CERIAS Tech Report 2005-48

ESTABLISHING AND PROTECTING DIGITAL IDENTITY IN FEDERATION SYSTEMS

by Abhilasha Bhargav-Spantzel, Anna C. Squicciarini, Elisa Bertino

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

Establishing and Protecting Digital Identity in Federation Systems *

Abhilasha Bhargav-Spantzel
CERIAS, Purdue University
bhargav@cerias.purdue.edu

Anna C. Squicciarini
Universita degli Studi di Milano
squiccia@dico.unimi.it

Elisa Bertino
CERIAS, Purdue University
bertino@cerias.purdue.edu

ABSTRACT

We develop solutions for the security and privacy of user identity information in a federation. By federation we mean a group of organizations or service providers which have built trust among each other and enable sharing of user identity information amongst themselves. We first propose a flexible approach to establish a single sign-on (SSO) ID in the federation. Then we show how a user can leverage this SSO ID to establish certified and un-certified user identity attributes without the dependence on PKI for user authentication. This makes the process more usable and privacy preserving. Our major contribution in this paper is a novel solution for protection against identity theft of these identity attributes. We provide protocols based on cryptographic techniques, namely zero knowledge proofs and distributed hash tables. We show how we can preserve privacy of the user identity without jeopardizing security. We formally prove correctness and provide complexity results for our protocols. The complexity results show that our approach is efficient. In the paper we also show that the protocol is robust enough even in case semi-trusted "honest-yet curious" service providers thus preventing against insider threat. In our analysis we give the desired properties of the cryptographic tools used and identify open problems. We believe that the approach represents a precursor to new and innovative cryptographic techniques which can provide solutions for the security and privacy problems in federated identity management.

*The work reported in this paper has been partially sponsored by NSF under the ITR Project 0428554 "The Design and Use of Digital Identities" and by the sponsors of CERIAS.

Keywords

Identity management, single sign on, federation, identity theft, cryptographic protocols, zero knowledge proof

1. INTRODUCTION

Digital identity corresponds to the electronic information associated with an individual in a particular identity system. Identity systems are used by online service providers (SP) to authenticate and authorize users to services protected by access policies. With the advent of distributed computing models such as web services, there are increased inter-dependencies among such SP's. As a result, the current trend [14, 12] is to focus on inter-organization and inter-dependent management of identity information [21] rather than identity management solutions for internal use. This is referred to as *federated identity management*. Federated identity is a distributed computing construct that recognizes the fact that individuals move between corporate boundaries at an increasingly frequent rate. Practical applications of federated identities are represented by large multinational companies which have to manage several heterogeneous systems at the same time [21]. An effort in this sense is represented by the notion of *Single Sign-On (SSO)* [25, 23], which enables a user to login to multiple organizations or SP's by using the same username and password. This approach increases usability and adds security by reducing the number of passwords that need to be managed.

Emerging standards [14, 12] are currently extending the notion of federated identity to other user information referred to as *identity attributes*. The main goal of such extensions is to enable interoperability and link together redundant user identities maintained by different SP's. An important requirement in this context is that the federation environment should enable SP's to exchange user data in a secure and trustworthy manner while also enforcing the original privacy preferences of the user. Current federation solutions are built on top of SAML[1] specification which depends on PKI with additional trust relationships for its security. As such, federations have to rely on PKI for exchanging data among SP's, and between users and SP's authentication [8]. However, PKI has experienced numerous implementation problems because of its technical com-

[1]Security Assertion Markup Language (SAML)

plexities. It is also oriented towards strong identity granted through Registration and Certification Authorities, which is not always suitable for user privacy. Hence, the assumption of relying on PKI for all types of interaction in the federation is not realistic. We thus need articulated identity solutions supporting multiple complementary options for digital identity.

A serious concern related with identity management, whatever solution is chosen, is the risk of identity theft. Despite guidelines have been provided on how to protect against identity theft [16, 11], not many identity theft protection solutions have been proposed so far. Sensitive information in the Internet is currently hard to track and also consistent usage of the proposed solutions is extremely hard to achieve. We believe however that in a federation environment it is possible to develop protocols able to achieve identity theft protection. As noted above, the security and privacy of the user identity information, both certified and uncertified, are of utmost importance today. Security prevents theft and impersonation when the identity attributes are used and privacy protects against the disclosure of identity when the user has the right or expectation of anonymity [15].

