Notarized Federated Identity Management for Web Services*

Michael T. Goodrich[1], Roberto Tamassia[2], and Danfeng Yao[2]

[1] Department of Computer Science, University of California
Irvine, CA 92697 USA
goodrich@acm.org
[2] Department of Computer Science, Brown University
Providence, RI 02912 USA
{rt, dyao}@cs.brown.edu

Abstract. We propose a *notarized* federated identity management model that supports efficient user authentication when providers are unknown to each other. Our model introduces a notary service, owned by a trusted third-party, to dynamically notarize assertions generated by identity providers. An additional feature of our model is the avoidance of direct communications between identity providers and service providers, which provides improved privacy protection for users. We present an efficient implementation of our notarized federated identity management model based on the Secure Transaction Management System (STMS). We also give a practical solution for mitigating aspects of the identity theft problem and discuss its use in our notarized federated identity management model. The unique feature of our cryptographic solution is that it enables one to proactively prevent the leaking of secret identity information.

1 Introduction

Digital identity management is becoming an integral part of our lives, as consumers and businesses rely more and more on online transactions for daily tasks, such as banking, shopping, and bill payment. These transactions crucially depend on networked computer systems to communicate sensitive identity data across personal, company, and enterprise boundaries.

Unfortunately, the overuse of personal information in online transactions opens the door to identity theft, which poses a serious threat to personal finances and credit ratings of users and creates liabilities for corporations. Moreover, the increasing dangers of identity theft are negatively affecting people's collective confidence on the digital world for online financial transactions [10]. Thus, effective solutions for managing digital identity on both the individual and enterprise levels are urgently needed.

* This work was supported in part by the National Science Foundation under grants IIS–0324846, CCF–0311510 and CNS–0303577, and by IAM Technology, Inc. The work of the first author was done primarily as a consultant to Brown University.

Additionally, end users are challenged with increasing numbers of websites that require access control and authentication. Studies show that users resort to using weak passwords or writing them down to alleviate the burden of memorizing multiple passwords. One well-known identity management solution that deals with this issue is the *single sign-on* (*SSO*) technique, which requires the user to authenticate only once to a website, and then automatically authenticates the user to other websites from then on, within a session.

The approach based on cryptographic-enabled assertions is embodied by the *Security Assertion Markup Language* (*SAML*) [7]. SAML 2.0 is generally believed to support general cross-domain authentication and SAML is quickly becoming the de-facto means for exchanging user credentials between trusted environments. The identity federation architecture of *Liberty Alliance* is compliant with the SAML 2.0 standard [14]. Indeed, SAML is specifically designed to support cross domain single sign-on, which is illustrated in the following example.

A user has a secure logon session to a website, (e.g., Airline.com), and is accessing resources on that site. Airline.com serves as the *identity provider* site. At some point in the session, the user is directed to another web site in a different DNS domain for a related service, and this outside domain is called the *service provider* site (e.g., CarRental.com). The identity provider (Airline.com) asserts to the service provider (CarRental.com) that the user is known to the identity provider and gives to the service provide the user's name and session attributes (e.g., Gold member). Since the service provider trusts the assertions generated by the identity provider, it creates a session for the user based on the information received. The user is not required to authenticate again when directed to the service provider site. Hence, single sign-on is achieved.

The *identity provider (IdP)* in SAML [7] is defined as the system, or administrative domain, that asserts information about a subject. An identity provider asserts that a user has been authenticated and has certain attributes. The *service provider (SP)* is defined as the system, or administrative domain, that relies on the information supplied to it by the identity provider.

1.1 Motivation for Notarized ID Federation

In existing federated identity management systems that support SAML, such as the *Liberty Identity Federation Framework* (*ID-FF*) [8] and *WS-Federation* [24], it is up to the service provider to decide whether it trusts the assertions provided to it. Service providers in SAML are also known as *relying parties* due to the fact that they rely on the information provided by an identity provider. This reliance implies that websites of different administrative domains need to trust each other's access control verdicts on end users. In fact, SAML single sign-on relies on the concept of *identity federation* in order for it to work at all. An identity federation is said to exist between an identity provider and a service provider, when the service provider accepts assertions regarding a user from the identity provider [7].

