This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

Anonymous Single Sign-on with Proxy Re-Verification

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

***Abstract*—An anonymous Single Sign-on (ASSO) scheme allows users to access multiple services anonymously using one credential. We propose a new ASSO scheme, where users can access services anonymously through the use of anonymous credentials and unlinkably through the provision of designated verifiers. Notably, verifiers cannot link a user's service requests even if they collude. The novelty is that when a designated verifier is unavailable, a central authority can authorise new verifiers to authenticate the user on behalf of the original verifier. Furthermore, a central verifier can also be authorised to de-anonymise users and trace their service requests. We formalise the scheme along with a security proof and provide an empirical evaluation of its performance. This scheme can be applied to smart ticketing where minimising the collection of personal information of users is increasingly important to transport organisations due to privacy regulations such as General Data Protection Regulations (GDPR).**

***Index Terms*—Proxy Verification, Anonymous Authentication, Designated Verification, Service Disruption**

I. Introduction

SINGLE Sign-on (SSO) is a mechanism that enables a user to access multiple services using only one credential. Existing SSO solutions include OpenID [1], SAML [2], and Kerberos [3], *etc*. SSO systems can reduce a user's burden on maintaining authentication credentials.

In order to protect users' privacy, anonymous SSO (ASSO) systems were proposed in [4]–[7]. In these systems, a user's personal identifiable information (PII) was considered, but the unlinkability of the user's service requests was not. Recently, Han *et al.* [8] proposed a new ASSO scheme which protects the identity of both the user and her service requests. Their scheme allows users to obtain a ticket from a ticket issuer to access multiple intended services. The ticket consists of a set of authentication tags that can only be validated by designated verifiers. Designated verifiers can validate their corresponding tags and cannot link a user's service requests, even if they collude. A third party, referred to as a central verifier, can de-anonymise a user's identity and trace her service requests.

In a transport application a ticket could represent an intended route of travel (e.g. from A to B to C). Traditionally, in the rail industry, tickets were paper based and hence anonymous. In the context of smart ticketing, which is one of the main digital strategies of the UK rail industry [9], customers' data may be stored when buying tickets. Thus, it will be important to consider passenger privacy in order to minimise the collection of personal information to reflect the requirements of the recently introduced General Data Protection Regulations (GDPR) [10]. Nonetheless, a smart ticketing solution will still need to provide guarantees as to who owns and uses a rail ticket. Using an anonymous scheme such as Han *et al.* [8] means that passenger information leakage between different companies is prevented because each train operating company is considered to be a separate designated verifier. However, the inclusion of a central verifier allows the relevant transport authorities to identify passengers and their journeys. This is important in the case of an emergency to enable transport authorities to know who the passengers using their transport systems are. It could also provide guards on a train access a user's whole journey information in order to provide the best journey advice during travel if appropriate.

In [8], an authentication tag can only be validated by a designated service provider, hence a user cannot access the services if the service provider is off-line or unavailable. In a cloud environment and when a service provider is off-line, a user would expect to be redirected to an alternative provider offering a similar service. While for a transport application (in the case of disruption), a ticket should still be valid and authorised for use on a redirected route. For example, a journey from A to C via B could be redirected to go via D and/or E when B is disrupted. In such cases a user should not be required to buy or change her ticket in order to access the alternative route. Moreover, in practice, the entities who hold the disruption information are disconnected from those who sell tickets. Therefore, rail authorities and train companies should manage and be responsible for the redirected travel routes and disruption information with minimal impact on users.

In this paper, we propose a new ASSO scheme which extends the scheme presented in [8] to allow a central authority to authorise another verifier to act as a proxy and validate the authentication tags for a service provider that is unavailable. In the ticket scenario it thus provides a central authority with the ability to allow a proxy verifier to validate a user's

J. Han is with the Centre for Secure Information Technologies (CSIT), Institute of Electronics, Communications and Information Technology (ECIT), Queen's University Belfast, Belfast, Northern Ireland, BT3 9DT, United Kingdom, and also with Surrey Centre for Cyber Security, University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom. This Research was carried out when he was at Surrey Centre for Cyber Security. E-mail: j.han@qub.ac.uk

L. Chen, S. Schneider, H. Treharne and S. Wesemeyer are affiliated to the Surrey Centre for Cyber Security, University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom
E-mail: {liqun.chen, s.schneider, h.treharne, s.wesemeyer}@surrey.ac.uk

N. Wilson is a Technical Architect at the Rail Delivery Group, United Kingdom. E-mail: nick.wilson@raildeliverygroup.org

ticket. Hence, proxy re-verification does not increase a user's authentication burden in case of a disruption, i.e. a user does not need to change her ticket. Our new scheme also preserves the following features from the original scheme of Han *et al.* [8]:

1) *Multiple Access:* a user can use one ticket to access multiple distinct services;
2) *Anonymity:* a user can obtain a ticket from a ticket issuer without releasing anything about her PII to the ticket issuer, especially, the ticket issuer cannot determine whether two ticket are issued to the same user or two different users;
3) *Unlinkability:* a designated verifier can determine whether a user is authorised to access its service but cannot link a user's different service requests nor collude with other verifiers to link a user's service requests;
4) *Unforgeability:* tickets can only be issued by ticket issuers and cannot be forged by other parties even the central authority;
5) *Traceability:* only the central verifier can de-anonymise a user and trace the identities of the verifiers whose services the user is authorised to access;
6) *Double Spending Detection:* designated verifiers can detect and prevent a user from making two authentication requests using the same authentication tag but cannot de-anonymise the user;

Contributions: Our main contributions in this paper are summarised as follows: (1) an ASSO with proxy re-verification scheme providing the above features is formally constructed; (2) the definition and security model are formalised; (3) the scheme has been implemented and an empirical efficiency analysis is presented; (4) the security of our scheme is formally reduced to well-known complexity assumptions.

The novelty of this paper is to prevent information leakage across multiple verifiers and implement proxy re-verification. To the best of our knowledge, our scheme is the first scheme to support users anonymously and unlinkably authenticating to multiple service providers and allowing authorised proxy verifiers to verify authentication on behalf of an original designated verifier when that verifier is unavailable.

A. Related Work

In this subsection, we review the work which is most closely related to our scheme. Previous authentication schemes mainly address the anonymity of users and implement multiple authentications using one credential.

1) Anonymous Single-Sign-On schemes: Elmufti *et al.* [4] proposed an ASSO scheme which is suitable to the Global System for Mobile communication (GSM). In [4], to access a service, a user needs to generate a new one-time identity and uses it to authenticate to a trusted third party (TTP). If the authentication is successful, the TTP forwards the user's one-time identity to the service provider who provides the service. As a result, the service provider cannot infer the user's real identity from this one-time identity. However, in our scheme, users can authenticate to service providers directly without the need of a TTP.

Han *et al.* [5] proposed a generic construction of dynamic SSO schemes where digital signature, broadcast encryption and zero-knowledge proof are adopted. In [5], after registering with the system, a user obtains a credential which is the encryption of a signature generated by the central authority on a set of service selected by the user and her public key. Consequently, only the service providers whose services have been selected by the user can decrypt the ciphertext and validate the signature. To prevent sharing a credential, a user needs to prove the knowledge of her secret key corresponding to the public key included in the credential. Hence, a user is anonymous only to the service providers who are not included in the credential. Nevertheless, unlike in our scheme, service providers know the user's identity (public key) and link her service requests.

Wang *et al.* [6] proposed an ASSO scheme based on group signatures [11]. When registering to the central authority, a user is issued a group member key. Then, to access a service, a user generates a group signature by using her group member key. A service provider checks whether the user is authorised to access services by validating the correctness of the signature. Furthermore, the central authority can use the open algorithm in the group signature scheme to trace a user's identity. Notably, a user can access all services in the system, while in our scheme a user can only access the selected services.

Lee [7] proposed an efficient ASSO scheme based on Chebyshev Chaotic Maps. When joining the system, an issuer (the smart card processing center) issues temporary secret keys to users and service providers. To access a service, a user interacts with a service provider to generate a session key by using their respective temporary secret keys. A service request is granted if and only if the session key can be generated correctly; otherwise, the request is denied. However, unlike our scheme, each service provider knows the identity of the user accessing his service. Hence, multiple service providers can profile a user's service requests if they collude. Moreover, a user can again access all services in the system, while in our scheme a user can only access the selected services.

2) Proxy Re-Encryption: Mambo and Okamoto [12] introduced the definition of proxy cryptosystems that enable a delegator to delegate the decryption power to a delegatee. Later, Blaze [13] proposed an atomic proxy cryptography scheme where a semi-trusted third party called proxy can convert ciphertexts for one user into ciphertexts for another user if the third party is given a proxy key.

Shamir [14] introduced an identity-based cryptosystem is a public key cryptosystem where a user's public key can be any arbitrary string and her secret key is obtained from a trusted central authority. Boneh and Franklin [15] first proposed a practical identity-based encryption (IBE) scheme based on paring. Green and Ateniese [16] introduced the concept of identity-based proxy re-encryption (IBPRE) where a proxy can convert a ciphertext for the original decryptor to a ciphertext for a designated decryptor if the proxy obtains a re-encryption key from the original decrytor. Han *et al.* [17] classified IBPRE schemes into two types according to the generation of re-encryption keys: (1) re-encryption keys are generated by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

the trusted central authority [18], [19]; (2) re-encryption keys are generated by the original decryptors [16], [20]. In [16], [18]–[20], given a re-encryption key, a proxy can convert all ciphertexts for the original decryptor to ciphertexts for the designated decryptor. The differences between our scheme and IBPRE schemes are: (1) a proxy is not required; (2) a re-key only enables a proxy verifier to validate tickets on behalf the original verifier in a specified period, instead of all tags.

3) Designated Verifier Schemes: Jakobsson *et al.* [21] introduced a designated verifier signature (DVS) scheme which is a digital signature scheme where a signature can only be verified by a single designated verifier. Furthermore, the verifier cannot convince others that a signature is from the real signer since the verifier could have generated the signature by himself. Fan *et al.* [22] presented an attribute-based DVS scheme where a signature can be verified by a group of verifiers whose attributes satisfies specified values. In our scheme, we adopt the high level concept of a designated verifier, i.e. given a valid authentication tag, only the corresponding designated verifier and the authorised proxy verifiers can validate it. The main difference between these DVS schemes [21], [22] and our scheme is that only the designated verifiers can verify a signature in DVS schemes, while in our scheme, everyone can verify a tag's signature generated by the ticket issuer but only the designated verifier of the tag can determine for whom it was generated.

Kuchta *et al.* [23] proposed an identity-based strong designated verifier group signature (ID-SDVGS) scheme that can provide the features of both designated verifier signatures and identity-based group signatures. In this scheme, all entities must obtain secret keys from a trusted third party referred to as "private key generator" (PKG). When joining the group, each user obtains a member credential from the group manager (GM). Then, a user can use her credential to anonymously generate a signature which can only be verified by the designated verifier and can be opened/de-anonymized by the GM. The verifier cannot convince others that the signature is from the real signer since the verifier can generate the signature by himself. However, in our scheme, only the secret keys of ticket verifiers are issued by the central authority. The secret keys of other entities including the ticket issuer, users and the central verifier are generated by themselves. Authentication tags can only be generated by the ticket issuer and its correctness can be publicly verified. Nevertheless, other entities cannot know for whom a tag is generated except the designated verifier.