In this paper we propose a flexible approach to assign unique identifiers to users within a federation employing SSO ID's with strong guarantees against identity theft. To assure user privacy, our protocols do not rely on PKI for user authentication, so that one can easily use uncertified attributes and be eligible for services with low clearance. For cases in which certificates are required we show how SP's can leverage the SSO ID to issue certificates to users. We also show how we can easily employ PKI protocols if a PKI infrastructure is available.

The core of our federated approach for identity management is a set of cryptographic protocols specifically designed to protect user attributes against identity theft. The key idea is to associate the different kinds of sensitive information of a user with each other and with the user's SSO ID. In such way, any of such sensitive information is not acceptable without one or more of the other associated identifying information. We refer to a set of such sensitive information as *attributes Secured from Identity Theft (SIT attributes for short)*. Under our approach, SIT attributes are protected themself: if a user wants to use any of his/her SIT attributes, he/she has to give along one or more other SIT attributes as proofs of identity. We show how we can preserve user privacy without jeopardizing security with the help of cryptographic techniques like zero knowledge proofs [10, 24] and distributed hash tables [18]. The use of zero knowledge protocols make it possible to hide user values of the proofs of identity even to entities like SP registrars[2]. To the best of our knowledge this is the first time a cryptographic solution to the problem of identity theft has been proposed in the context of federated digital identity management. A federation environment inherently protects user attributes more than an open environment. However it is usually assumed that all entities in the federation are completely trusted. In our solution we show how the protocol is robust enough even in the case when semi-trusted "honest-yet curious" SP's are in place. In the paper we also formally prove correctness of our protocols and provide complexity results; these results show that our approach is very efficient.

The remainder of the paper is organized as follows. In the next section we introduce preliminary concepts and definitions concerning digital identity in a federation. In particular, in Section 2.2. we illustrate the approach we adopt to establish digital identity in a federation for both servers and users. In Section 3 we show how certificates are issued and used in the federation. Section 4 gives detailed description of our solution to the problem of identity theft with the help of a running example. In particular Section 4.1 gives our security model followed by Section 4.2 with basic registration protocol for establishing SIT attributes. This is followed by protocols for SIT attributes usage in Section 4.3 and 4.4. In Section 5 we give the formal analysis for the correctness of the protocols and the complexity analysis. In Sections 6 and 7 we discuss related work and we outline some conclusions, respectively.

2. DIGITAL IDENTITY IN FEDERATIONS

In this section we present preliminary concepts related with identity. We first briefly review the notion of identity and possible identifying techniques. Then, we present the identity system we have devised, focusing on the approaches we adopt for identifying both users and SP's.

2.1 Preliminary Concepts and Definitions

A federation is a group of organizations which trust certain kinds of information from any member of the group to be valid. In this paper we consider federations involving two types of entity: SP's and users. A SP is an entity providing one or more services to users within the federation. Services are protected by a set of rules defining the requirements users have to satisfy in order to use the service. Often such requirements are expressed as conditions against properties of users. Such properties are usually encoded by means attributes or credentials.

To interact within the federation, users need to be identified. Identification is the process of mapping claimed or observed attributes of an individual to his/her associated *identifier*. Identifiers can be either encoded using attributes or certificates, or they might be user's knowledge (i.e., passwords, etc). Identifiers are assigned to users by an *identity system*. Identifiers can be classified into weak and strong identifiers. A strong identifier uniquely identifies an individual in a population[3], while a weak identifier can be applied to many individuals in a population. Whether an identifier is strong or weak depends upon the size of the population and the uniqueness of the identifying attribute. Multiple weak identifiers may lead to a unique identification [31]. Examples of strong identifiers are a user's passport or social security number. Weak identifiers are attributes like age and gender. The types of possible identifiers and their organization are summarized in Figure 1. In the remainder of the paper when mentioning identifiers we will always refer to strong identifiers, unless explicitly stated otherwise.

[2]If a user trusts an SP to store his/her hidden SIT attributes then that SP is the registrar for that user. More on this in Section 4.

