a seller $S$ first needs to prove to $U$ that he is authorised by the CA. This is done by constructing a proof of knowledge $\Pi_{X_S}$ of the seller’s credential $\sigma_S$. If the proof holds, the user $U$ proceeds by generating a new pseudonym $Y$ which involves her secret key $x_u$ and constructs a proof of knowledge $\Pi_{X_Y}$ of $Y$. This proof shows to $S$ that the CA has certified her as a legitimate user who has the claimed attributes which entitle her to buy the ticket corresponding to her provided attributes. After $S$ has successfully verified her proof, he constructs a ticket $T_U$ applying a BBS+ signature scheme which includes the user’s pseudonym $Y$, the applicable range and set policies of the user relevant to the ticket, a serial number to enable double spend detection as well as the ticket’s price and validity period $VP_U$. Note that while the ticket price and its validity period are included in the construction of $T_U$, they are just free text entries and should only be used when price and validity periods are required by the application context, e.g. when the validity period is important, $S$ should check the user’s credential valid period $VP_U$ and make sure that the ticket valid period $VP_T$ is no later than $VP_U$. Then together with its associated details is then sent back to the user who can use the information together with the public key of the seller $Y_S$ to verify the validity of the information. Note that our scheme provides ticket unlinkability due to the use of user pseudonyms which prevents the seller $S$ as well as any verifier $V$ from linking any two tickets requested by the same user even if they collide.

Ticket Validation. Figure 5 depicts the steps to validate a ticket. The user $U$ initializes an empty table $Table_U$ to store the identity information of any verifier $V$. The purpose of this table is to ensure that a verifier can only ask for a ticket once to prevent an honest user from being de-anonymised by a malicious verifier. The verifier $V$, on the other hand, initializes an empty table $Table_V$ to store the authentication transcript from $U$ to determine if a ticket has already been used (i.e. double spend detection). The ticket verification process is started by the verifier sending a fresh nonce $r$ and its identity $ID_V$ to the user. It is assumed that there is some out-of-band channel which allows the user to “authenticate” the verifier, e.g. it is a guard on the train or a gate at the entrance of the platform, etc. $U$ first checks that $V$ does not yet have an entry in $Table_U$. If an entry exists, $U$ aborts the process to avoid de-anonymisation. Otherwise, she proceeds to send $V$ a “ticket transcript”, $Trans_T$, of her ticket $Ticket_U$, which includes a zero-knowledge proof of knowledge $\Pi_{X_T}$. The transcript should convince $V$ that she is a legitimate user who is in the possession of a valid ticket $Ticket_U$. Because $Ticket_U$ includes the user’s secret key $x_u$ as part of her pseudonym $Y$, knowledge of which needs to be demonstrated as part of $\Pi_{X_T}$. Our scheme ensures ticket untransferability assuming $U$’s private key has not been
Ticket Seller $S$
Selects $x_s \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes $Y_S = \rho^{x_s}$.
Computes the proof $\Pi_S^\ast : \text{PoK}\{x_s : Y_S = \rho^{x_s}\}$.
Verifies $e(\sigma_S, \tilde{g} g^{x_s}) = e(g, \tilde{g})$.
\[ e(\Pi_S^\ast, g) \cdot e(g, \tilde{g})^{x_s}. \]
Keeps the credential $\text{Cred}_S = (c_s, r_s, \sigma_S)$.

Central Authority $CA$
Selects $c_s, r_s \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes $\sigma_S = (g_0 g_i^{H(VP_S)} Y_S g^{x_s})^{\frac{1}{x_s}}$, where $VP_S$ is a valid period.

User $U$
Selects $x_u \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes $Y_U = \xi^{x_u}$.
Selects $r \overset{R}{\leftarrow} \mathbb{Z}_p$ and compute $R = g^r$.
Computes the proof $\Pi_U^\ast : \text{PoK}\{(x_u, r) : Y_U = \xi^{x_u} \land R = g^r\}$.
Verifies $e(\sigma_U, \tilde{g} g^{x_u}) = e(g, \tilde{g} g_i^{H(VP_U)}).
\[ e(Y_U, g) \cdot e(g, \tilde{g})^{x_u} \cdot \prod_{i=1}^{N_1} e(g_i, g_i^{H(l_i)}). \]
Keeps the credential $\text{Cred}_U = (c_u, r_u, \sigma_U)$.