4) k-time Anonymous Authentication Schemes: Anonymous authentication schemes enable a user to authenticate to a verifier without releasing her PII to the verifier. To limit the authentication time, Teranishi *et al.* [24] proposed a k-time anonymous authentication (k-TAA) scheme where users register with a central authority and obtain an anonymous credential. A verifier generates k authentication tags. For each access, a user proves to the verifier that she has obtained a valid credential from the central authority and selects a fresh authentication tag. As a result, no party can identify a user if she authenticates no more than k times, while any party can identify a user if she authenticates more than k times. In [24], the central authority decides a user's access permission and service verifiers do not have control on the access permissions.

Camenisch *et al.* [25] proposed a periodic k-TAA scheme where a user can anonymously authenticate herself to a service verifier no more than k times in a given time period. The authentication tags automatically refresh every time period. When a user makes an anonymous authentication request, she proves to a verifier that she has obtained a valid credential (CL signature [26]) from the central authority. Lastly, Camenisch *et al.* proposed an identity mixer scheme [27], [28] in which users need to obtain a credential for their attributes. To access a service, a user proves to the service verifier that she has the required attributes.

In all these schemes [24], [25], [27]–[29], authentication is not bound to a particular verifier, whereas in our scheme an authentication tag can only be verified by a designated verifier. Furthermore, k-TAA schemes allow verifiers to de-anonymise a user's identity when she has authenticated more than k times, while in our scheme a service verifier can detect whether a user has used the tag (double spending) but cannot de-anonymise a user's identity. Notably, our scheme allows a central verifier to de-anonymise a user and trace her service requests.

In Table I, we compare our scheme with related ASSO schemes in terms of anonymity, the inclusion of a designated verifier, traceability, re-verification, whether a trusted third party (TTP) is required to authenticate users on behalf of service provers as well as efficiency which mainly considers whether bilinear groups are required or not.

B. Paper Organisation

The remainder of this paper is organised in the following sections. Section II provides a high-level overview of our scheme and its security requirements. Section III introduces the formal definition and security model. Section IV presents the preliminaries for our scheme and a formal construction of our scheme is given in Section V. Section VI and Section VII present the security proof and the performance evaluation of our scheme, respectively. Finally, Section VIII concludes the paper.

II. Scheme Overview and Security Properties

The notation used throughout this paper is summarised in Table II.

Our ASSO with proxy re-verification scheme consists of the following entities:

- a trusted central authority, $\mathcal{CA}$, which initialises the system, issues credentials to other entities in the scheme and authorises proxy verification;
- a user, $\mathcal{U}$, who wants to access some distinct services anonymously and unlinkabily;
- a ticket issuer, $\mathcal{I}$, issues tickets to registered, yet anonymous users for a set of selected services;
- a designated verifier, $\mathcal{V}$, who can only validate the authentication tags generated for him and cannot link a user's service requests;
- an authentication tag, Tag_V, which is bound to a user $\mathcal{U}$ and a designated verifier $\mathcal{V}$ and is used to convince $\mathcal{V}$ that $\mathcal{U}$ is authorised to access its service;

TABLE I
THE COMPARISON BETWEEN OUR SCHEME AND RELATED SCHEMES

Schemes	Anonymity	Designated Verifiers	Traceability	Re-Verification	Trusted Third Party (TTP)	Efficiency (bilinear group)
Elmufti *et al.* [4]	✓	×	✓	×	✓	×
Han *et al.* [5]	×	×	✓	×	×	not applicable
Wang *et al.* [6]	✓	×	✓	×	×	not applicable
Lee [7]	✓	×	×	×	×	×
Han *et al.* [8]	✓	✓	✓	×	×	✓
Our Scheme	✓	✓	✓	✓	×	✓

TABLE II
NOTATION SUMMARY

Notation	Explanations	Notation	Explanations
1^ℓ	A security parameter	$\mathcal{V}_i$	The i-th ticket verifier
$\mathcal{CA}$	Central authority	J_U	The service set of $\mathcal{U}$ consisting of the
$\mathcal{I}$	Ticket issuer		identities of ticket verifiers & ID_{CV}
$\mathcal{V}$	Ticket verifier	PP	Public parameters
$\mathcal{U}$	User	Ps_U	A set of pseudonyms of $\mathcal{U}$
$\mathcal{CV}$	Central verifier	Ps_V	The pseudonym generated for $\mathcal{V}$
ID_I	The identity of $\mathcal{I}$	Tag_V	An authentication tag for $\mathcal{V}$
ID_V	The identity of $\mathcal{V}$	Tag_{CV}	An authentication tag for $\mathcal{CV}$
ID_U	The identity of $\mathcal{U}$	T_U	A ticket issued to $\mathcal{U}$
ID_{CV}	The identity of $\mathcal{CV}$	$\|X\|$	The cardinality of the set X
$\epsilon(\ell)$	A negligible function in ℓ	$x \stackrel{R}{\leftarrow} X$	x is randomly selected from the set X
σ_I	The credential of $\mathcal{I}$	$A(x) \to y$	y is computed by running the
σ_V	The credential of $\mathcal{V}$		algorithm $A(\cdot)$ with input x
σ_U	The credential of $\mathcal{U}$	$\mathcal{KG}(1^\ell)$	A secret-public key pair generation algorithm
σ_{CV}	The credential of $\mathcal{CV}$	$\mathcal{BG}(1^\ell)$	A bilinear group generation algorithm
MSK	Master Secret Key	PPT	Probable polynomial-time
H_1, H_2, H_3	Cryptographic hash functions	p	A prime number

- a ticket, T_U, which consists of a set of authentication tags generated for the designated verifiers of the requested services;
- a central verifier, $\mathcal{CV}$, which is another trusted third party which, given a ticket T_U, can de-anonymise the identities of the user and trace her service requests.

A. Overview of proposed scheme

A simplified pictorial description of our scheme is presented in Fig. 1. $\mathcal{CA}$ initialises the system. When joining the system, $\mathcal{I}$, $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{CV}$ authenticate to the $\mathcal{CA}$ and obtain their credentials from $\mathcal{CA}$. To buy a ticket, $\mathcal{U}$ sends her service information J_U consisting of a set of verifiers' identities ID_V to $\mathcal{I}$. Subsequently, $\mathcal{I}$ generates a ticket T_U for $\mathcal{U}$. The ticket comprises a set of tags $T_U = \{Tag_V | ID_V \in J_U\} \cup \{Tag_{CV}\}$ which can only be validated by the corresponding designated verifiers. When being validated by $\mathcal{V}$, $\mathcal{U}$ sends the corresponding tag Tag_V to $\mathcal{V}$. In the case that $\mathcal{U}$'s service information needs to be traced, $\mathcal{CV}$ is allowed to trace the whole service information of $\mathcal{U}$ given a ticket T_U. Especially, when the original verifier $\mathcal{V}$ is unavailable, $\mathcal{CA}$ can authorise a new verifier $\mathcal{V}'$ to validate the tag on behalf of $\mathcal{V}$.

B. Security Properties of Our Scheme

Having defined the different entities and described how they interact, we now list the security properties of our scheme:
Anonymity: a user can obtain a ticket from a ticket issuer anonymously;
Unlinkability: a designated verifier cannot link a user's different service requests nor collude with other verifiers to link a user's service requests;
Unforgeability: tickets are generated by ticket issuers and cannot be forged by other parties even the central authority;
Traceability: given a valid ticket, $\mathcal{CV}$ can de-anonymise the ticket holder and trace her service requests;
Proxy Re-verification: in the case that a designated verifier $\mathcal{V}$ is unavailable, $\mathcal{CA}$ can assign one or more verifiers $\mathcal{V}'$ to validate a user's tag designated for $\mathcal{V}$;
Double Spending: a designated verifier can detect whether a tag has been used or not, but cannot de-anonymise the user.

III. FORMAL DEFINITION AND SECURITY REQUIREMENT

In this section, we review the formal definition and security requirement of ASSO with proxy re-verification. The formal definition of ASSO with proxy re-verification is defined using a series of algorithms and is given in the full version of this paper (see Section III in [30]).

A. Security Requirements

The security model of our scheme is defined by the following three games.

Unforgeability. This is used to define the unforgeability of tickets, namely even if users, verifiers and the central verifier collude, they cannot forge a valid ticket. This game is formalised in the full version (see Section III in [30]).

Unlinkability. This is used to define the unlinkability, namely even if some ticket verifiers collude with potential users, they cannot profile the whole service information of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

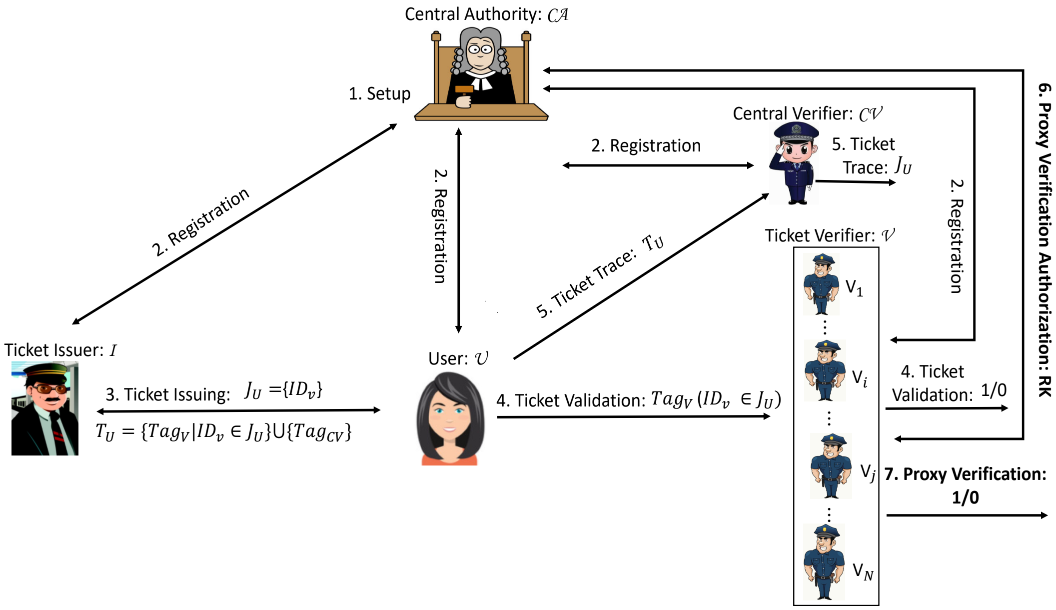

Fig. 1: Pictorial description of our scheme

other users. We assume that $\mathcal{I}$ and $\mathcal{CV}$ cannot be compromised because they can know a user's whole service information by themselves. The game is formalised in the full version (see Section III in [30]).

Traceability. This is used to formalise the traceability of tickets, namely even if a group of users collude, they cannot generate a ticket which $\mathcal{CV}$ would not catch as belonging to some member of the colluding group. We suppose that the ticket issuer is honest. This game is formalised in the full version (see Section III in [30]).