In fact, most existing SSO solutions assume preexisting trust relationship among providers and do not provide concrete mechanisms for the trust establish-

ment between providers. The WS-Federation specification [24] discusses several trust relationships between identity providers and service providers, including directed trust, indirected brokered trust, and chained trust. However, details on how the trust relationships and identity brokers can be instantiated are not given. This limitation hinders the wide deployment of SSO in web-service environments, because providers may be unknown to each other. Therefore, flexible, reliable, and secure trust establishment mechanisms need to be provided for federated identity management.

1.2 Our Contributions

1. We propose a notarized federated identity management model that supports automatic user authentications when the providers are unknown to each other. Our model introduces a *notary server*, which is owned by a trusted third-party to dynamically notarize assertions generated by identity providers. As an extra feature provided by the notary server, our federated identity management model reduces possible collusions between identity providers and service providers, and gives improved privacy protections for users.
2. We describe an efficient implementation of the federated identity management protocol with the existing *Secure Transaction Management System* (*STMS*) [1, 11]. The notary server caches the assertions at a collection of responders deployed in the network. Even when the responders are located in insecure, untrusted locations, a service provider can easily identify a forged or tampered assertion so that the integrity of an assertions is maintained.
   Our protocol is a concrete solution for a trust broker model proposed by existing federated identity management systems [24]. Besides brokering trust, our solution offers additional features. *Accountability* is supported by archiving signatures on requests and assertions. *User privacy* is achieved by encrypting assertions stored by the notary server. *Verification efficiency* is achieved by using the authenticated-dictionary technique (see, e.g., [1, 11, 18, 21]) implemented in STMS.
3. We also give a practical solution for mitigating aspects of the identity theft problem, and discuss how it is used in our federated identity management protocol. Our cryptographic solution is based on the *Identity-Based Encryption* (*IBE*) scheme [4]. The main feature of our cryptographic solution is that it enables one to proactively prevent the leaking of secret identity information.

Organization of the paper. Our model for notarized federated identity management is described in Section 2. The STMS implementation of the notarized federated identity management protocol is presented in Section 3. In Section 4, we give an IBE-based authentication protocol. The security of the federated identity management protocol and the IBE-based authentication protocol is analyzed in Section 5. Related work is given in Section 6.

2 Notarized Federated Identity Management

Our notarized federated identity management model introduces a *notary server*, a trusted third-party that dynamically maintains assertions generated by identity providers. Assertions are generated by identity providers and stored by the notary server. When a service provider needs to verify an assertion, it queries the notary server for a *notarized assertion* that shows the trustworthiness of the identity provider generating the assertion.

2.1 Notary Server and Notarized Assertion

In a notarized ID federation, a notary server is trusted by both identity providers and service providers. Identity providers that have good internet behavior and reputation are allowed to register with the notary server, and thus are trusted. The notary server stores the assertions generated by registered identity providers. A notary server supports two operations, SUBMIT and QUERY.

- SUBMIT(id, S_{id}, sig): a registered identity provider *IdP* authenticates itself to the notary server, and submits via a secure channel the tuple (id, assertion, signature), denoted by (id, S_{id}, sig), to the notary server. The assertion S_{id} states the attributes of an identity *id*, and the signature *sig* is signed by *IdP* on the assertion S_{id}. The notary server stores the tuple.
- QUERY(*id*): a service provider *SP* queries in a *public (insecure)* channel the notary server for assertions associated with identity *id*, and the notary server returns the *notarized assertion(s)*.

A notarized assertion has a proof showing that the assertion is indeed stored by the notary server, which implies that the identity provider that generates the assertion is trustworthy. The reason for not using a secure channel in QUERY is for higher efficiency and scalability in a distributed environment. The challenge, thus, becomes how to efficiently generate and verify the notarized assertion, even when it is transmitted in a insecure channel. Our solution is based on the authenticated dictionary technique [1, 11, 18, 21], which is more scalable than using a signature scheme.

The notary server provides the assurance of the trustworthiness of assertions when identity providers are unknown to the service providers. The notary server is a bridge of trust between providers in web-service transactions. Another advantage of storing assertions on the notary server is the prevention of direct contact between identity providers and service providers. A notarized assertion does not contain the name of the identity provider. This further increases the difficulty of collusions among providers to discover private user information.

We assume that the notary server is trustworthy, and is trusted by all entities (users, identity providers, service providers). The security properties of our notarized federated identity management protocol are summarized below and are analyzed in Section 5.

– *Security* is defined as that no polynomial-time adversary can forge a notarized assertion that can be accepted by a service provider.
– *Secrecy* is defined intuitively as that the protocol does not leak any information about a notarized assertion to a (polynomial-time) adversary. This property provides privacy protection to the users.
– *Accountability* is defined as that identity providers should be held accountable for the assertions generated, and for any unauthorized information disclosure about the users.