Double Spend Detecting. Figure 6 shows the double spend detection process. To determine whether a ticket is being double spent, $V$ checks $\text{Table}_V$ for another ticket transcript $\text{Trans}_T$ with the same serial number $D$. If there is, the ticket is being double spent and $V$ can de-anonymise $U$ by extracting her public key $Y_U$ from the two transcripts; otherwise, it is a new ticket.

It is worth pointing out that the construction of our scheme has the following additional benefit.

Limited Dynamic Policy Update. If the seller $S$ needs to either update some policies in $\mathfrak{P}$ or create new ones, he can contact the central authority $CA$ to update or create the relevant public parameters $\mathfrak{params}$. As a result, when buying a ticket, a user $U$ proves to $S$ that his attributes satisfy the updated policies by using the updated $\mathfrak{params}$ and $S$ will generate tickets according to the updated policies. $U$ only needs to obtain new credentials from the $CA$ if her current attributes do not satisfy the updated policies any more. For example, suppose that Alice is 16 years old. If the seller $S$ requests that the existing policy range of $[12, 18]$ is changed to $[15, 20]$ instead, then Alice can still use her existing credentials. However, if the policy were changed from $[12, 18]$ to $[18, 25]$ instead, then Alice would need to contact the $CA$ to update her credentials.

Correctness. The correctness of our scheme is shown in the full version of this paper [55].

5 Benchmarking Results

In this section we evaluate the performance of our scheme. The source code of the scheme’s implementation is available at [55] and its performance has been measured on a Dell Inspiron Latitude E5270 laptop with an Intel Core i7-6600U CPU, 1TB SSD and 16GB of RAM running Fedora 27. The implementation makes use of bilinear maps defined over elliptic curves as well as other cryptographic primitives. We used the JPBC library [57] for the bilinear maps and bouncyCastle [58] for the other cryptographic required by our scheme. Note that the Java based implementation of the JPBC API [57] was used throughout.

Recall from Section 2 that our scheme requires a Type I symmetric bilinear map, $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_r$. The JPBC library [57] provides three different instances of a symmetric pairing with their Type A, A1 or E pairings. The Type A and A1 pairings are based on the elliptic curve $E : y^2 = x^3 + x$ over the finite field $F_p$. In both cases, the group $\mathbb{G}$ in is the group of points on the elliptic curve, $E(F_p)$. The Type E pairing, on the other hand, is based on the Complex Multiplication (CM) method of constructing elliptic curves starting with the Diophantine equation $DV^2 = 4p - t^2$. The details of each construction can be found in [59].

In our implementation, we use the default parameters during the instantiation of the different pairings, e.g. Type A is constructed using $rBits = 160$, $qBits = 512$, Type A1 uses 2 primes of size $qBits = 512$ and Type E is instantiated with $rBits = 160$ and $qBits = 1024$.

Note that according to Table 1 in [60], JPBC’s default Type A pairing provides approximately the equivalent of 80-bit symmetric or 1024 RSA-style security. This is sufficient for providing a baseline for taking time measurements.

For the hash functions $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H' : \{0,1\}^* \rightarrow \mathbb{G}$ required by our scheme (see Fig 2), we used...
Let $\mathbb{P}_U$ consists of the names of range polices and set policies satisfied by $U$.

Let $a_l \in A_U, \ a_l \in [c_l, d_l)$, $a_l - c_l, a_l - d_l + q^k \in [0, q^k)$,

$$a_l - c_l = \sum_{i=0}^{k-1} w_i l_i q^i, \ a_l - d_l + q^k = \sum_{i=0}^{k-1} w'_i l_i q^i,$$

where $w_i, w'_i \in [0, 1, \ldots, q - 1]$.

Selects $d, \alpha, \beta, \gamma_1, \gamma_2, \ldots, \gamma_{N_l}, t_0, t_1, l_1, \ldots, l_{k-1}$,

$$\nu_0, \nu_1, \ldots, \nu_k, e_1, e_2, \ldots, e_{N_2} \leftarrow \mathbb{Z}_p.$$

Computes $C = \sigma_0 \nu_0, D = g^q \nu_q, D^\nu_q = g^{\nu_1} \nu_q$,

$$Y = \xi x g^d_1 (Z_l = g^\alpha l_i, A_{w_l} = h_{w_l}^i, \ A'_{w_l} = h_{w_l}^i, (B_i = \eta_i^i)_{l_1}^{N_l}),$$

where $\alpha = \alpha_{u, \nu}, \beta = \beta_{c, u}$.