IV. Preliminaries

In this section, the preliminaries used in this paper are introduced.

A. Bilinear Group

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_\tau$ be cyclic groups with prime order p. A map $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_\tau$ is a bilinear map/pairing if it satisfies the following properties [15]: (1) Bilinearity: For all $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$ and $x, y \in \mathbb{Z}_p$, $e(g^x, h^y) = e(g^y, h^x) = e(g, h)^{xy}$; (2) Non-degeneration: For all $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$, $e(g, h) \neq 1_\tau$ where 1_τ is the identity element in $\mathbb{G}_\tau$; (3) Computability: For all $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$, there exists an efficient algorithm to compute $e(g, h)$.

Let $\mathcal{BG}(1^\ell) \longrightarrow (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ be a bilinear group generator which takes as input a security parameter 1^ℓ and outputs a bilinear group $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$. Bilinear maps can be divided into three types [31]: Type-I: $\mathbb{G}_1 = \mathbb{G}_2$; Type-II: $\mathbb{G}_1 \neq \mathbb{G}_2$ but there exist an efficient map: $\phi : \mathbb{G}_2 \longrightarrow \mathbb{G}_1$; Type-III: $\mathbb{G}_1 \neq \mathbb{G}_2$ but there is no efficient map between $\mathbb{G}_1$ and $\mathbb{G}_2$. Type-III pairings are the most efficient pairings [32]. Our scheme is based on Type-III pairings where the size of elements in $\mathbb{G}_1$ is short (160 bits).

B. Complexity Assumptions

Definition 1: (Discrete Logarithm (DL) Assumption [33]) Let $\mathbb{G}$ be a cyclic group with prime order p and g be a generator of $\mathbb{G}$. Given $Y \in \mathbb{G}$, we say that the DL assumption holds on $\mathbb{G}$ if all PPT adversaries can output a number $x \in \mathbb{Z}_p$ such that $Y = g^x$ with a negligible advantage, namely $Adv_{\mathcal{A}}^{DL} = \Pr[Y = g^x | \mathcal{A}(p, g, \mathbb{G}, Y) \rightarrow x] \leq \epsilon(\ell)$.
The proof of the traceability property of our scheme is reduced to the DL assumption.

Definition 2: (Decisional Bilinear Diffie-Hellman (DBDH) Assumption [15]) Let $\mathcal{BG}(1^\ell) \longrightarrow (e, p, \mathbb{G}, \mathbb{G}_\tau)$ where $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ and g be a generator of $\mathbb{G}$. Suppose that $a, b, c \xleftarrow{R} \mathbb{Z}_p$. Given a tuple $\mathbb{T} = (g, g^a, g^b, g^c, \Upsilon)$, we say that the DBDH assumption holds on $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ if all PPT adversary $\mathcal{A}$ can distinguish $\Upsilon = e(g, g)^{abc}$ from a random element $R \in \mathbb{G}_\tau$ with a negligible advantage, namely $Adv_{\mathcal{A}}^{DBDH} = \Big| \Pr[\mathcal{A}(\mathbb{T}, \Upsilon = e(g, g)^{abc}) = 1] - \Pr[\mathcal{A}(\mathbb{T}, \Upsilon = R) = 1] \Big| \leq \epsilon(\ell)$.

The security of the Boneh-Franklin IBE used to implement flexible verification was reduced to the DBDH assumption.

Definition 3: (Decisional asymmetric Bilinear Diffie-Hellman (DaBDH) Assumption [32]) Let $\mathcal{BG}(1^\ell) \longrightarrow (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ and $g, \mathfrak{g}$ be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Suppose that $a, b, c \xleftarrow{R} \mathbb{Z}_p$. Given a tuple $\mathbb{T} = (g, \mathfrak{g}, g^a, g^b, g^c, \mathfrak{g}^b, \mathfrak{g}^c, \Upsilon)$, we say that the DaBDH assumption holds on $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ if all PPT adversaries can distinguish $\Upsilon = e(g, \mathfrak{g})^{abc}$ from a random element $R \in \mathbb{G}_\tau$ with a negligible advantage, namely $Adv_{\mathcal{A}}^{DaBDH} = \Big| \Pr[\mathcal{A}(\mathbb{T}, \Upsilon = e(g, \mathfrak{g})^{abc}) = 1] - \Pr[\mathcal{A}(\mathbb{T}, \Upsilon = R) = 1] \Big| \leq \epsilon(\ell)$.

The DaBDH assumption is used to prove the unlinkablity of our scheme.

Definition 4: ((JoC Version) q-Strong Diffie-Hellman (JoC-q-SDH) Assumption [34]) Let $\mathcal{BG}(1^\ell) \longrightarrow (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$. Given a $(q+3)$-tuple $(g, g^x, \cdots, g^{x^q}, \mathfrak{g}, \mathfrak{g}^x) \in \mathbb{G}_1^{q+1} \times \mathbb{G}_2^2$, we say that the JoC-$q$-SDH assumption holds on $(e, p, \mathbb{G}_1, \mathbb{G}_2,$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

$\mathbb{G}_\tau$) if all PPT adversaries $\mathcal{A}$ can output $(c, g^{\frac{1}{x+c}}) \in \mathbb{Z}_p \times \mathbb{G}_1$ with a negligible advantage, namely $Adv_{\mathcal{A}}^{\text{JOC-q-SDH}} = \Pr\left[(c, g^{\frac{1}{x+c}}) \leftarrow \mathcal{A}(g, g^x, \cdots, g^{x^q}, \mathfrak{g}, \mathfrak{g}^x)\right] \leq \epsilon(\ell)$, where $c \in \mathbb{Z}_p \setminus \{-x\}$.

The unforgeability of our scheme is reduced to the JoC-q-SDH assumption.

C. Zero-Knowledge Proof

We follow the definition introduced by Camenish and Stadler in [35] and formalised by Camenish *et al.* in [36]. By PoK:$\{(x_1, x_2, x_3) : \Upsilon = g^{x_1}h^{x_2} \wedge \tilde{\Upsilon} = \mathfrak{g}^{x_1}\mathfrak{h}^{x_3}\}$, we denote a zero knowledge proof on knowledge of integers x_1, x_2 and x_3 such that $\Upsilon = g^{x_1}h^{x_2}$ and $\tilde{\Upsilon} = \mathfrak{g}^{x_1}\mathfrak{h}^{x_3}$ hold on the groups $\mathbb{G} = \langle g \rangle = \langle h \rangle$ and $\tilde{\mathbb{G}} = \langle \mathfrak{g} \rangle = \langle \mathfrak{h} \rangle$, respectively. The convention is that the letters in the parenthesis (x_1, x_2, x_3) stand for the knowledge which is being proven, while the other parameters are known by the verifier.

D. BBS+ Signature

This signature was proposed by Au *et al.* [37]. Its security was reduced to the q-SDH assumption in Type-II pairing setting in [37]. Recently, Camenisch *et al.* [38] reduced its security to the JoC-q-SDH assumption in Type-III pairing setting.

Theorem 1: (Camenisch *et al.* [38])The BBS+ signature is existentially unforgeable against adaptive chosen message attacks (EU-CMA) if the JoC-q-SDH assumption holds on the bilinear group $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$.

E. Boneh-Franklin Identity-Based Encryption

Boneh and Franklin [15] proposed the first IBE scheme based on the Type-I pairing: $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_\tau$.

Theorem 2: (Boneh and Franklin [15]) This IBE scheme is secure against chosen-plaintext attack (CPA) if the DBDH assumption holds on the bilinear map group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$.

Abdalla *et al.* [39] observed that Boneh-Franklin IBE [15] is an anonymous IBE scheme where ciphertext does not release the identity of the receiver. Chatterjee and Menezes [32] transferred Boneh-Franklin IBE scheme from Type-I pairing setting to Type-III pairing setting, and claimed that the security of the transferred scheme can be reduced to DaBDH assumption. In this paper, the Boneh-Franklin [15] IBE scheme is applied to implement proxy re-verification.

V. Construction of our scheme

A. Formal Construction

The formal construction of our ASSO with proxy re-verification scheme including messages sent between its entities and their relevant computations is presented in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7 and Fig. 8. Notably, Fig. 7 and Fig. 8 are new in our scheme compared to Han *et al.*'s construction in [8] and Fig. 3, Fig. 4 have been modified to reflect the IBPRE scheme used.

B. High-Level Overview

At a high level, our scheme works as follows.

Setup. $\mathcal{CA}$ initializes the systems and generates a master secret key $MSK = (\alpha, \beta)$ and the corresponding public parameters PP. Actually, α is used to issue credentials to $\mathcal{I}$, $\mathcal{U}$ and $\mathcal{CV}$ when they join the system, while β is used to issue secret keys to $\mathcal{V}$s.

Registration. When joining the system, $\mathcal{I}$, $\mathcal{U}$ and $\mathcal{CV}$ generate their secret-public key pairs $(x_i, Y_I, \tilde{Y}_I)$, (x_u, Y_U) and (x_{cv}, Y_{CV}), and register with the $\mathcal{CA}$ by sending their identities (ID_I, ID_U, ID_{CV}) and public keys $((Y_I, \tilde{Y}_I), Y_U, Y_{CV})$, respectively. Finally, $\mathcal{I}$, $\mathcal{U}$ and $\mathcal{CV}$ obtain their credentials (d_i, e_i, σ_I), (d_u, e_u, σ_U) and $(d_{cv}, e_{cv}, \sigma_{CV})$ from $\mathcal{CA}$, respectively. Note, (d_i, e_i, σ_I), (d_u, e_u, σ_U) and $(d_{cv}, e_{cv}, \sigma_{CV})$ are generated by $\mathcal{CA}$ using the master secret key α and are BBS+ signatures on the public keys Y_I, Y_U and Y_{CV}, respectively. When joining the system, $\mathcal{V}$s only submit their identities to $\mathcal{CA}$. $\mathcal{CA}$ uses the master secret key β to generate a secret key SK_V for the identity ID_V of $\mathcal{V}$. This is one of the main differences in the scheme's construction compared to Han *et al.* [8] where the verifiers generate their own secret-public key pairs. Moving this generation to the $\mathcal{CA}$ is required to facilitate the proxy re-verification. Furthermore, the $\mathcal{CA}$ generates a credential (d_v, e_v, σ_V) for $\mathcal{V}$ which is a BBS+ signature on ID_V. $\mathcal{CV}$ stores $((d_v, e_v, \sigma_V), SK_V)$, and sends them to $\mathcal{V}$.