Note that the notary server only certifies that the source of an assertion is trustworthy; it is not required to examine and certify the content of an assertion. In fact, our protocol, which is described next, deliberately avoids disclosing assertion contents to the notary server by encrypting the assertions. This feature is for the purpose of user privacy, and prevents the notary server from gaining knowledge of private user information.

2.2 Protocol

In this section, we present the protocol for our notarized federated identity management model. The following entities participate in the protocol: a user, an identity provider, a service provider, and a notary server. The protocol gives an instantiation of operations SUBMIT and QUERY. Note that the roles of identity provider and service provider are interchangeable. For example, a bank can be the identity provider in one scenario and the service provider in another scenario.

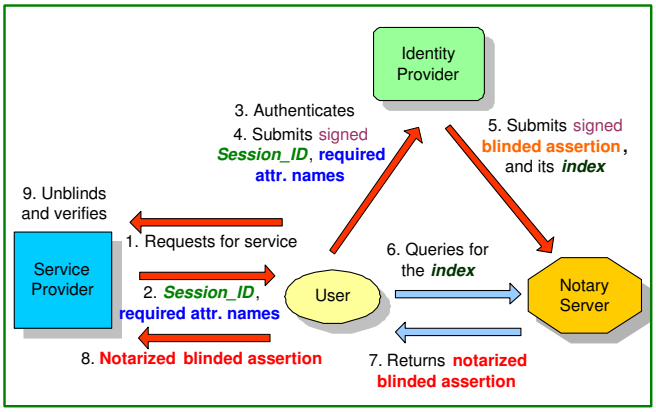

Fig. 1. Overview of the notarized federated identity management protocol.

We assumes that the notary server knows the public keys of registered identity providers. In addition, the public key of the notary server is known by all of the providers. A schematic drawing of the protocol is shown in Figure 1.

In the protocol, the user only needs to authenticate once to an identity provider. Subsequent requests for service from multiple service providers do not require the user for authentication. Nevertheless, for protecting personal privacy, the user is given the ability to examine the contents of assertions to be given to the service providers in our protocol. If the assertions are generated by the identity provider according to the user's request, then they are passed on to the service providers. We argue that having the user involved in the identity management protocol for privacy purpose is a feasible solution. This concept was also proposed by other federated identity management solution [2]. The process can be automated to minimize the user's manual participation.

Public parameters include a collision-resistant one-way hash function, H, that takes a binary string of arbitrary length and hashes to a binary string of fixed length k: $H : \{0,1\}^* \rightarrow \{0,1\}^k$. For the blinding purpose, the public parameters also include two public strings P_1 and P_2. Providers also agree on a symmetric-key encryption scheme for blinding and unblinding assertions. The encryption and decryption of a message x with a secret key K are denoted as $E_K(x)$ and $D_K(x)$, respectively. Our protocol is described as follows.

1. The user requests services from a service provider *SP*. *SP* requires attribute information of the user needed to complete the service.
2. *SP* opens a secure communication channel with the user. The user and *SP* each generate a random integer of the same length. They first exchange the cryptographic hashes of these integers as commitments using the secure channel, and then they exchange the integers using the secure channel. The session_ID N is finally computed as the XOR of the two integers. *SP* also informs the user of the attribute names that are needed for the service (e.g., billing address and age).
3. The user authenticates to her identity provider *IdP*. If the authentication is successful, the user opens a secure channel with *IdP*, and transmits a *signed* request that contains the session_ID and the required attribute names.
4. *IdP* verifies and stores the signed request by the user. The signature is for the accountability purpose in case of dispute (see Section 5).
5. *IdP* then computes the *index* of the assertion as the hash of session_ID concatenated with the public parameter P_1: $h = H(N, P_1)$. It then generates an assertion S_h about the user using index h. For example, S_h states that h is a university student.
6. To prevent information leaking, *IdP* blinds the assertion as follows.
   (a) *IdP* computes the *blinding factor* K as the hash of the session_ID concatenated with the public parameter P_2: $K = H(N, P_2)$.
   (b) *IdP* encrypts S_h with the symmetric encryption scheme, using K as the secret key. This gives the *blinded assertion* $S'_h = E_K(S_h)$.