[3]This is with respect to the domain in which the user is being identified. We do not address the domain specific details in the paper as it is not important to understanding the main idea.

An identity system should satisfy some requirements related with identification and authentication of the identified users. Identification ensures *accountability* of the users and the ultimate goal is to *authorize* users to obtain required services and/or data by service providers. By *authentication* we mean the process of establishing confidence in the truth of some claim. *Authorization* is the process of ensuring that the policies specifying who may execute which actions on which resources are followed. Authorization decisions do not require the unique identity of the requester to enforce policies. Finally, *accountability* is the ability to associate a consequence with a past action of an individual [22]. It is required that the individual can be linked to action or event for which he/she is to be held accountable. Unlike authorization, accountability requires the ability to uniquely identify the individual.

One of the most common approaches for authenticating the website of an enterprise in today's world is the use of public keys. Trusted commercial entities [29, 28] certify that a given public key belongs to that enterprise. Following this practice, we assume that the SP's use the public key infrastructure (PKI). Public keys are thus used to identify and authenticate any SP throughout the federation. Although this approach is adequate for SP's, it is not however suitable for users. Indeed, the management of digital identity by means of user certificates has proven to be very difficult [30]. To understand the reasons of this failure we revisit the two principles identified in [2]. Conforming to them one might selectively reveal elements of his/her identity:

1. The *"least revealing means" principle* implies that the minimal identity information should be provided by the user to complete a particular transaction. This is definitely not true in the current systems because much more information than required has to be given to certification authorities (CA) even for a simple certificate. For example, to be issued a certificate certifying a public key with an email address of a user, the actual identity of the user is irrelevant. However even in this case the user is asked to give his/her critical identity information like passport number, social security number, name, date of birth and so forth [27], to be able to obtain a public key. This is because the CA wants to prevent fraud and excessive unaccountable usage. Nevertheless such an approach violates privacy and therefore new methodologies need to be devised for general users.

2. The *"most convenient means" principle* implies that a user would selectively reveal the combination of identifying information that are most convenient. Here the information should not exceed an upper bound of the amount of information the user is willing to reveal. This principle also points to the need of economy of mechanism and usability. One of the main problems with PKI is the management of the keys and it is not easy for a user to easily first establish a public key, and then use it with different applications. We address the former concern in detail in this paper.

In the following sections we address the problems outlined above in the context of a federation. Our main goal is to establish, manage and use digital identities of users in a federation.

2.2 Establishing Unique Identifiers

In this subsection we focus on how to establish unique identifiers for the entities in a federation, that is, SP's and users.

Service Providers

SP's within a federation are identified uniquely by their public keys. We denote the two keys belonging to each service provider as K_SP_{Pub} and K_SP_{Priv}, denoting public and private key, respectively. SP's are also responsible for verifying user attributes and issuing digital certificates to certify them. Therefore, each SP also has an additional public key shared with all the other providers in the federation, which is used to verify user's certified attributes issued by *any* SP. This public key can be generated by using group key algorithms [7] and can be used to identify any SP belonging to the federation. We denote this public key as K_{FED} and the corresponding private key as $K\_SP_{FedPriv}$. Note that, according to the protocols in [7] each SP has a different $K\_SP_{FedPriv}$ associated with the public key K_{FED}.

Users

A user digital identity is basically a set of his/her identifiers. In our work, we employ single sign-on (SSO) ID's to uniquely identify users within a federation. Users affiliated[4] with a SP are identified by their name and the SP name, separated by symbol \$. That is, if Alice is a user affiliated with SP1, her SSO ID would be $Alice\$SP1$. Users receive their ID's when joining the federation. We assume all sharable attributes of registered users to be certified and stored with the member SP they are affiliated with. External users can also join the federation by establishing their SSO user name and password with any SP within the federation. For example, if some user Bob wants to establish a SSO ID in the federation he sends a request to $SP1$ with desired user name Bob. If this user name does not already exist in $SP1$, $SP1$ registers Bob giving him the SSO ID $Bob@SP1$. Note that other user naming mechanisms could be used here. The essential property is that a user SSO ID be unique within the federation.