Ticket Issuing. To buy a ticket, $\mathcal{U}$ determines her service information J_U consisting of the identities of the corresponding $\mathcal{V}$ whose services $\mathcal{U}$ wants to access. Furthermore, for each $ID_V \in J_U$, $\mathcal{U}$ generates a pseudonym (P_V, Q_V) using her secret key and proves to $\mathcal{I}$ that she is a registered user and the pseudonyms are generated correctly ($\prod_U^1$). If the proof is correct, for each $ID_V \in J_U$, $\mathcal{I}$ generates an authentication tag $Tag_V = ((P_V, Q_V), (E_V^1, E_V^2, E_V^3, K_V, Text_1, Text_2), (s_v, w_v, z_v, Z_V))$. Within the tag (E_V^1, E_V^2) are used by $\mathcal{V}$ to validate Tag_V while (E_V^1, E_V^2, E_V^3) are used by a proxy verifier $\mathcal{V}'$ to validate Tag_V on behalf of $\mathcal{V}$ in a specified time period TP. Since TP is embedded in E_V^3 it is used to restrict the time of proxy re-verification to reflect that time period. In a rail application, a TP could be the travel day printed on the ticket (e.g. September 1, 2018) and can be decided by the ticket issuer.

Additionally, within the tag $((P_V, Q_V), E_V^2, K_V)$ are used by $\mathcal{CV}$ to de-anonymize $\mathcal{U}$'s identity and trace her service requests. Note also that s_v is the serial number of Tag_V and (w_v, z_v, Z_V) is a BBS+ signature on s_v. To prevent $\mathcal{U}$ from combing the authentication tags in different tickets, $\mathcal{I}$ generates another BBS+ signature (w, z, Z) on the ticket issue number $s = H_1(s_1||s_2||\cdots||s_{|J_U|})$. The ticket is $T_U = \{Tag_V | ID_V \in J_U\} \cup (s, w, z, Z)$.

Ticket Validation. When validating a ticket, $\mathcal{V}$ sends its identity ID_V to $\mathcal{U}$. $\mathcal{U}$ selects the corresponding tag Tag_V, and then sends it to $\mathcal{V}$ with a proof of the knowledge ($\prod_U^2$) of the secrets included in the pseudonyms (P_V, Q_V). $\mathcal{V}$ validates the tag Tag_V by checking the proof and the signature. However, in the case that $\mathcal{U}$ needs to confirm whether $\mathcal{V}$ is a designated verifier, $\mathcal{V}$ sends ID_V and his credential (d_v, e_v, σ_V) to $\mathcal{U}$. Then, $\mathcal{U}$ checks $e(\sigma_V, Y_A \mathfrak{g}^{e_v}) \stackrel{?}{=} e(g_1 g_2^{d_v} \tilde{g}^{H_1(ID_V)})$. If it holds,

$\mathcal{V}$ is a designated verifier; otherwise, $\mathcal{V}$ is not. In this paper, we assume that $\mathcal{V}$ is clear and $\mathcal{U}$ does not need to confirm it. For example in the rail scenario, the verifier/station is clear to $\mathcal{U}$.

Ticket Trace. To de-anonymize a user and trace her service requests, $\mathcal{CV}$ initialises a set Ω_U. Given a ticket T_U, $\mathcal{CV}$ uses his secret key to de-anonymize $\mathcal{U}$ from the pseudonyms (P_V, Q_V) for $ID_V \in J_U$ and traces the service request from (E_V^2, K_V). Finally, $\mathcal{CV}$ can determine $\mathcal{U}$'s service requests by recording all the identities $ID_V \in \Omega_U$.

Proxy Key Generation. In the case that a verifier $\mathcal{V}$ is unavailable, $\mathcal{CA}$ can authorize a proxy verifier $\mathcal{V}'$ to validate the tag Tag_V in a ticket T_U by issuing a re-key $RK_{\mathcal{V}\rightarrow\mathcal{V}'}$ to $\mathcal{V}'$. $RK_{\mathcal{V}\rightarrow\mathcal{V}'}$ is generated by using both secret keys SK_V and $SK_{V'}$. To limit the proxy verification period, a time period TP, which is embedded in E_V^3 during the Ticket Issuing, is also embedded in $RK_{\mathcal{V}\rightarrow\mathcal{V}'}$so that only tickets within that TP period can be validated by the proxy verifier. To prevent an unauthorised verifier from claiming to be a legal proxy, the $\mathcal{CA}$ or another trusted third party should broadcast the proxy information $(ID_{V'})$ to both $\mathcal{U}$ and $\mathcal{V}'$. For example, in a rail scenario, when a station $\mathcal{V}$ is unavailable and an alternative plan is provided, both the user $\mathcal{U}$ and the proxy $\mathcal{V}'$ need to be notified.

Proxy Ticket Validation. To verify a tag Tag_V on behalf of $\mathcal{V}$, $\mathcal{V}'$ sends the identity ID'_V to $\mathcal{U}$. $\mathcal{U}$ returns the tag Tag_V to $\mathcal{V}'$ and proves the knowledge included in Tag_V. If the proof is correct, $\mathcal{V}'$ validates Tag_V by using his secret key $SK_{V'}$ and the re-key $RK_{\mathcal{V}\leftarrow\mathcal{V}'}$. Both the user and the proxy verifier $\mathcal{V}'$ know that $\mathcal{V}'$ is a proxy for the verifier $\mathcal{V}$ as discussed above. For example, in a transport application a public announcement would identify the alternative route and hence the corresponding proxy verifier to the user.

C. Correctness

The correctness of our scheme and the details of the zero-knowledge proofs of $\prod_U^1$ and $\prod_U^2$ are provided in in the full version (see Section V and Appendix A in [30]).

VI. Security Analysis

In this section, the security of our scheme is formally proven.

Theorem 3: Our scheme described in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7 and Fig. 8 is $(\varrho, \epsilon'(\ell))$-unforgeable if and only if the $(q, \epsilon(\ell))$-JoC-q-SDH assumption holds on the bilinear group $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ and H_1, H_2 and H_3 are secure cryptographic hash functions, where ϱ is the number of ticket issuing queries made by the adversary $\mathcal{A}$, $\varrho < q$ and $\epsilon(\ell) \geq (\frac{p-q}{p} + \frac{1}{p} + \frac{p-1}{p^3})\epsilon'(\ell)$.

Proof (Sketch): The unforgeability of our scheme is due to the unforgeability of the BBS+ signature [38]. The strategy used to prove the unforgeability of our scheme is as follows. If there exists an adversary $\mathcal{A}$ which can break the unforgeability of our scheme, we can construct an algorithm $\mathcal{B}$ which can use $\mathcal{A}$ to break the JoC-q-SDH assumption. Let $f(x) = \prod_{i=1}^{q-1}(x + \pi_i) = \sum_{i=0}^{q-1}\theta_i x^i$ where $\pi_i \xleftarrow{R} \mathbb{Z}_p$ for $i = 1, 2, \cdots, q-1$. Given $(g, g^x, \cdots, g^{x^q}, \mathfrak{g}, \mathfrak{g}^x) \in \mathbb{G}_1^{q+1} \times \mathbb{G}_2^2$, let $\tilde{g} = g^{f(x)} = \prod_{i=0}^{q-1}(g^{x^i})^{\theta_i}$ and $\tilde{g}^x = \prod_{i=0}^{q-1}(g^{x^{i+1}})^{\theta_i}$. $\mathcal{B}$ sets the public key of the ticket issuer $\mathcal{I}$ as $(Y_I = \tilde{g}^x, \tilde{Y}_I = \mathfrak{g}^x)$, an thus implicitly sets the secret key of $\mathcal{I}$ as $x_i = x$. In our scheme, a ticket is a BBS+ signature on the selected services $(ID_V \in J_U)$, pseudonyms $((P_V, Q_V)_{ID_V \in J_U})$, authentication tags $((E_V^1, E_V^2, E_V^3, K_V)_{ID_V \in J_U})$ and auxiliary information $(Text_1$ and $Text_2)$. When the adversary $\mathcal{A}$ makes a ticket issuing query, a ticket is generated by using the technique deveoped in the proof of BBS+ signature [38]. For other queries, $\mathcal{B}$ first generates secret keys for $\mathcal{CA}$, $\mathcal{V}$ and $\mathcal{CV}$, and then uses them to respond $\mathcal{A}$'s queries. Since the unforgeability of BBS+ signature was reduced to JoC-q-SDH assumption, $\mathcal{B}$ can use $\mathcal{A}$ to break the JoC-q-SDH assumptions if $\mathcal{A}$ can forge a ticket. ■

The formal proof of *Theorem 3* is presented in the full version (see Section VI in [30]).

Theorem 4: Our scheme described in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7 and Fig. 8 is $\epsilon'(\ell)$-unlinkable if and only if the $\epsilon(\ell)$-DaBDH assumption holds on the bilinear group $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$, H_1, H_2 and H_3 are secure cryptographic hash functions, and H_2 is a random oracle, where $\epsilon'(\ell) \geq \frac{\epsilon(\ell)}{2\mathfrak{e}(1+q_{VA})}$, $\mathfrak{e} \approx 2.71$ is the natural logarithm, q_{VA} is the number of ticket verifiers selected by $\mathcal{A}$ to query the Ticket-Verifier-Reg oracle.

Proof (Sketch): The unlinkability of our scheme is due to the security and anonymity of the Boneh-Franklin IBE scheme [15]. The strategy used to prove the unlinkability of our scheme is as follows. If there exists an adversary $\mathcal{A}$ which can break the unlinkability of our scheme, we can construct an algorithm $\mathcal{B}$ which can use $\mathcal{A}$ to break the DaBDH assumption. Given $(g, \mathfrak{g}, g^a, g^b, g^c, \mathfrak{g}^b, \mathfrak{g}^c, \Upsilon)$, $\mathcal{B}$ selects $\alpha, \gamma, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5 \xleftarrow{R} \mathbb{Z}_p$, and computes $\tilde{g} = g^\gamma$, $g_1 = g^{\gamma_1}$, $g_2 = g^{\gamma_2}$, $g_3 = g^{\gamma_3}$, $\vartheta_1 = g^{\gamma_4}$, $\vartheta_5 = g^{\gamma_5}$, $Y_A = \mathfrak{g}^\alpha$, $\tilde{Y}_A = (g^b)^\gamma$. $\mathcal{B}$ selects two hash functions: $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_3 : \{0,1\}^* \rightarrow \{0,1\}^{\ell'}$ and sets H_2 as a random oracle defined in [15]. $\mathcal{B}$ sets the public parameters as $PP = (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau, \tilde{g}, g, g_1, g_2, g_3, \mathfrak{g}, \vartheta_1, \vartheta_2, Y_A, \tilde{Y}_A, H_1, H_3)$, an thus implicitly sets the secret key of $\mathcal{CA}$ as $MSK = (\alpha, b)$. We suppose that both $\mathcal{I}$ and $\mathcal{CV}$ cannot be corrupted by $\mathcal{A}$ as each of them can link the services in a ticket. $\mathcal{B}$ sets the secret-public key pairs of $\mathcal{I}$ and $\mathcal{CV}$ to be $(x_i, (Y_I \tilde{Y}_I))$ and (x_{cv}, Y_{CV}), respectively, where $x_i, x_{cv} \xleftarrow{R} \mathbb{Z}_p$, $Y_I = g^{x_i}$, $\tilde{Y}_I = \mathfrak{g}^{x_i}$ and $Y_{CV} = g^{x_{cv}}$. When $\mathcal{A}$ makes a H_2 query on an identity ID_V, $\mathcal{B}$ responses $\mathcal{A}$ with a value $H_2(ID_V)$ by using the technique developed in [15] where g^c is used. When $\mathcal{A}$ makes a registration query for a verifier $\mathcal{V}$, $\mathcal{B}$ first generates a secret key SK_V for $\mathcal{V}$ by using the key generation technique introduced in [15], and then generates a BBS+ signature σ_V on $H_2(ID_V)$ by using the secret key α. $\mathcal{B}$ responses $\mathcal{A}$ with (SK_V, σ_V). When $\mathcal{A}$ makes registration queries for $\mathcal{I}$ and $\mathcal{CV}$, $\mathcal{B}$ generates BBS+ signatures σ_I on Y_I and σ_{CV} on Y_{CV} by using α, and responds $\mathcal{A}$ with $((Y_I, \tilde{Y}_I), \sigma_I)$ and (Y_{CV}, σ_{CV}), respectively. When $\mathcal{A}$ makes a registration query on a user $\mathcal{U}$ with public key Y_U, $\mathcal{B}$ responses $\mathcal{A}$ a BBS+ signature on Y_U by using α. When $\mathcal{A}$ makes a ticket issuing query, $\mathcal{B}$ responses $\mathcal{A}$ with a BBS+ signature on the pseudonyms, authentication tags, and auxiliary information by using $\mathcal{I}$'s secret key x_i.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