   The blinded assertion S'_h is signed by *IdP* with its private key, which gives a signature sig_h.
7. *IdP* runs SUBMIT(h, S'_h, sig_h) with the notary server to submit tuple (h, S'_h, sig_h) through a secure channel as follows.
   (a) *IdP* first authenticates to the notary server to establish a secure communication channel.

(b) *IdP* then transmits tuple (h, S'_h, sig_h) to the notary server.
(c) The notary server verifies signature sig_h, and stores (S'_h, sig_h) indexed by h. The signature is stored for accountability purposes.

8. The user computes the index $h = H(N, P_1)$ from N and P_1, and runs QUERY(h) to obtain the assertion for h. The notary server processes the query as follows.
   (a) The blinded assertion S'_h associated with index h is retrieved.
   (b) The notary server *notarizes* the assertion S'_h, and returns the notarized assertion. We describe two approaches for the realization of notarized assertion in the following sections. Note that the QUERY operation between the user and the notary server does not require a secure channel.
9. Once the user obtains the returned notarized blinded assertion, she unblinds it with the blinding factor $K = H(N, P_2)$. This is done by decrypting S'_h with K, which gives $S_h = D_K(S'_h)$. The user verifies that the assertion does not release any unauthorized personal information about her.
10. The notarized blinded assertion is then relayed from the user to the service provider, who verifies that it is notarized by the notary server. This implies that the identity provider *IdP* is trusted by the notary server. If the verification succeeds, the assertion S'_h is unblinded in the same way as in Step 9. The attribute information is obtained from the assertion, and index h is compared with the hash $H(N, P_1)$ of session_ID N and P_1. The user is then granted the service if the verification passes. The service provider also stores the notarized assertion for accountability purposes.

The computation of blinding factor K and index h is equivalent to using two different hash functions on the session_ID N, i.e., $h = H_1(N)$ and $K = H_2(N)$, where H_1 and H_2 are collision-resistant one-way hash functions.

3 Realization of Notarized Assertions

Notarized assertions can be realized using simple time-stamped signatures. The notary server individually signs every assertion and the current time-stamp with its private key. The notarized assertion consists of this signature along with the assertion and time-stamp. To verify a notarized assertion, the service provider verifies the signature against the public key of the notary server, which can be obtained through usual means such as a public key certificate.

In the above approach, notarizing assertions can be a performance bottleneck because the notary server needs to sign every individual assertion. To improve the efficiency of the notary server, we give an improved realization of notarized assertions using the secure transaction management system (STMS).

The main advantage of implementing notary assertions with STMS in comparison to the simple time-stamped signature approach is its high efficiency of computation. The notary server only needs to generate one signature as opposed to a signature for each assertion. In addition, STMS also provides a distributed architecture for fast real-time dissemination of assertion updates. STMS has

been previously used to build an accredited domainkeys framework for secure e-mail systems [12]. Next, we first introduce the components and algorithms of STMS, then we describe how to use STMS to scale up the notary service.

3.1 Secure Transaction Management System (STMS)

The computational abstraction underlying STMS is a data structure called an *authenticated dictionary* (see, e.g., [1, 11, 17, 18, 21, 22]), which is a system for publishing data and supporting authenticated responses to queries about the data. In an authenticated dictionary, the data originates at a secure central site, called *STMS source* and is distributed to servers scattered across the network, called *STMS responders*. The responders answer queries on behalf of the source about the data made by clients. It is desirable to delegate query answering to the responders for two reasons: (1) the source is subject to risks such as denial-of-service attacks if it provides services directly on the network, and (2) the large volume and diverse geographic origination of the queries require distributed servers to provide responses efficiently.

The STMS source sends real-time updates to the responders together with a special signed time-stamped fingerprint of the database called the *basis*. A user's query to the responder asks whether an element is contained in the authenticated dictionary maintained by STMS source. A responder replies to the query with an authenticated response. This consists of the answer to the query, the proof of the answer, the basis and its signature signed by the STMS source. Informally speaking, the proof is a *partial fingerprint* of the database that, combined with the subject of the query, should yield the fingerprint of the entire database. A proof consists of a very small amount of data (less than 300 bytes for most applications) and can be validated quickly against the signed basis by a client. We refer readers to the authenticated dictionary literature [1, 11] for more information.

3.2 Realization of Notarized Assertions with STMS

Using STMS, the notary server consists of a *notary source* and several *notary responders*. The notary source needs to be a trusted server that stores assertion inputs from identity providers. Notary responders can be strategically placed in geographically dispersed locations to accommodate fast queries. They obtain real-time updates from the notary source, and answer queries from users. Notary responders do not need to be trusted servers. The notarized assertions returned by them can be authenticated by verifying against the public key of the notary source by anyone.