After the user has successfully established a SSO ID in the federation, he/she can then establish different types of digital attributes. The SP services that a user can be eligible for depends on the different attributes required by the service policy. In this paper we consider three types of attributes (see Figure 2): 1) uncertified attributes, corresponding to voluntary information given by user; 2) certified attributes, corresponding to attributes that have been verified and issued as signed digital

[4]By affiliated users we mean users who have repeated interactions with one or more SP's in a federation and are interested in a continuous relatioship with the federation as opposed to external users who have only random interactions and are not interested in establishing a relationship with the federation.

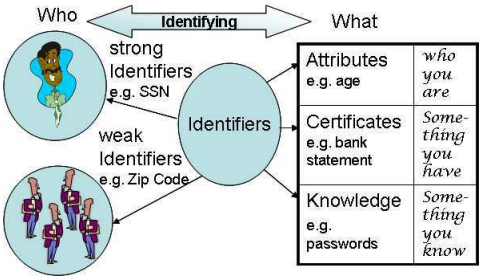

Figure 1: Classification of User Identifiers

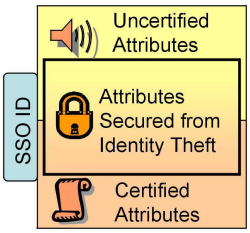

Figure 2: Attribute Types

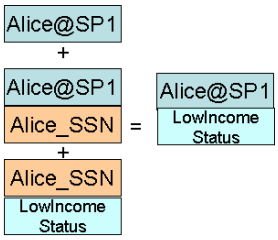

Figure 3: Type A Cert

certificates by trusted SP's or CA's; and 3) attributes secured from identity theft (SIT attributes, for brevity) corresponding to identity attributes that are relevant for the user identification and thus need to be secured by our protection mechanism. Using the SSO ID the user can log on to different SP's and get access to the provided services. We consider three main cases. If the user initially does not have digital certificates he/she can access services for which only voluntary user information needs to be provided. In most cases, the information is for non critical services or is irrelevant for authorization of that service. This is the first and the most trivial case. For services requiring higher clearance and thus requiring certified information from the user, the user has to apply to the SP's or some trusted external CA's for the required certificates. The third and more interesting case is when the user requires access to a given service with protection against identity theft for the user identity attributes that are to be supplied to get access to the service according the the service policies. These attributes can be both certified or uncertified. We present protocols for this purpose later in this paper. Protection of the identity attributes against identity is of course in the best interest for any user. However, also SP's may want to offer reliable and secure services and thus interested in requiring that user attributes be protected against theft.

We elaborate in each of the above cases in detail in the rest of the paper.

3. USAGE OF CERTIFICATES

For a user to be authorized for a service, the SP may require some certified identity information. Certified information is encoded as certificates issued by SP's within the federation or by external CA's. Upon federation setting, it is agreed that the SP's will follow an acceptable well defined procedure for the verification and certification of different attributes. These certificates will be considered reliable within the federation.

3.1 Certificate Provisioning

In order to get certificates usable in the federation the user has to obtain that his/her uncertified attributes and claims are verified by any trusted SP or CA. We employ two basic approaches for verification. The first approach addresses the case of a user that does not possess any initial digital certificate. Here it is inevitable that the user has to go to a physical location to register any strong identifier. If for example Alice, who does not have any digital certificate, needs a certificate asserting that $Alice\_SSN$ is her social security number (SSN), she has to show this to an authorized personnel in a physical office at a SP (say SP1). As a result, she gets a signed digital identity which asserts that $Alice\_SSN$ belongs to user-id $Alice@SP1$.

The second approach is used when a user either already has some digital certificates, or the claimed information can be verified by accessing some reliable online databases. Additional certificates can be issued based on this information. We assume that certificate provisioning policies are in place at the SP. Certificates issued by SP's are of two main types: `Type A` certify's *credential ownership* - here the SP certifies that the user owns a given set of certificates; `Type B` certify's issuance of new credentials depending on existing user certificates and the certificate provisioning policies of the SP.