Setup(1^λ)

$\mathcal{CA}$ runs $\mathcal{BG}(1^\ell) \rightarrow (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ with $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_\tau$. Let $\tilde{g}, \bar{g}, g_1, g_2, g_3$ be generators of $\mathbb{G}_1$ and $\mathfrak{g}$ be generators of $\mathbb{G}_2$. Suppose that $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $H_2 : \{0,1\} \rightarrow \mathbb{G}_2$ and $H_3 : \{0,1\}^* \rightarrow \{0,1\}^{\ell'}$ $(\ell' \leq \ell)$ are cryptographic hash functions. $\mathcal{CA}$ selects $\alpha, \beta, \xleftarrow{R} \mathbb{Z}_p$ and $\vartheta_1, \vartheta_2 \xleftarrow{R} \mathbb{G}_2$. $\mathcal{CA}$ computes $Y_A = \mathfrak{g}^\alpha$ and $\tilde{Y}_A = \tilde{g}^\beta$. The master secret key is $MSK = (\alpha, \beta)$ and the public parameters are $PP = (e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau, \tilde{g}, \bar{g}, g_1, g_2, g_3, \mathfrak{g}, \vartheta_1, \vartheta_2, Y_A, \tilde{Y}_A, H_1, H_2, H_3)$.

Fig. 2: Setup Algorithm

Ticket-Issuer-Reg($\mathcal{I}(x_i, Y_I, \tilde{Y}_I, ID_I, PP) \leftrightarrow \mathcal{CA}(MSK, PP)$)

Ticket Issuer: $\mathcal{I}$		Central Authority: $\mathcal{CA}$
Select $x_i \xleftarrow{R} \mathbb{Z}_p$ and compute $Y_I = \tilde{g}^{x_i}$, $\tilde{Y}_I = \mathfrak{g}^{x_i}$		
The secret-public key pair is $(x_i, Y_I, \tilde{Y}_I)$.	$\xrightarrow{ID_I, Y_I, \tilde{Y}_I}$	Select $d_i, e_i \xleftarrow{R} \mathbb{Z}_p$ and compute
Verify: $e(\sigma_I, Y_A \mathfrak{g}^{e_i}) \stackrel{?}{=} e(g_1 g_2^{d_i} Y_I, \mathfrak{g})$	$\xleftarrow{\sigma_I, d_i, e_i}$	$\sigma_I = (g_1 g_2^{d_i} Y_I)^{\frac{1}{\alpha + e_i}}$.
Keep the credential as $Cred_I = (d_i, e_i, \sigma_I)$		Store $(ID_I, Y_I, \tilde{Y}_I, (d_i, e_i, \sigma_I))$.

Ticket-Verifier-Reg($\mathcal{V}(ID_V, PP) \leftrightarrow \mathcal{CA}(MSK, PP)$)

Ticket-Verifier: $\mathcal{V}$		Central Authority: $\mathcal{CA}$
	$\xrightarrow{ID_V}$	Select $d_v, e_v \xleftarrow{R} \mathbb{Z}_p$ and compute
Verify: $e(\sigma_V, Y_A \mathfrak{g}^{e_v}) \stackrel{?}{=} e(g_1 g_2^{d_v} \tilde{g}^{H_1(ID_V)}, \mathfrak{g})$;	$\xleftarrow[SK_V]{\sigma_V, d_v, e_v}$	$\sigma_V = (g_1 g_2^{d_v} \tilde{g}^{H_1(ID_V)})^{\frac{1}{\alpha + e_v}}$,
$e(\tilde{g}, SK_V) \stackrel{?}{=} e(\tilde{Y}_A, H_2(ID_V))$;		$SK_V = H_2(ID_V)^\beta$.
Keep the credential as $Cred_V = (r_v, e_v, \sigma_V)$ and the secret key as SK_V.		Store $(ID_V, (d_v, e_v, \sigma_V), SK_V)$.

User-Reg($\mathcal{U}(x_u, Y_U, ID_U, PP) \leftrightarrow \mathcal{CA}(MSK, PP)$)

User: $\mathcal{U}$		Central Authority: $\mathcal{CA}$
Select $x_u \xleftarrow{R} \mathbb{Z}_p$, and compute $Y_U = \tilde{g}^{x_u}$		
This secret-public key pair is (x_u, Y_U)	$\xrightarrow{ID_U, Y_U}$	Select $d_u, e_u \xleftarrow{R} \mathbb{Z}_p$ and compute
Verify: $e(\sigma_U, Y_U \mathfrak{g}^{e_u}) \stackrel{?}{=} e(g_1 g_2^{d_u} Y_U, \mathfrak{g})$	$\xleftarrow{\sigma_U, d_u, e_u}$	$\sigma_U = (g_1 g_2^{d_u} Y_U)^{\frac{1}{\alpha + e_u}}$
Keep the credential as $Cred_U = (d_u, e_u, \sigma_U)$.		Store $(ID_U, Y_U, (d_u, e_u, \sigma_U))$

Central-Verifier-Reg($\mathcal{CV}(x_{cv}, Y_{CV}, ID_{CV}, PP) \leftrightarrow \mathcal{CA}(MSK, PP)$)

Central Verifier: $\mathcal{CV}$		Central Authority: $\mathcal{CA}$
Select $x_{cv} \xleftarrow{R} \mathbb{Z}_p$, and compute $Y_{CV} = \tilde{g}^{x_{cv}}$.		
The secret-public key pair is (x_{cv}, Y_{CV})	$\xrightarrow{ID_{CV}, Y_{CV}}$	Select $d_{cv}, e_{cv} \xleftarrow{R} \mathbb{Z}_p$ and compute
Verify: $e(\sigma_{cv}, Y_A \mathfrak{g}^{e_{cv}}) \stackrel{?}{=} e(g_1 g_2^{d_{cv}} Y_{CV}, \mathfrak{g})$	$\xleftarrow{\sigma_{CV}, d_{cv}, e_{cv}}$	$\sigma_{CV} = (g_1 g_2^{d_{cv}} Y_{CV})^{\frac{1}{\alpha + e_{cv}}}$
Keep the credential as $Cred_{CV} = (d_{cv}, e_{cv}, \sigma_{CV})$		Store $(ID_{CV}, Y_{CV}, (d_{cv}, e_{cv}, \sigma_{CV}))$

Fig. 3: Registration Algorithm

When $\mathcal{A}$ makes a ticket trace query, $\mathcal{B}$ uses $\mathcal{CV}$'s secret key x_{cv} to decrypt (E_V^2, K_V), and responds $\mathcal{A}$ with the identities of verifiers included in the ticket. When $\mathcal{A}$ makes a proxy key generation query on $(\mathcal{V}, \mathcal{V}')$, $\mathcal{B}$ responds $\mathcal{A}$ with a re-key $R_{\mathcal{V} \rightarrow \mathcal{V}'}$ by using the secret keys SK_V and $SK_{V'}$ recorded in the hash table H_2^{list}, and stores it into a table PQ. When $\mathcal{A}$ makes a proxy ticket validation query, $\mathcal{A}$ responses $\mathcal{A}$ the identities of verifiers included in the ticket by using the re-key recorded in the table PQ. After queries, $\mathcal{A}$ outputs two pseudonym-verifier pairs $((P_{V_0}^*, Q_{V_0}^*), ID_{V_0^*})$ and $((P_{V_1}^*, Q_{V_1}^*), ID_{V_1^*})$ with the limitations: (1) both V_0^* and V_1^* are not corrupted by $\mathcal{A}$; and (2) both $RK_{\mathcal{V}_0^* \rightarrow \mathcal{V}_1^*}$ and $RK_{\mathcal{V}_1^* \rightarrow \mathcal{V}_0^*}$ are not in the table PQ. $\mathcal{B}$ randomly selects $((P_{V_\rho^*}, Q_{V_\rho^*}), ID_{V_\rho^*})$ where $\rho \in \{0,1\}$. $\mathcal{B}$ embeds g^a into the challenged authentication tag $(E_{V_\rho^*}^1, E_{V_\rho^*}^2)$ by using the challenged ciphertext generation technique in [15]. $E_{V_\rho^*}^3$ and $K_{V_\rho^*}$ are generated by using x_{cv}. The signature on the pseudonym, authentication tag and auxiliary information is generated by using x_i. $\mathcal{B}$ sends the challenged ticket T_U^* to $\mathcal{A}$. $\mathcal{A}$ outputs his guess ρ' on ρ. If $\rho' = \rho$, $\mathcal{B}$ can use $\mathcal{A}$ to break the DaBDH assumption since the Boneh-Franklin IBE scheme is secure and anonymous under the DaBDH assumption [15]. The probability is obtained by using the calculation method introduced in [15].

■

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

Ticket-Issuing $(\mathcal{U}(x_u, Cred_U, PP) \leftrightarrow \mathcal{I}(x_i, Crd_I, PP))$

Suppose that J_U is $\mathcal{U}$'s service set consisting of the identities ID_V of ticket verifiers and the central verifier ID_{CV}.