With STMS, a notarized assertion returned by QUERY operation consists of two parts: assertion itself and a STMS proof. As described in the previous section, the proof is a sequence of hash values of elements in the notary server for proving the existence of the assertion. The size of the proof is quite compact, even for large number of items in the notary server. Therefore, transmitting the proof can be quite fast. The service provider then obtains the signed STMS basis

of the current time quantum from the notary responder, if it does not yet have it. The proof of the assertion is verified against the basis, and the signature of the basis is verified against the public key of the notary source. If the verification is successful, the request is granted. The signed basis remains the same for the duration of a time quantum, therefore it only needs to be obtained once for each time quantum. The rest of the notarized federated identity management protocol with STMS follows the protocol in Section 2.2.

Due to page limit, the protocol and security of STMS implemented notarized federated identity management are not presented. The security is based on the security of STMS, which has been previously proved [1].

4 Reducing the Risks of Identity Theft

Recently, several practical solutions against on-line identity theft have been proposed [2, 16]. In this section, we first analyze causes of a successful identity theft. Then, we give a practical solution, and describe how to use our scheme in our notarized federated identity management protocol.

4.1 Identity Theft and Its Causes

Identity theft is a type of crime in which an imposter obtains key pieces of personal information, such as Social Security or driver's license numbers, in order to impersonate someone else. Although an identity thief might crack into a database to obtain personal information, it is believed that a thief is more likely to obtain information using Trojans or even old-fashion methods such as dumpster diving and shoulder surfing.

We observe that the current authentication protocols, both physical and digital ones, are fundamentally susceptible to identity theft, even if an individual is careful in protecting her sensitive information. Physical authentication protocols include the procedures for obtaining a driver's license at a government office, opening a bank account, and applying for mortgage. Digital authentication protocols include the corresponding on-line transactions. In current solutions, key pieces of personal information are usually communicated in the clear or stored in the clear. This makes stealing of information easier for identity thieves. Although the SSL protocol encrypts communications between a user and a server, this does not prevent Trojan keyloggers, or shoulder surfing, because the user still needs to disclose and type over and over sensitive information such as her social security number.

We argue that this fundamental characteristic of the existing authentication protocols is one of the main causes of identity theft, namely using sensitive information in clear form for authentication. We propose a simple and practical cryptographic protocol for authentication. Our solution ties personal information to random secrets, which are used to prove *interactively* the ownership of the personal information but are *never disclosed*.

Motivation for using IBE. In public key encryption schemes, the private key information is never disclosed. Yet, a challenge-response process can be used by a user to prove the possession of the private key to an identity provider. The private key is usually protected by encrypting it with a passphrase, and storing it in a portable device, such as a smart card or a USB flash drive. Observe that the private key is never disclosed in clear during transactions, hence it never appears in any printed form or display. Therefore, it is difficult for attackers to retrieve someone's private key using standard identity theft techniques. To steal the private key, an attacker would need to obtain the physical device and know the passphrase. In order to associate identity information with public keys, the only known encryption scheme is the Identity-Based Encryption (IBE) scheme [4, 20, 25]. A public key in IBE will be the personal information (e.g., the social security number of an individual). For authentication, an individual not only needs to know her personal information (e.g., social security number), but also needs to prove the possession of the corresponding private key for authentication.

4.2 A Cryptographic Authentication Protocol

We propose to use ID-based encryption scheme for implementing an authentication protocol for sensitive personal information. Our protocol minimizes the exposure of secret personal information and thus is more robust against identity theft than existing authentication methods. Entities in our protocol include a user, an ID authority, an identity provider, and a revocation server controlled by the ID authority. Our authentication protocol has the following operations: Setup, Register, Authenticate, and Refresh. It requires an on-line revocation server maintained by the ID authority.

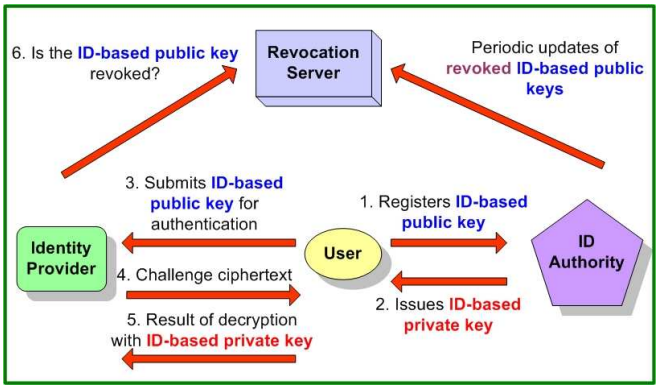

Fig. 2. A diagram of the IBE-based authentication protocol.