An example showing the issuance of a certificate of `Type A` is illustrated in Figure 3. As shown user Alice has two certificates. The first certificate is issued by a SP and states that $\{Alice@SP1$ has SSN $Alice\_SSN\}$. The second certificate is from a trusted CA and states that $\{Alice\_SSN$ has a Low_Income_Status, (LIS) for brevity$\}$. Alice wants to get a certificate stating that she (represented by her SSO identifier) has a LIS to be used within the federation without revealing her SSN. Alice can obtain such certificate by submitting a trusted SP the LIS certificate and the certificate associating her SSN with her SSO ID. In return she obtains the final certificate associating $Alice@SP1$ with LIS. Here the actual revelation of the SSN is not required, but just needs to be the same for the two certificates. LIS is signed by the private group key $K\text{-}SP_{FedPriv}$ of the issuing trusted SP and can be verified by any SP in the federation.

When a user requires the issuance of a certificate of `Type B`, he/she must prove possession of pre-requisite certificates according to the certificate provisioning policies of SP's and the federation requirements concerning the provisioning of certificates stating a claims for users. For example if $Alice@SP1$ has the certificate associating the SSO ID with her SSN, she may be eligible to obtain a $Trusted\text{-}User$ certificate. This may be because Alice can uniquely be identified by the federation, hence be held more accountable and therefore can be trusted.

3.2 Sharing User Attributes

In [26] we have shown how a user can negotiate with SP's to submit the appropriate certificates and attributes in order to obtain the requested service. The key idea is that if the user has agreed to share this information within the federation, the SP's would negotiate these sharable user attributes amongst themselves. Such an approach saves that the user from having to repeatedly provide these attributes and thus improves usability. We have also shown that attribute sharing can be achieved efficiently and with privacy guarantees. By using trust negotiation techniques [5] only minimal information about users are required to satisfy the requesting SP's service policy. Otherwise, the privacy of the attributes may be vulnerable as they would reside in multiple locations within a federation some of which might not be trusted by the user. The use of tickets namely *trust tickets* and *session tickets* has been shown to be critical in order to determine the user's past activities and related information in the federation. If the user does not want to share his sensitive identifiers, he/she can directly negotiate with the SP to provide this information when required. The above approach can be used by our identity system for privacy preserving sharing of user identity attributes. We can also use the standard attribute sharing protocols as given in [12, 14].

4. PROTECTION FROM IDENTITY THEFT

Identity theft occurs when a malicious person uses an honest user's personal information such as the users name, Social Security number (SSN), credit card number (CCN) or other identifying information, without his/her permission. In this section we offer one solution to prevent identity theft in a federation. The key idea is to associate the different kinds of sensitive information of a user with each other and the user's SSO ID. Any of such sensitive information is not acceptable without one or more of the other associated identifying information. We refer to the set of such sensitive information as attributes secured from identity theft (SIT attributes for short). This is similar to real world situations where a user is asked for different kinds of sensitive personal information to be assured that the user is really who he/she claims to be. However, in online transactions with different SP's it is desirable that one can prove the possession of a sensitive information without the actual revelation of this information in clear.

For the purposes of clarity we introduce a running example which is used in the following sections.

Example 1 *User Alice has established an ID in the federation namely $Alice@SP1$. She intends to use her CCN and wants to protect it against identity theft. To do this she registers her SSN and CCN with the SP SP_{reg}. Now, within the federation her CCN is not valid unless she provides information regarding her SSN as registered earlier. So when another SP providing a service, say SP_{prov}, asks for her CCN, first her SSN information has to be validated. Following that the CCN information can be used with SP_{prov} successfully. Even if Alice uses her sensitive SSN as a proof of identity she still does not want to reveal this information to the SP's in clear.*

Next subsection gives the security model and assumptions on which our approach relies. Then, we present the bootstrapping procedure which shows how a user can register his/her SIT attributes with *any* SP in the federation. Once the registration is completed, a set of SIT attributes are associated with the user SSO ID and with each other. Following that we describe in detail how these attributes are used. A key feature of our approach is that we avoid the use of a centralized entity, thus being consistent with the truly distributed nature of protocols in the federation. To protect against identity theft is important that an adversary be prevented from registering as its own SIT attributes of other users; therefore we describe how we detect duplicates within a federation.