User: $\mathcal{U}$		Ticket Issuer: $\mathcal{I}$
Compute $A_U = g_1 g_2^{d_u} Y_U$. Select $y_1, y_2, y_3 \xleftarrow{R} \mathbb{Z}_p$ and compute $y_4 = \frac{1}{y_1}$, $\bar{\sigma}_U = \sigma_U^{y_1}$, $y = d_u - y_2 y_4$, $\bar{A}_U = A_U^{y_1} g_2^{-y_2}$, $\tilde{\sigma}_U = \bar{\sigma}_U^{-e_u} A_U^{y_1} (= \bar{\sigma}_U^{\alpha})$, $(k_v = H_1(y_3 \| ID_V)$, $P_V = Y_U Y_{CV}^{k_v}$, $Q_V = \tilde{g}^{k_v})_{ID_V \in J_U}$. Compute the proof $\prod_U^1$: $\text{PoK}\{(x_u, d_u, e_u, \sigma_U, y, y_1, y_2, y_4, (k_v)_{ID_V \in J_U}) : \frac{\tilde{\sigma}_U}{\bar{A}_U} = \bar{\sigma}_U^{-e_u} g_2^{y_2} \wedge g_1^{-1} = \bar{A}_U^{-y_4} g_2^{y} \tilde{g}^{x_u} \wedge (P_V = \tilde{g}^{x_u} Y_{CV}^{k_v} \wedge Q_V = \tilde{g}^{k_v})_{ID_V \in J_U}\}$	$\xrightarrow[((P_V, Q_V)_{ID_V \in J_U})]{\bar{\sigma}_U, \tilde{\sigma}_U, \bar{A}_U, J_U, \prod_U^1}$	Verify $\prod_U^1$ and $e(\bar{\sigma}_U, Y_A) \stackrel{?}{=} e(\tilde{\sigma}_U, \mathfrak{g})$. Select $r_u \xleftarrow{R} \mathbb{Z}_p$, and compute $R_U = \bar{g}^{r_u}$. For $ID_V \in J_U$, select $t_v, w_v, z_v \xleftarrow{R} \mathbb{Z}_p$, and compute $D_V = H_3(R_U \| ID_V)$, $E_V^1 = e(\tilde{Y}_A, H_2(ID_V))^{t_v}$, $E_V^2 = \tilde{g}^{t_v}$, $E_V^3 = (\vartheta_1 \vartheta_2^{H_1(TP \| Text_1{}^{a})})^{t_v}$, $K_V = \tilde{g}^{H_1(ID_V)} Y_{CV}^{t_v}$, $s_v = H_1(P_V \| Q_V \| E_V^1 \| E_V^2 \| E_V^3 \| K_V \| Text_2{}^{b})$ and $Z_V = (g_1 g_2^{w_v} g_3^{s_v})^{\frac{1}{x_i + z_v}}$. The authentication tag is $Tag_V = ((P_V, Q_V), (E_V^1, E_V^2, E_V^3, K_V, Text_1, Text_2), (s_v, w_v, z_v, Z_V))$, where s_v is the serial numbers of Tag_V. Select $w, z \xleftarrow{R} \mathbb{Z}_p$ and compute $s = H_1(s_1 \| s_2 \| \cdots \| s_{\lvert J_U \rvert})$ and $Z = (g_1 g_2^{w} g_3^{s})^{\frac{1}{x_i + z}}$ where s is the serial number of the ticket.
For $ID_V \in J_U$, verify $D_V \stackrel{?}{=} H_3(R_U \| ID_V)$, $s_v \stackrel{?}{=} H_1(P_V \| Q_V \| E_V^1 \| E_V^2 \| E_V^3 \| K_V \| Text_2)$. $s \stackrel{?}{=} H_1(s_1 \| s_2 \| \cdots \| s_{\lvert J_U \rvert})$, $e(Z_V, \tilde{Y}_I \mathfrak{g}^{z_v}) \stackrel{?}{=} e(g_1 g_2^{w_v} g_3^{s_v}, \mathfrak{g})$. and $e(Z, \tilde{Y}_I \mathfrak{g}^{z}) = e(g_1 g_2^{w} g_3^{s}, \mathfrak{g})$ Keep (x_3, R_U) secret.	$\xleftarrow{R_U, T_U}$	The ticket is: $T_U = \{(D_V, Tag_V) \mid ID_V \in J_U\} \cup \{(s, w, z, Z)\}$.

[a] $Text_1$ specifies the travel time and other information required by the proxy verification.
[b] $Text_2$ consists of the system version information and all other information which can be used by verifiers to validate the ticket, e.g. valid period, ticket type, *etc*.

Fig. 4: Ticket Issuing Algorithm

Ticket-Validation $(\mathcal{U}(x_u, Tag_V, PP) \leftrightarrow \mathcal{V}(ID_V, PP))$

User: $\mathcal{U}$		Ticket verifier: $\mathcal{V}$ $(ID_V \in J_U)$
Compute $D_V = H_3(R_U \| ID_V)$ and search (D_V, Tag_V).	$\xleftarrow{ID_V}$	Initialize a table T_V.
Compute $k_v = H_1(y_3 \| ID_V)$ and the proof: $\prod_U^2$: $\text{PoK}\{(x_u, z_v) : P_V = \tilde{g}^{x_u} Y_{CV}^{k_v} \wedge Q_V = \tilde{g}^{k_v}\}$.	$\xrightarrow{\prod_U^2, Tag_V}$	If $(s_v, w_v, z_v, Z_V) \in T_V$, abort; otherwise, add (s_v, w_v, z_v, Z_V) in T_V and go to the next step. Check: (1) The correctness of $\prod_U^2$; (2) $s_v \stackrel{?}{=} H_1(P_V \| Q_V \| E_V^1 \| E_V^2 \| E_V^3 \| K_V \| Text_2)$; (3) $e(E_V^2, SK_V) \stackrel{?}{=} E_V^1$; (4) $e(Z_V, Y_I \mathfrak{g}^{z_v}) \stackrel{?}{=} e(g_1 g_2^{w_v} g_3^{s_v}, \mathfrak{g})$. If (1), (2), (3) and (4) hold, the ticket is valid; otherwise, it is invalid.

Fig. 5: Ticket Validation Algorithm

Ticket-Trace(x_{cv}, T_U, PP)

Given a ticket T_U, $\mathcal{CV}$ works as follows:

(1) Let $\Omega_U = \{\}$. For each Tag_V in T_U:
a) Compute: $Y_U = \frac{P_V}{Q_V^{x_{cv}}}$ and $g^{H_1(ID_V)} = \frac{K_V}{(E_V^2)^{x_{cv}}}$; b) Look up $g^{H_1(ID_V)}$ and $\mathcal{V}$'s identity.

Check:

(c1) $s_v \stackrel{?}{=} H_1(P_V||Q_V||E_V^1||E_V^2||E_V^3||K_V||Text_1)$; (c2) $e(Z_V, Y_I \mathfrak{g}^{z_v}) \stackrel{?}{=} e(g_1 g_2^{w_v} g_3^{s_v}, \mathfrak{g})$;
(d) If (c1) and (c2) hold, set $\Omega_U = \Omega_U \cup \{ID_V\}$; otherwise abort.
(e) Verify Y_U remains the same for all tags.

(2) $s \stackrel{?}{=} H_1(s_1||s_2||\cdots||s_{|J_U|})$;

(3) $e(Z, \tilde{Y}_I \mathfrak{g}^z) \stackrel{?}{=} e(g_1 g_2^w g_3^s, \mathfrak{g})$.

If (1), (2) and (3) hold, $\mathcal{CV}$ can determine that the service information of $\mathcal{U}$ with public key Y_U is: $J_U = \Omega_U$; otherwise, the trace has failed.

Fig. 6: Ticket Trace Algorithm

Proxy-Key-Generation $(\mathcal{V}(ID_V) \leftrightarrow \mathcal{CA}(\alpha, ID_V, ID_{V'}))$

Verifier: $\mathcal{V}'$		Central Verifier: $\mathcal{CA}$
		If there is a disruption on the verifier $\mathcal{V}$ and users should go through verifier $\mathcal{V}'$. $\mathcal{CA}$ works as follows:
		(1) check the registration information and find $(ID_V, (d_v, e_v, \sigma_V), SK_V)$ and $(ID_{V'}, (d_{v'}, e_{v'}, \sigma_{V'}), SK_{V'})$;
		(2) choose $\beta_v \xleftarrow{R} \mathbb{Z}_p$ and compute $RK_1 = \tilde{g}^{\beta_v}, RK_2 = (\vartheta_1 \vartheta_2^{H_1(TP\|\|Text_1)})^{\beta_v} \cdot \frac{SK_V}{SK_{V'}}$
Keep $RK_{\mathcal{A}\to\mathcal{A}'}$.	$\xleftarrow{RK_{\mathcal{A}\to\mathcal{A}'}}$	which is bound with the time period TP and system requirements. The re-key is $RK_{\mathcal{A}\to\mathcal{A}'} = (RK_1, RK_2)$.
		$\mathcal{CA}$ publishes the proxy information $(ID_V, ID_{V'})$.

Fig. 7: Proxy Key Generation Algorithm

Proxy-Ticket-Validation $(\mathcal{U}(x_u, Tag_V, PP) \leftrightarrow \mathcal{V}'(ID_{V'}, SK_{V'}, RK_{\mathcal{A}\to\mathcal{A}'}, PP))$

User: $\mathcal{U}$		Ticket verifier: $\mathcal{V}'$ $(ID_{V'} \notin J_U)$
Compute $D_V = H_2(C_U\|\|ID_V)$ and search (D_V, Tag_V). Compute $k_v = H_1(z_u\|\|ID_V)$ and the proof: $\prod_U^2$:	$\xleftarrow{ID_{V'}}$	Initialize a table $T_{V'}$.
PoK$\{(x_u, z_v) : P_V = \tilde{g}^{x_u} Y_{CV}^{k_v} \wedge Q_V = \tilde{g}^{z_v}\}$.	$\xrightarrow{\prod_U^2, Tag_V}$	If $(s_v, w_v, z_v, Z_V) \in T_{V'}$, abort; otherwise, add (s_v, w_v, z_v, Z_V) in $T_{V'}$ and go to the next step. Compute: $\Theta_1 = RK_2 \cdot SK_{V'} = (\vartheta_1 \vartheta_2^{H_1(TP\|\|Text_1)})^{\beta_v} \cdot H_2(ID_V)^\alpha$ and $\Theta_2 = \frac{e(E_V^2, \Theta_1)}{e(RK_1, E_{V,}^3)}$. Check: (1) The correctness of $\prod_U^2$; (2) $s_v \stackrel{?}{=} H_1(P_V\|\|Q_V\|\|E_V^1\|\|E_V^2\|\|E_V^3\|\|K_V\|\| Text_2)$; (3) $\Theta_2 \stackrel{?}{=} E_V^1$; (4) $e(Z_V, Y_I \mathfrak{g}^{z_v}) \stackrel{?}{=} e(g_1 g_2^{w_v} g_3^{s_v}, \mathfrak{g})$. If (1), (2), (3) and (4) hold, the ticket is valid; otherwise, it is invalid.

Note: To prevent double spend a tag/ticket: (1) a central server/database is required to store the verification records and can be accessed by any verifier in the systems; or (2) $\mathcal{V}$ should send the verification records on the disruption day to $\mathcal{V}'$ and $\mathcal{V}'$ should send back the proxy verification records on the disruption day to $\mathcal{V}$.

Fig. 8: Proxy Ticket Validation Algorithm

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

The formal proof of *Theorem 4* is presented in in the full version (see Section VI in [30]).