Computes the proof $\Pi_2^2$ :

$\text{PoK} \left\{ (x_u, c_u, r_u, d, \alpha, \beta, \alpha', \beta', (a_l, (l_i, n_i, w_i),

\nu_0, \nu_1, \ldots, \nu_k, e_1, e_2, \ldots, e_{N_2} \leftarrow \mathbb{Z}_p, \ Y = \xi x g^d_1 \right.$

$$\wedge Z_l = g^\alpha l_i \wedge D = g^q \nu_q \wedge D^\nu_q = g^{\nu_1} \nu_q$$

$$\wedge e(C, g) = e(x, g) \nu_u \cdot e(\nu_q, g)^{\nu_u},$$

$$\wedge e(C, g)^{-\nu_q} \cdot e(\nu_q, g)^{\nu_u} \wedge e(\nu_q, g)^{\nu_u}$$

$$\wedge (Z_l h^{-\nu_u} = g^{\nu_u} \cdot \prod_{i=1}^{k-1} h_i$$

$$\wedge Z_l h^{-\nu_u} = g^{\nu_u} \cdot \prod_{i=1}^{k-1} h_i$$

$$\wedge e(A_{w_l}, h) = e(h, h')^{l_1} \cdot e(A_{w_l}, h')^{-w_l} (l_1, w_l)$$

$$\wedge e(A_{w_l}, h) = e(h, h')^{l_1} \cdot e(A_{w_l}, h')^{-w_l} (l_1, w_l)$$

$$\wedge e(B_i, \eta_i) = e(\eta, \eta_i)^{c_i} \cdot e(B_i, \eta_i)^{H(1_1)} (l_1, N_l) \}$$. 

Computes $d_u = d + d'$ and checks

$$e(T, Y_{d_u}^{\nu_u}) = e(y, \rho) \cdot e(Y, \rho) \cdot e(g_1, \rho)^{d_u},$$

$$e(g_2, \rho)^{\nu_u} \cdot e(\nu_q, \rho)^{\nu_u}.$$  

Keeps the pseudonym as $P_{SU} = \xi x g^d_1$ and the ticket as $Ticket = (d_u, s_u, \omega_u, T_U, P_U, Price, Serv, VP_T )$.

$\text{Selects } z, v \leftarrow \mathbb{Z}_p$ and computes

$$Q = \sigma_S \tilde{Z}, Z = g^z \tilde{v}, Z_{c_u} = g^{z'} \tilde{v}'$$(

$$\tilde{v} = z c_u, \tilde{v}' = v c_u.$$

Computes the proof $\Pi_3^2$

$\text{PoK} \left\{ (c_u, r_u, \sigma_S, z, v) : Z = g^z \tilde{v} \wedge Z_{c_u} =$$

$$g^{z'} \tilde{v}'$$(

$$\wedge e(Q, \tilde{Z}) = e(\rho, \tilde{v}) \xi z \wedge e(g, \tilde{Z})^{\nu_u} \wedge e(\tilde{v}, \tilde{Z})^{\nu_u} \}$$. 

$\text{Selects } d', s_u, \omega_u \leftarrow \mathbb{Z}_p$ and computes

$$T_U = (y g_1 d_1 g_2 s_u g_3 \omega_u)^{\frac{r_u}{r_u}}$$ where $s_u$ is a serial number, $\psi_u = H(\mathbb{P}_U || Price || Serv || VP_T )$, $Price$ is the price of the ticket, $Serv$ are the services which $U$ wants to access and $VP_T$ is a valid period.

$\text{Fig. 4. Ticket Issuing Algorithm}$

$SHA256$ for $H$ and rely on the implementation of “newElementFromHash()” method in the JPBC library for $H^t$.

### 5.1 Timings

Table 5 shows the results of the computational time spent in the various phases of our proposed scheme which required more complex computations (i.e. some form of verification using bilinear maps or generation of zero knowledge proofs). The timings shown have been calculated as the average over 20 iterations.  

The maximum range interval in this instance was 7 which is covered by the interval $[0, 2^3]$ and thus $k = 3$ in the set-up algorithm described in Fig 2. The maximum set size used was 10. It is clear from the computations involved in the generation of $\Pi_2^2$ (see the full version 55) that the computational cost of a range proof increases with $k$ while the number of computations for a set membership proof is independent on the size of the set. As such the numbers presented below provide a reasonable lower bound of the computational costs for range proofs assuming that any useful ranges will have at least an interval length of 4 or more.