4.1 Security Model and Assumptions

As introduced, the two main entities in the federation are users and SP's. SP's can also act as *registrars* for certain users. If a user trusts an SP to store his/her SIT attributes, then that SP is the registrar for that user. Registrars are assumed to be semi-honest[5] for the user attributes they keep track of. Users and SP's which are not registrars can be either semi-honest or malicious. The interactions are mainly between two parties at a time and we assume at least one of the two parties be semi-honest. Two of the three approaches presented in the following subsections rely on the Schnorr's zero knowledge proof (ZKP)[24]. To execute the protocols given below, during the formation of the federation the following basic system parameters need to be selected:

1. p: A large prime (i.e, $p \approx 2^{1024}$) such that the discrete logarithm problem in $\mathbb{Z}p^*$ is intractable.
2. q: A large prime divisor of $p-1$ (i.e., $q \geq 2^{160}$).
3. g: $g = \beta^{(p-1)/q} mod\ p$, where β is a primitive root of p. (Note that $g^q mod\ p = 1$ by Fermat's Theorem.)
4. t: A security parameter such that $q > 2^t$. For most practical applications, $t = 40$ will provide adequate security.

Also we assume that the SP's are identified using their unique public keys and the users are identified using their SSO ID.

4.2 Bootstrapping: Registration Procedure

Because a SP is not considered completely trustworthy, the values of the sensitive attributes are not to be released in clear. The main goal of registration is thus to store unique and hidden sensitive SIT attributes to such semi-honest SP's. In the next subsections we describe the two alternative registration procedures we have devised.

4.2.1 Physical Registration

This type of registration requires the user to go to a SP in order to register his/her sensitive identifiers and attributes in person. Following from Example 1, Alice wants to register her SSN and CCN with SP SP_{reg}. An authorized officer of SP_{reg} verifies the actual $a = SSN$ and $b = CCN$ with the physical cards or certified papers. Then he/she lets Alice enter these values in an offline dual screen computer (or some special purpose device) such that the officer monitors with the help of the second screen that Alice enters the correct values. Such computer calculates $g^{-a} mod\ p$ with tag SSN_{tag} and

[5]According to the accepted definition of semi-honest entities, we assume registrars will follow the protocol but may also want to learn more information than they are supposed to.

$g^{-b} mod\ p$ with tag CCN_{tag}. Given the calculated value it is not computationally feasible for an attacker to get the secret values assuming the intractability of Discrete Log Problem (DLP). These values are stored with the user and SP_{reg} with the corresponding user id $Alice@SP1$. Here we assume that the officer is trusted and does not keep a copy of the sensitive information.

Physical registration is the strongest and sometimes the most reliable form of identification. However, referring to our principles introduced in Section 2.1, this method reveals a minimal amount of information, but is not the most convenient procedure. We therefore look into the second kind of registration which is executed through online message exchange.

4.2.2 Online Registration

Online registration of sensitive attributes is a challenging in the absence of user public keys or any electronic certified information. To achieve the goal of privacy, we require that sensitive information of the user are *never* given in clear to even the SP which is storing this information. Of course this requirement adds a level of complexity to the whole registration procedure: the SP cannot guarantee that the information registered is correct, but it can guarantee that the user knows the secret information whose exponentiated value is stored with it. To reach this last goal the SP and the user engage in a zero knowledge proof (ZKP) as given shortly. To understand the former concern let us consider three cases following from Example 1. The first case corresponds to the ideal situation, that is, Alice is honest and submits correct values of the sensitive attributes. The second case is when Alice submits a random value instead of the SSN and tags it with SSN_{tag}. If no one other than Alice knows the random correct value, this works perfectly fine. However if the actual value of the SSN is required to be given in clear, then Alice's transaction will fail. The final case is when a malicious user, say Carl, tries to register Alice's SSN. Now if Alice has already registered her SSN, the attempt by Carl would be detected by the federation as explained in a Section 4.4. If Alice has not registered this attribute and tries to register it later an alarm would be raised. The alarm would then trigger an auditing procedure to determine which user has already registered the SSN of Alice and a subsequent recovery procedure that will undo the registration made by Carl.