Refreshing the secret key of identity information can be tricky, because the identity information typically does not change, e.g. social security number. We

show later how to use multiple pieces of identity information and on-line revocation checking to leverage this. A diagram of the protocol is shown in Figure 2. Here, we describe the realization of the above operations with IBE scheme.

1. SETUP: The ID authority runs the PKG SETUP operation of IBE.
2. REGISTER: A user requests for an *ID-based private key* from an ID authority. The user needs to be physically present in the ID office, for example the passport office, with paper identifications such as passport, birth certificate. The ID authority authenticates the user's identity.
   If the user's identity is verified, the ID authority generates the ID-based private key for the user. The ID authority runs the EXTRACT operation of IBE with the user's *ID-based public key*, which is the user's identity information concatenated with a unique serial number l. For example, l can be the driver's license number. l is used for revocation purpose. Because the identity information such as social security number cannot be easily revoked, we need an additional replaceable field l. Note that l cannot be any random number, because using a random value as public key requires public-key certification, which defies the purpose of identity-based encryption. In what follows, we use the driver's license number as l. The ID-based private key generated by EXTRACT is given to the user. The user's driver's license can be equipped with a smart card chip and store the private key.
3. AUTHENTICATE: The user and the identity provider engage in a challenge-response protocol as follows.
   (a) The user gives his ID-based public key to the identity provider, which is the identity information concatenated with the driver's license number l to the identity provider.
   (b) The identity provider picks a random nonce m. It runs ENCRYPT of IBE to encrypt m using the user's ID-based public key.
   (c) The ciphertext is given to the user, who runs DECRYPT of IBE with his ID-based private key. If the user is unable to correctly decrypt the ciphertext, the authentication fails and returns false.
   (d) The identity provider queries the revocation server maintained by the ID authority for the number l in the public key of the user. If l has been revoked, then the authentication fails. Otherwise, the authentication is successful and returns true.
4. REFRESH: The ID authority refreshes the ID-based private key of the user as follows.
   (a) The user authenticates his current ID-based public key to the ID authority.
   (b) The ID authority puts the the driver's license number l on the revocation server to indicate that l has been revoked.
   (c) The ID authority generates a new driver's license number l' for the user. The new ID-based public key of the user associated with his identity information is that identity information concatenated with l'. For example, the public key is 999-99-9999 $\circ$ 1234567890, where 999-99-9999 is the social security number and 1234567890 is the new driver's license number l'.

(d) The ID authority runs EXTRACT of IBE to compute a new private key, which is transmitted to the user via a secure channel or in person. The user stores the new ID-based private key in his smart card.

The main advantage of our authentication protocol is that the secret personal information is not released during the transaction, which minimizes identity theft attacks such as dumpster diving and shoulder surfing. Our protocol can be used in any user authentication applications. In particular, it can be used in any federated identity management system when a user authenticates his personal information with an identity provider. For example, a user is required to run the AUTHENTICATE algorithm with the identity provider when his social security number is needed. Without the corresponding private key, it is impossible for an identity thief to accomplish this.

5 Security Analysis

In this section, we first analyze the security of the notarized federated identity management protocol, and then analyze the IBE-based authentication protocol.

The security of our notarized federated identity management protocol is analyzed from the perspectives of the user, the identity provider, the service provider, and the notary server, as each of them has different requirements on the security provided by the system. In what follows, we assume the existence of a signature scheme that is secure against existential forgery by polynomial-time adversaries in the security parameter of the signature scheme. Existential forgery means that an adversary forges a signature that the notary server has not signed in the past. An adversary in our protocol can monitor traffics in unsecured channels, request for services, request the identity provider to blind assertions of her choice, and request the notary server to notarize assertions of her choice.

We assume that the notary server is trustworthy, and is trusted by all entities (users, identity providers, service providers). All entities are assumed to follow the federated identity management protocol presented in Section 2. The following theorem states the nonforgeability of a notarized assertion.