Theorem 5: Our scheme described in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7 and Fig. 8 is $(\varrho, \epsilon(\ell))$-traceable if the JoC-$q$-SDH assumption holds on the bilinear group $(e, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\tau)$ with the advantage at most $\epsilon_1(\ell)$, the DL assumption holds on the group $\mathbb{G}_1$ with the advantage at most $\epsilon_2(\ell)$, and H_1, H_2 and H_3 are secure cryptographic hash functions, where $\epsilon(\ell) = max\left\{\frac{\epsilon_1(\ell)}{2}(\frac{p-q}{p} + \frac{1}{p} + \frac{p-1}{p^3}), \frac{\epsilon_2(\ell)}{2}\right\}$, ϱ is the total number of ticket issuing queries made by $\mathcal{A}$ and $\varrho < q$.

Proof (Sketch): The strategy used to prove the traceability of our scheme is derived from the group signature scheme [40] and is as follows. Each pseudonym is an EIGamal encryption of a user's public key under the CA's public key and (E_V^2, K_V) included in an authentication tag is the encryption of the verifier $\mathcal{V}$'s identity under the CA's public key. Furthermore, a ticket in our scheme is a BBS+ signature on the on $(ID_V \in J_U)$, pseudonyms $((P_V, Q_V)_{ID_V \in J_U})$, authentication tags $((E_V^1, E_V^2, E_V^3, K_V)_{ID_V \in J_U})$ and auxiliary information $(Text_1, Text_2)$. If $\mathcal{A}$ can generate a ticket which cannot be traced to the real user, the following two types of forgers are considered. Type-I forger outputs a ticket which includes a new pseudonym which has not been used to buy a ticket. Type-II forger outputs a ticket which includes a pseudonym which has been used to buy a ticket but can be traced to a different user. If the Type-I forger succeeds, $\mathcal{B}$ can use $\mathcal{A}$ to break the JoC-q-SDH assumption due to the unforgeability of the BBS+ signature scheme as $\mathcal{A}$ forges a BBS+ signature. For the Type-II forger, when buying a ticket, $\mathcal{A}$ needs to generates a proof of the knowledge included in the pseudonym. If the Type-II forger succeeds, $\mathcal{B}$ can use it to break the DL assumption by using the rewinding technique.

■

The proof of *Theorem 5* is presented in the full version (see Section VI in [30]).

Notably, other entities including the $\mathcal{CA}$, verifiers $\mathcal{V}$ and users $\mathcal{U}$ cannot trace a user's services even if they collude. Because, for each selected service, a user associates it with a pseudonym which is an EIGamal encryption of her public key under the $\mathcal{CV}$'s public key. Moreover, each authentication tag consists of a Boneh-Franklin IBE encryption [15] for the verifier and an EIGamal encryption of the verifier's identity under the $\mathcal{CV}$'s public key. Therefore, no entity can link a user's services by using her pseudonyms and authentication tags, except the $\mathcal{CV}$. This property is very important to protect users' privacy and trace users if required.

VII. Benchmarking

In this section we evaluate the performance of our scheme. The source code of the scheme's implementation is available at [41] 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 28. The implementation makes use of bilinear maps defined over elliptic curves as well as other cryptographic primitives. For the bilinear maps, we used the JPBC library [42] wrapper for the C-based implementation of the PBC libraries [43] while bouncycastle [44] provides the other cryptographic primitives required by our scheme. Note that the implementation by Han *et al.* [8] was based on a Java implementation.

Recall from Section IV that our scheme requires a Type III bilinear map, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_\tau$. The PBC library [43] provides such an instances in the form of the "Type F" pairing which is based on the elliptic curve $E : y^2 = x^3 + b$. The order of groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_\tau$ is determined by the group of points on the elliptic curve, $\#E(F_q) = p$. Note that the Type F curve is a pairing-friendly Barreto-Naehrig (BN) elliptic curve [45]. In our implementation, we instantiate the Type F curve using $rBits = 256$ and $rBits = 638$ where $rBits$ indicates the number of bits needed to represent the prime p.

These bit sizes were chosen to follow the default values specified in the ECC-DAA standard [46] for these curves. Notably, there have been recent attacks [47], [48] against BN curves which reduced of the security of an implementation based on the 256-bit curve. However, the 638-bit curve is still considered to be secure.

For the hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p$, $H_2 : \{0,1\}^* \to \mathbb{G}_2$ and $H_3 : \{0,1\}^* \to \{0,1\}^{\ell'}$ $(\ell' < \ell)$ required by our scheme (see Fig 2), we used $SHA-256$ for both H_1 and H_3 while for H_2 (the random oracle hash function), we used $SHA-256$ and the "newRandomElementfromHash()" method in the JPBC library to construct an element of $\mathbb{G}_2$.

A. Timings

Table III shows the results of the computational time spent in those phases of our scheme that 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 50 iterations.

The set-up phase is a one off process run by the $\mathcal{CA}$ and only takes 233ms for $rBits = 256$ or 971ms for $rBits = 638$.

During the registration phase of the protocol, the generation of credentials by the $\mathcal{CA}$ for the central verifier, $\mathcal{CV}$, takes the most computational effort (8ms and 42ms) as this involves the creation of two credentials. The first credential is equivalent to creating a user credential while the second one is the equivalent of creating a credential for a designated verifier. Similarly, the verification of the $\mathcal{CV}$'s credentials requires the most computational effort (150ms or 782ms for 256 bits and 638 bits respectively). Note that because of this, it is unsurprising that the timings for a user and a verifier in this phase add up to almost the exact number for the $\mathcal{CV}$.

The ticket issuing phase of our implementation is also reasonably fast when $rBits = 256$. For example, when requesting 4 services, the whole process takes ≈ 646ms, 44ms to generate the request, 291ms to produce the ticket and 311ms to verify that the ticket is valid. Even when increasing the field size to 638 bits, the whole issuing process takes ≈ 3243ms of which 1432ms is spent on the actual ticket generation by the issuer. Note that a user can pre-compute her ticket request thus shortening the interaction with the issuer by 44ms or 195ms for 256-bits and 638-bits respectively). The issuer, on the other hand, can also pre-compute some values as part of the ticket issuing process (e.g. D_V, E_V^1 , E_V^2, K_V and parts of Z_V, cf.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

TABLE III
BENCHMARK RESULTS (IN MS)

Protocol phase	Entity	$rBits = 256$	$rBits = 638$
Set-up - Central Authority ($\mathcal{CA}$)			
initialise the system	CA	233	971
Registration - Issuer ($\mathcal{I}$)			
generate credentials	CA	3	14
verify credentials	Issuer	53	280
Registration - User ($\mathcal{U}$)			
generate credentials	CA	3	14
verify credentials	User	52	266
Registration - Verifiers ($\mathcal{V}$)			
generate credentials	CA	6	28
verify credentials	Verifier	101	520
Registration - Central Verifier ($\mathcal{CV}$)			
generate credentials	CA	8	41
verify credentials	Central Verifier	150	782
Ticket Issuing (4 services + CV = 5 tags)			
generate ticket request	User	44	195
generate ticket	Issuer	291	1432
verify ticket	User	311	1616
Ticket Validation - Verifier ($\mathcal{V}$)			
send tag & proof	User	4	20
verify proof & tag	Verifier	81	421
Proxy Verification - Proxy Verifier ($\mathcal{V}'$)			
generate re-key	CA	5	22
verify proof & tag	Proxy Validation	105	545
Ticket Trace (5 tags) - Central Verifier ($\mathcal{CV}$)			
Send ticket & proof	User	4	20
verify proof & trace ticket	Central Verifier	402	2083

Fig. 4). This can reduce the ticket issuing phase by another 193ms or 965ms for 256-bits and 638-bits respectively.

In the validation phase, verifying an individual tag by a designated verifier only takes $\approx$ 85ms or $\approx$ 441ms for $rBits = 256$ and $rBits = 638$ respectively while acting as a proxy verifier takes slightly longer (105ms or 545ms) due to the required re-keying of the provided tag. Note, however, that the generation of the re-key by the $\mathcal{CA}$ is fast (5ms or 22ms).

Evaluating the performance of a scheme is important to demonstrate its viability. For example, in the UK, Transport for London (TfL) [49] has a requirement for the verification of contactless payment cards used for travelling to be below 500ms in order to avoid congestion at ticket barriers. Given the above performance figures and ignoring any latency introduced by the communication channel, our ASSO with proxy re-verification scheme is well below this requirement for $rBits = 256$. For $rBits = 638$, only the proxy re-verification is slightly slower (545ms) than the required 500ms. However, the computation costs and communication cost at gates in a station may not be so suitable for portable devices such as mobile phone or smart card, and so this work can be seen as an initial step. Given that our implementation has not been optimised for any specific elliptic curve, additional improvements in speed should be possible, and further research is needed in this direction. However, with the developing communications network including LTE and 5G, better transport connectivity on trains and at stations and the power of cloud computing and handheld devices, these computation and communication costs might not be a such a barrier in the future.

VIII. CONCLUSIONS

In this paper, a new ASSO with proxy re-verification scheme is proposed which protects users' privacy and allows a user to authenticate herself to a designated verifier anonymously. A central authority can authorise new verifiers to authenticate the user in cases where proxy verification is needed. The re-key enables the proxy verifier to verify tickets on behalf of the original ticket verifier on the specified day. However, the proxy verifier cannot use the re-key to verify tickets with different travel days on behalf of the original verifier. Furthermore, our scheme is formally treated in terms of definition, security model and security proof and its performance has been empirically evaluated.

This work represents one more step in the direction of defining a scheme that provides strong security and privacy properties in the context of smart ticketing. We constructed our scheme using the most efficient pairing (Type-III pairing) available, but the computation cost and communication cost may be not suitable for portable devices, e.g. mobile phone, smart card, tablet, *etc.* Further research is needed to optimise the scheme's construction to minimise the use of pairings in order to potentially improve the efficiency of the scheme and the associated performance of the implementation in order to align to the requirements for verification of contactless payment cards [49]. An alternative approach to improve performance is to construct an ASSO with proxy re-verification scheme which does not rely on bilinear groups and this is an interesting area of future work.

The rail industry is particularly focused on addressing problems associated with disruption and the issues surrounding the sharing of passenger details. Addressing these two concerns is challenging because of the separation of information held by third party retailers and rail service providers. Retailers know about passenger travel information but do not necessarily know about events likely to affect the journey whereas rail service providers know about service disruptions but do not necessarily have the details to contact the passengers who might be affected and hence cannot warn them. Our future research directions will explore using the techniques presented in this paper to facilitate privacy-preserving sharing of passenger details between different parties in the event of disruptions.

Acknowledgments

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

References

[1] D. Recordon and D. Reed, "OpenID 2.0: a platform for user-centric identity management," in *DIM 2006*. ACM, 2006, pp. 11–16.

[2] B. Campbell, C. Mortimore, and M. B. Jones. (2015) Saml 2.0 profile for oauth 2.0 client authentication and authorization grants. [Online]. Available: https://www.oasis-open.org/committees/download.php/14361/sstc-saml-tech-overview-2.0-draft-08.pdf

[3] MIT Kerberos, "Kerberos: The network authentication protocol," 2017. [Online]. Available: https://web.mit.edu/kerberos/

[4] K. Elmufti, D. Weerasinghe, M. Rajarajan, and V. Rakocevic, "Anonymous authentication for mobile single sign-on to protect user privacy," *International Journal of Mobile Communications*, vol. 6, no. 6, pp. 760–769, 2008.