Table 5 shows the timings for our current implementation of our scheme with 2 small range policies and 4 set
If there exist \((r, D, E), F, J\) ∈ TableV and transcript \(((r', D', E'), F', J')\) ∈ TableV with \(D = D'\) and \(E \neq E'\), the ticket with serial number \(s_u\) is being double spent. Let \(E = \xi^{s_u}H'(IDV)^{rs_u}\) and \(E' = \xi^{x_u}H'(IDV)^{rs_u}\).

To detect the double spend user, \(V\) computes \(E_{rsu}' = \frac{\xi^{s_u}H'(IDV)^{rsu}}{\xi^{x_u}H'(IDV)^{rsu}} = \xi^{s_u}(r' - r)\) and \(Y_U = \frac{E_{rsu}'}{E_{rsu}}\).

Hence, \(U\) with public key \(Y_U\) is a double spend user.

### TABLE 3

Benchmark results (in ms)

<table>
<thead>
<tr>
<th>Protocol phase</th>
<th>Entity</th>
<th>(#range policies,set policies)= (2,4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Initialisation - Central Authority (CA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>initialise the system</td>
<td>CA</td>
<td>626.05.1</td>
</tr>
<tr>
<td></td>
<td>Type A</td>
<td>9155.95</td>
</tr>
<tr>
<td></td>
<td>Type A1</td>
<td>2895.25</td>
</tr>
<tr>
<td>Issuing phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>generate PoK (\Pi_U^2)</td>
<td>Seller</td>
<td>184.25</td>
</tr>
<tr>
<td>verify (\Pi_U^2)</td>
<td>User</td>
<td>107.9</td>
</tr>
<tr>
<td>generate ticket request, (\Pi_U^2)</td>
<td>User</td>
<td>1008.7</td>
</tr>
<tr>
<td>verify (\Pi_U^2)</td>
<td>Seller</td>
<td>787.3</td>
</tr>
<tr>
<td>generate ticket</td>
<td>Seller</td>
<td>47.85</td>
</tr>
<tr>
<td>verify ticket</td>
<td>User</td>
<td>52.5</td>
</tr>
<tr>
<td>Total system run time</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TABLE 4

Type A: benchmark results for different ranges and set sizes (in ms)

<table>
<thead>
<tr>
<th>Ticket issuing phase</th>
<th>(k = 5)</th>
<th>(k = 10)</th>
<th>(k = 20)</th>
<th>(s = 10)</th>
<th>(s = 100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>([0, 31])</td>
<td>([0, 1023])</td>
<td>([0, 1048755])</td>
<td>(x</td>
<td>1 &lt; x &lt; 10)</td>
<td>(x</td>
</tr>
<tr>
<td>range/set proof creation</td>
<td>(\approx 512)</td>
<td>(\approx 961)</td>
<td>(\approx 1998)</td>
<td>(\approx 35)</td>
<td>(\approx 36)</td>
</tr>
<tr>
<td>range/set proof verification</td>
<td>(\approx 367)</td>
<td>(\approx 599)</td>
<td>(\approx 1116)</td>
<td>(\approx 22)</td>
<td>(\approx 23)</td>
</tr>
</tbody>
</table>
policies using the default instantiation of the three different symmetrical pairings available in JPBC. The fastest performance is achieved by the JPBC Type A curve based Type I pairing, where ticket issuing and verification take \( \approx 2.2s \) and \( \approx 450ms \) respectively.

Table 4 illustrates the impact of different range and set sizes on the computational effort during the ticket issuing phase using the JPBC Type A curve, where \([0, q^k]\) is the range length and \(s\) is the cardinality of the set. It is clear that set membership proofs can be computed much faster than range proofs and their computational cost is independent of the set size whereas for range proofs the computational effort increases linearly with \(k\).

However, range proofs provide an additional benefit which is best illustrated with an example: a young person’s effort increases linearly with \(k\) of the set size whereas for range proofs the computational set membership proofs can be computed much faster than range length and \(k\) would need to return to the CA

However, if the set policy approach had been used, Alice can now prove her age falls within the updated range. As she would not have been eligible for her signed “young person” attribute of 23 years old and the current age attribute of 25, Alice cannot obtain a discount. However, if it is later changed to \(age \in [16, 25]\), Alice can still use her existing age attribute of 23 to obtain a young person discount as she can now prove her age falls within the updated range. However, if the set policy approach had been used, Alice would need to return to the CA to update her credentials as she would not have been eligible for her signed “young person” attribute, previously.

Consequently, for any real system, it is important to look at the trade-off between the flexibility that range policies allow in terms of dynamic updates and their more expensive computational cost.