Theorem 1. *In the notarized federated identity management protocol, no polynomial-time adversary can successfully forge a valid notarized assertion that is not generated by the notary server.*

For the privacy protection of a user, an important privacy requirement is the secrecy of assertions. This is summarized in the following theorem.

Theorem 2. *Assume the existence of a collision-resistant one-way hash function, and a secure symmetric key encryption scheme. In the notarized federated identity management protocol, a polynomial-time adversary and untrusted notary responders cannot obtain any information from a blinded assertion.*

For decentralized authorization systems such as the federated identity management, an important security requirement is accountability. To prevent possible disputes, identity providers should be held accountable for the assertions that they have generated. In addition, to prevent unauthorized information exchange among providers, users should be able to dispute any fraudulent assertion requests. These properties are achieved in our protocol.

Theorem 3. *In the notarized federated identity management protocol, the identity provider is held accountable for the assertions that it generates.*

Theorem 4. *In the notarized federated identity management protocol, providers are held accountable for any unauthorized information exchange among them.*

Theorem 5. *The notarized federated identity management protocol is secure against replay attacks.*

The security of the IBE-based authentication protocol is defined as the adversary's inability of impersonating a user, and is based on the security of the identity-based encryption scheme as stated in Theorem 6.

Theorem 6. *Given an identity-based encryption scheme that is semantic-secure against an adaptive polynomial-time adversary, the IBE-based authentication protocol is secure.*

The proofs of the above theorems are in the full version of the paper [13].

6 Related Work

Our approach of using privacy protection as a means to avoid identity theft is related to anonymous credential systems [5, 6, 9, 15, 26]. Anonymous credential systems (a.k.a. pseudonym systems) allow anonymous yet authenticated and accountable transactions between users and service providers. Existing anonymous credential systems are different from our single sign-on system, in that they do not consider a federated identity infrastructure behind the providers. In comparison, our system focuses on how to manage user authentication in the more realistic setting of a federation of providers. Our system achieves simple pseudonym solutions and efficient single sign-on by taking advantages of the federated structure. In particular, we do not need a credential system, because the assertions can be short-lived and generated on-line by identity providers.

The federated identity management solution proposed by Bhargav-Spantzel, Squicciarini, and Bertino [2] emphasizes the need for proving the knowledge of personal information without actually revealing it, in order to help prevent identity theft. In their solution, personal data such as a social security number is never transmitted in the clear. Commitment schemes and zero-knowledge proofs are used to commit data and prove the knowledge of the data. Our identity-based solution has a similar goal to this approach, but there is one important difference. We allow personal data such as social security numbers and credit card numbers to be transmitted in the clear. Yet, every time this information is used, the user needs to prove the possession of corresponding private keys. This requires minimal changes to the existing financial and administrative infrastructure, as personal information in our scheme is stored the same way as it is currently. IBE [4] conveniently makes this possible, and, interestingly, this approach is also more efficient than zero-knowledge proof-of-knowledge protocols.

BBAE is the federated identity-management protocol proposed by Pfitzmann and Waidner [19]. They give a concrete browser-based single sign-on protocol

that aims at the security of communications and the privacy of user's attributes. Their protocol is based on a standard browser, and therefore does not require the user to install any program. The main difference with this and our approach is that we provide a notary mechanism for authenticating assertions when *IdP* and *SP* are not previous known to each other.

In the access control area, the closest work to ours is the framework for regulating service access and release of private information in web-services by Bonatti and Samarati [3]. They study the information disclosure using a language and policy approach. We designed cryptographic solutions to control and manage information exchange. Their framework mainly focuses on the client-server model, whereas our architecture include two different types of providers.

A counter measure for identity theft through location cross-checking and information filtering was recently proposed [23]. This paper addresses the identity cloning problem, and proposes to use personal location devices such as GPS and central monitoring systems to ensure the uniqueness of identities. However, the central monitoring system in their solution is likely to be a performance bottleneck. Moreover, because identity thieves are geographically dispersed, distributing the monitoring task into several locations is not feasible. In comparison, our solution is simple and efficient to adopt. Because we tie the secret identification information to a tamper-resistant smart card (e.g., driver's license), card theft can be easily noticed and reported by the card owner.

Acknowledgements. The authors would like to thank David Croston for useful comments.