[5] J. Han, Y. Mu, W. Susilo, and J. Yan, "A generic construction of dynamic single sign-on with strong security," in *SecureComm 2010*, ser. LNICST, vol. 50. Springer, 2010, pp. 181–198.

[6] J. Wang, G. Wang, and W. Susilo, "Anonymous single sign-on schemes transformed from group signatures," in *INCoS 2013*. IEEE, 2013, pp. 560–567.

[7] T.-F. Lee, "Provably secure anonymous single-sign-on authentication mechanisms using extended chebyshev chaotic maps for distributed computer networks," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1499 – 1505, 2015.

[8] J. Han, L. Chen, S. Schneider, H. Treharne, and S. Wesemeyer, "Anonymous single-sign-on for n designated services with traceability," in *ESORICS 2018*, ser. LNCS, vol. 11098. Springer, 2018, pp. 470–490.

[9] Technical Strategy Leadership Group (TSLG), "The future railway - the industry's rail technical strategy 2012," 2012. [Online]. Available: https://www.rssb.co.uk/Library/FutureRailway/innovation-in-rail-rail-technical-strategy-2012.pdf

[10] European Commission and European Council, "Regulation (EU) 2016/679: General Data Protection Regulation," 2016. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679&from=EN

[11] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions," in *EUROCRYPT 2003*, ser. LNCS, vol. 2656. Springer, 2003, pp. 614–629.

[12] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E80-A, no. 1, pp. 54–63, 1997.

[13] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *EUROCRYPT 1998*, ser. LNCS, vol. 1403. Springer, 1998, pp. 127–144.

[14] A. Shamir, "Identity-based cryptosystems and signature schemes," in *CRYPTO 1984*, ser. LNCS, vol. 196. Springer, 1984, pp. 47–53.

[15] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *CRYPTO 2001*, ser. LNCS, vol. 2139. Springer, 2001, pp. 213–229.

[16] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *ACNS 2007*, ser. LNCS, vol. 4521. Springer, 2007, pp. 288–306.

[17] J. Han, W. Susilo, and Y. Mu, "Identity-based secure distributed data storage schemes," *IEEE Transactions on Computers*, vol. 63, no. 4, pp. 941–953, 2014.

[18] T. Matsuo, "Proxy re-encryption systems for identity-based encryption," in *Pairing 2007*, ser. LNCS, vol. 4575. Springer, 2007, pp. 247–267.

[19] L. Wang, L. Wang, M. Mambo, and E. Okamoto, "Identity-based proxy cryptosystems with revocability and hierarchical confidentialities," in *ICICS 2010*, ser. LNCS, vol. 6476. Springer, 2010, pp. 383–400.

[20] C.-K. Chu and W.-G. Tzeng, "Identity-based proxy re-encryption without random oracles," in *ISC 2007*, ser. LNCS, vol. 4779. Springer, 2007, pp. 189–202.

[21] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *EUROCRYPT 1996*, ser. LNCS, vol. 1070. Springer, 1996, pp. 143–154.

[22] C.-I. Fan, C.-N. Wu, W.-K. Chen, and W.-Z. Sun, "Attribute-based strong designated-verifier signature scheme," *Journal of Systems and Software*, vol. 85, no. 4, pp. 944–959, 2012.

[23] V. Kuchta, R. A. Sahu, V. Saraswat, G. Sharma, N. Sharma, and O. Markowitch, "Anonymous yet traceable strong designated verifier signature," in *ISC 2018*, ser. LNCS, vol. 11060. Springer, 2018, pp. 403–421.

[24] I. Teranishi, J. Furukawa, and K. Sako, "k-times anonymous authentication (extended abstract)," in *ASIACRYPT 2004*, ser. LNCS, vol. 3329. Springer, 2004, pp. 308–322.

[25] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich, "How to win the clonewars: Efficient periodic n-times anonymous authentication," in *CCS 2006*. ACM, 2006, pp. 201–210.

[26] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in *SCN 2002*, ser. LNCS, vol. 2576. Springer, 2002, pp. 268–289.

[27] J. Camenisch, S. Mödersheim, and D. Sommer, "A formal model of identity mixer," in *FMICS 2010*, ser. LNCS, vol. 6371. Springer, 2010, pp. 198–214.

[28] IBM Research Zürich, "Identity mixer," 2018. [Online]. Available: https://www.zurich.ibm.com/identity_mixer/

[29] L. Nguyen and R. Safavi-Naini, "Dynamic k-times anonymous authentication," in *ACNS 2005*, ser. LNCS, vol. 3531. Springer, 2005, pp. 318–333.

[30] J. Han, L. Chen, S. Schneider, H. Treharne, S. Wesemeyer, and N. Wilson, "Anonymous single sign-on with proxy re-verification," 2018. [Online]. Available: https://arxiv.org/abs/1811.07642

[31] S. D. K. G. N. P.Smart, "Pairings for cryptographers," *Discrete Applied Mathematics*, vol. 156, no. 6, pp. 3113–3121, 2008.

[32] S. Chatterjee and A. Menezes, "On cryptographic protocols employing asymmetric pairings - the role of ψ revisited," *Discrete Applied Mathematics*, vol. 159, no. 13, pp. 1311–1322, 2011.

[33] D. M. Gordon, "Discrete logarithms in GF(P) using the number field sieve," *SIAM Journal on Discrete Mathematics*, vol. 6, no. 1, pp. 124–138, 1993.

[34] D. Boneh and X. Boyen, "Short signatures without random oracles and the SDH assumption in bilinear groups," *Journal of Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[35] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups (extended abstract)," in *CRYPTO 1997*, ser. LNCS, vol. 1294. Springer, 1997, pp. 410–424.

[36] J. Camenisch, A. Kiayias, and M. Yung, "On the portability of generalized Schnorr proofs," in *EUROCRYPT 2009*, ser. LNCS, vol. 5479. Springer, 2009, pp. 425–442.

[37] M. H. Au, W. Susilo, and Y. Mu, "Constant-size dynamic k-TAA," in *SCN 2006*, ser. LNCS, vol. 4116. Springer, 2006, pp. 111–125.

[38] J. Camenisch, M. Drijvers, and A. Lehmann, "Anonymous attestation using the strong Diffie-Hellman assumption revisited," in *TRUST 2016*, ser. LNCS, vol. 9824. Springer, 2016, pp. 1–20.

[39] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in *CRYPTO 2005*, ser. LNCS, V. Shoup, Ed., vol. 3621. Springer, 2005, pp. 205–222.

[40] D. Boneh and H. Shacham, "Group signatures with verifier-local revocation," in *CCS 2004*. ACM, 2004, pp. 68–177.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2919926, IEEE Transactions on Information Forensics and Security

[41] DICE Project, "Benchmark E-ticketing Systems (BETS)," 2017. [Online]. Available: https://github.com/swesemeyer/BenchmarkingETicketingSystems

[42] A. De Caro and V. Iovino, "JPBC: Java pairing based cryptography," in *ISCC 2011*. IEEE, 2011, pp. 850–855.

[43] B. Lynn, "The pairing-based cryptography library," 2006. [Online]. Available: https://crypto.stanford.edu/pbc/

[44] Legion of the Bouncy Castle Inc, "Bouncy Castle Crypto APIs," 2013. [Online]. Available: https://www.bouncycastle.org/

[45] P. S. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *SAC 2005*, ser. LNCS, vol. 3897. Springer, 2005, pp. 319–331.

[46] J. Camenisch, M. Drijvers, A. Edgington, A. Lehmann, and R. Urian, "Fido ecdaa algorithm, implementation draft," 2018. [Online]. Available: https://fidoalliance.org/specs/fido-uaf-v1.1-id-20170202/fido-ecdaa-algorithm-v1.1-id-20170202.html

[47] R. Barbulescu and S. Duquesne, "Updating key size estimations for pairings," *Journal of Cryptology*, 2018. [Online]. Available: DOI:10.1007/s00145-018-9280-5

[48] T. Kim and R. Barbulescu, "Extended tower number field sieve: A new complexity for the medium prime case," in *CRYPTO 2016*, ser. LNCS, vol. 9814. Springer, 2016, pp. 543–571.

[49] Mastercard, "Case study: Contactless Payments Travel Well in London," 2017. [Online]. Available: https://www.mastercard.us/content/dam/mccom/en-us/documents/transport-for-london-case-study-april-2017.pdf

Jinguang Han received his Ph.D from University of Wollongong, Australia, in 2013. He currently is a lecturer in the Centre for Secure Information Technologies (CSIT), Institute of Electronics, Communications and Information Technology (ECIT), Queen's University, UK. His main research interests include cryptography, access control, blockchain and privacy-preserving systems. He has served as a program co-chair of ProvSec 2016 and a program committee member of over 60 international conferences. He is a senior member of the IEEE.

Liqun Chen joined the Department of Computer Science at the University of Surrey as Professor in Secure Systems in 2016. Prior to this appointment, she was a Principal Research Scientist at Hewlett Packard Laboratories in Bristol, UK, which she joined in 1997. Before that she worked at Royal Holloway, University of London, the University of Oxford, and Southeast University in P.R. China. Liqun is a visiting professor at Southeast University. Her research interests include cryptography applied to a broad range of areas in secure systems, such as trusted computing, hardware security, 5G, cloud computing, distributed ledger and Internet of Things, and she is also interested in quantum-resistant cryptographic solutions.

Steve Schneider is the director of the Surrey Centre for Cyber Security, which is recognised by NCSC as one of the UK's Academic Centres of Excellence in Cyber Security Research. He is also Associate Dean for Research and Innovation in the Faculty of Engineering and Physical Sciences. He joined the Department of Computer Science in 2004 and was Head of Department 2004-2010. Before that he worked at Royal Holloway, University of London and at the University of Oxford. His research interests include blockchain and distributed ledger technologies, electronic voting and formal verification.

Helen Treharne is a Reader in Secure Systems in the Department of Computer Science at the University of Surrey. Her main research interests are in data privacy and formal verification. She is leading an interdisciplinary research team on the impact of data privacy on customer experience in the transport domain. Her interests also include the use of trusted computing to improve privacy within secure systems. She is also an expert in formal verification in intelligent transport systems. She was awarded a Royal Academy of Engineering/Leverhulme Trust Senior Research Fellowships during 2013-14 in the area of formal verification of railway control systems.

Stephan Wesemeyer received his PhD from Exeter in 1997. He worked as research fellow for the Centre for Communication Systems Research (now the Institute for Communication Systems) at the University of Surrey for 18 months before joining an IT consultancy in 1999. After almost 10 years working as an IT consultant, he rejoined the university as a research associate in the Department of Computer Science in 2009. He works closely with rail industry partners to provide technology transfer of rail security research and his main research interests include formal verification and trusted and secure implementations.

Nick Wilson joined the Rail Delivery Group's (RDG) Strategy Team to head up their Information Team in March 2018. He previously worked at RDG as a Technical Architect helping to design the major projects to deliver the Customer Information Strategy. Before this Nick worked in a consultancy capacity on railway franchise bids for the UK rail industry and Department for Transport. Nick began working in the Rail Sector at privatisation, as Head of Group IT for one of the owning groups.