6 Security Analysis

To demonstrate its security, we need to prove indistinguishability between the behaviours of the real-world adversary \(A\) and the behaviours of the ideal-world adversary \(A'\). Given a real-world adversary \(A\), there exist an ideal-world adversary \(A'\) such that no environment \(E\) can distinguish whether it is interacting with \(A\) or \(A'\). The proof is based on sublemmas where different corrupted parties are considered. The following cases are not considered: (1) the CA is the only honest party; (2) the CA is the only dishonest party; (3) all parties are dishonest; and (4) all parties are honest. The first three do not make a sensible system while the last one is trivially secure. Since the CA needs to know U’s attributes to issue her with her credentials, we assume that CA is honest and fully trusted by the other entities.

In order to prove the indistinguishability between \(\text{Real}_{E,A}(\ell)\) and \(\text{Ideal}_{E,A'}(\ell)\), a sequence of games \(\text{Game}_0, \text{Game}_1, \cdots, \text{Game}_n\) are defined. For each \(\text{Game}_i\), we construct a simulator \(\text{Sim}_i\) that runs \(A\) as a subroutine and provides \(E’s\) view, for \(i = 0, 1, \cdots, n. \text{Hybrid}_{E, \text{Sim}_i}(\ell)\) denotes the probability that \(E\) outputs 1 running in the world provided by \(\text{Sim}_i\). \(\text{Sim}_0\) runs \(A\) and other honest parties in the real-world experiment, hence \(\text{Hybrid}_{E, \text{Sim}_0}(\ell) = \text{Real}_{E,A}(\ell)\) runs \(A'\) in ideal-world experiment, hence \(\text{Hybrid}_{E, \text{Sim}_i}(\ell) = \text{Ideal}_{E,A'}(\ell)\). Therefore,

\[
|\text{Real}_{E,A}(\ell) - \text{Ideal}_{E,A'}(\ell)| \leq |\text{Real}_{E,A}(\ell) - \text{Hybrid}_{E, \text{Sim}_i}(\ell)| + |\text{Hybrid}_{E, \text{Sim}_i} - h\text{Hybrid}_{E, \text{Sim}_2}(\ell)| + \cdots + |\text{Hybrid}_{E, \text{Sim}_{n-1}} - \text{Hybrid}_{E, \text{Sim}_n}(\ell)|.
\]

Theorem 3. Our privacy-preserving electronic ticket scheme with attribute-based credentials described in Fig. 3, Fig. 4, Fig. 5 and Fig. 6 is secure if the \(q\)-strong Diffie-Hellman assumption (\(q\)-SDH) holds on the bilinear group \((e, p, \mathbb{G}, \mathbb{G}_T)\).

The proof of Lemma 1 is given in the full version of this paper [55]. Since anonymity, ticket unlinkability and ticket untransferability are part of user privacy, they are therefore proved by Lemma 2.

Lemma 2. (Seller Security) For all environments \(E\) and all real-world adversaries \(A\) who statically control the ticket seller \(S\) and verifier \(V\), there exists an ideal-world adversary \(A'\) such that \(|\text{Real}_{E,A}(\ell) - \text{Ideal}_{E,A'}(\ell)| \leq 2^t\).

The proof of Lemma 2 is given in the full version of this paper [55]. Since unforgeability, double spending detection and de-anonymization are included in the seller security, they are therefore proved by Lemma 2. Therefore, Theorem 4 is proven because both Lemma 1 and Lemma 2 hold.

7 Future Work and Conclusions

To protect user privacy in e-ticket schemes, various schemes have been proposed but they do not address attribute-based ticketing. Similarly, privacy-preserving attribute-based credential schemes are proposed where attributes can either be elements of a set or within a range but this paper proposes a scheme to support both the use of sets and ranges within the same scheme. The paper has defined such a scheme together with its security model and security proof. The benefit of this scheme is to provide a policy maker with the flexibility to decide which kind of policies should be used. Set policies are computationally more efficient than range policies but the latter could potentially accommodate future policy changes. Currently, our scheme is not suitable for portable devices, e.g. smart phone, tablet, etc., due to its high computational cost and communication overhead.

Our future work will be looking at the impact on the security model and proof when dynamic policy updates are allowed as well as changes to scheme’s implementation to improve its performance, e.g. by pre-computing static values where possible and using the C-based PBC library.
ACKNOWLEDGEMENT

This work has been supported by the EPSRC Project DICE: “Data to Improve the Customer Experience”, EP/N028295/1.

REFERENCES