References

1. A. Anagnostopoulos, M. T. Goodrich, and R. Tamassia. Persistent authenticated dictionaries and their applications. In *Proc. Information Security Conference (ISC 2001)*, volume 2200 of *LNCS*, pages 379–393. Springer-Verlag, 2001.
2. A. Bhargav-Spantzel, A. C. Squicciarini, and E. Bertino. Establishing and protecting digital identity in federation systems. In *Proceedings of the 2005 ACM Workshop on Digital Identity Management*, pages 11–19, November 2005.
3. P. A. Bonatti and P. Samarati. A uniform framework for regulating service access and information release on the web. *Journal of Computer Security*, 10(3):241–272, 2002.
4. D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology — Crypto '01*, volume 2139 of *LNCS*, pages 213–229. Springer-Verlag, 2001.
5. J. Camenisch and A. Lysyanskaya. Efficient non-transferable anonymous multishow credential system with optional anonymity revocation. In B. Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer Verlag, 2001.
6. J. Camenisch and E. Van Herreweghen. Design and implementation of the *idemix* anonymous credential system. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, pages 21–30, 2002.
7. S. Cantor, F. Hirsch, J. Kemp, R. Philpott, E. Maler, J. Hughes, J. Hodges, P. Mishra, and J. Moreh. Security Assertion Markup Language (SAML) V2.0. Version 2.0. OASIS Standards.

8. S. Cantor and J. Kemp. Liberty ID-FF Protocols amd Schema Specification. Version 1.2. Liberty Alliance Project. `http://www.projectliberty.org/specs/`.
9. D. Chaum. Security without identification: transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, October 1985.
10. Cyber Security Industry Alliance. Internet security national survey, No. 2, December 2005. `https://www.csialliance.org/StateofCyberSecurity2006/`.
11. M. T. Goodrich, R. Tamassia, and A. Schwerin. Implementation of an authenticated dictionary with skip lists and commutative hashing. In *Proc. 2001 DARPA Information Survivability Conference and Exposition*, volume 2, pages 68–82, 2001.
12. M. T. Goodrich, R. Tamassia, and D. Yao. Accredited DomainKeys: a service architecture for improved email validation. In *Proceedings of the Conference on Email and Anti-Spam (CEAS '05)*, July 2005.
13. M. T. Goodrich, R. Tamassia, and D. Yao. Notarized federated identity management for web services, April 2006. Brown University Technical Report. `http://www.cs.brown.edu/cgc/stms/`.
14. Liberty Alliance Project. `http://www.projectliberty.org`.
15. A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In H. Heys and C. Adams, editors, *Selected Areas in Cryptography*, volume 1758 of *Lecture Notes in Computer Science*. Springer Verlag, 1999.
16. P. Madsen, Y. Koga, and K. Takahashi. Federated identity management for protecting users from ID theft. In *Proceedings of the 2005 ACM Workshop on Digital Identity Management*, pages 77–83, November 2005.
17. C. Martel, G. Nuckolls, P. Devanbu, M. Gertz, A. Kwong, and S. G. Stubblebine. A general model for authenticated data structures. *Algorithmica*, 39(1):21–41, 2004.
18. M. Naor and K. Nissim. Certificate revocation and certificate update. In *Proceedings of the 7th USENIX Security Symposium*, pages 217–228, 1998.
19. B. Pfitzmann and M. Waidner. Federated identity-management protocols. In *Security Protocols Workshop*, pages 153–174, 2003.
20. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology — Crypto '84*, volume 196 of *LNCS*, pages 47–53. Springer-Verlag, 1984.
21. R. Tamassia. Authenticated data structures. In *Proc. European Symp. on Algorithms*, volume 2832 of *Lecture Notes in Computer Science*, pages 2–5. Springer-Verlag, 2003.
22. R. Tamassia and N. Triandopoulos. Computational bounds on hierarchical data processing with applications to information security. In *Proc. Int. Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of *LNCS*, pages 153–165. Springer-Verlag, 2005.
23. P. van Oorschot and S. Stubblebine. Countering identity theft through digital uniqueness, location cross-checking, and funneling. In *Proceedings of Financial Cryptography and Data Security (FC '05)*, pages 31–43, 2005.
24. Web Services Federation Language (WS-Federation), 2003. `ftp://www6.software.ibm.com/software/developer/library/ws-fed.pdf`.
25. D. Yao, N. Fazio, Y. Dodis, and A. Lysyanskaya. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 354 – 363. ACM Press, 2004.
26. D. Yao and R. Tamassia. Cascaded authorization with anonymous-signer aggregate signatures. In *Proceedings of the Seventh Annual IEEE Systems, Man and Cybernetics Information Assurance Workshop (IAW '06)*, June 2006.