The general ZKP's [10] is explained as follows. The protocol allows a committer to have a private secret, and prove its possession without releasing it. The committer releases some information called the *commitment*. The protocol has two main properties: *hiding*, the verifier cannot compute the secret from the commitment; and *binding*, the committer cannot change his mind after having committed, but it can later open the commitment to reveal the secret to convince the verifier that this was indeed the original value it was committed to.

We use Schnorr's ZKP to commit to the supposedly sensitive information as given in Protocol 1. Following from Example 1, if Alice wants to commit her SSN$= a$ then she commits the value $c = g^{-a} mod\ p$ to SP_{reg}. Using the protocol she can prove that she knows the value a corresponding to the commitment. Similarly, she can commit other sensitive values.

Protocol 1 SIT Registration: Schnorr's Zero Knowledge Protocol

Require: *Federation System Parameters*: p, q, g such that $q|p-1$, and g is an order q element in $\mathbb{Z}p^*$. t is a security parameter. $User$ has a valid SSO ID uid, Service Provider SP_{reg} is a member of the Federation. Time stamps T_i can be generated.

Goal: User knows the secret a of the committed value c which is registered with SP_{reg}.

1: $User \rightarrow SP_{reg} : c = g^{-a} mod\ p, uid, T_1$
2: $User \rightarrow SP_{reg} : d = g^r mod\ p$ {User selects r from $[1..q]$},uid, T_2
3: $SP_{reg} \rightarrow User : e$ {SP_{reg} selects e from $[1..2^t]$}
4: $User \rightarrow SP_{reg} : y = r + ea\ mod\ q, uid, T_3$
5: SP_{reg} : **if** $d = g^y * c^e mod\ p$ **then** return OK

At the end of the registration procedure in Example 1 the registrar SP_{reg} has the information as given in Table 1. We

Tag	Committed Value	Registration Procedure
SSN_{tag}	g^{-a}	In Person, T_1
CCN_{tag}	g^{-b}	Online, T_2

Table 1: Information registered for Alice at SP1 *(Refer example 1)*.

now describe how the SIT attributes, that are registered through either the bootstrapping or the registration procedure, are used in the federation to protect against identity theft.

4.3 Using SIT Attributes to Protect Against Identity Theft

The main aim of the protocols we present here is to make the use of SIT attributes possible only under the submission of a subset of additional SIT attributes which have been registered. The exact subset of the additional SIT attributes required is determined based on the user and/or SP's identification policy. Such attributes act as a proofs of identity and enable association of required SIT attributes with the other registered SIT attributes. This gives assurance that the user is in control of his/her sensitive attributes and is therefore honest. The solution we have devised to deal with identity theft consists of two main phases. The first and key phase is a *ZKP and symmetric key exchange* protocol. Here the user proves that he/she knows the actual value of a specified SIT attribute without revealing its value in clear. In addition, at the end of a successful run of this protocol the user and SP share a single symmetric key related to the proof of knowledge of that SIT attribute. With repeated runs of this protocol multiple symmetric keys can be shared. These keys are used to retrieve the required SIT attributes from the final message given by the user to the SP. *Creation of the final message* is the second phase of the solution. Here, the user creates a message encrypted in a nested manner with the symmetric keys generated in the first phase. Next we elaborate on these two phases in detail.

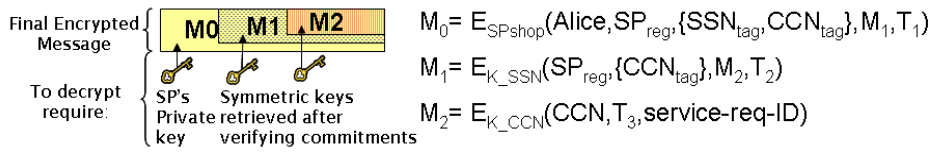

Figure 4: Final message format from $Alice$ to SP_{prov}. *Refer example 1*

Protocol 2 ZKP and single symmetric key exchange

Require: *Federation System Parameters*: p, q, g such that $q|p-1$, and g is an order q element in $\mathbb{Z}p^*$. t is a security parameter. $User$ has a valid SSO ID uid and has registered his/her attributes with Service Provider SP_{reg} and wants service from SP_{prov}. Both the $SP's$ are members of the Federation. Time stamps T_i can be generated.

Goal: SP_{prov} verifies correctly that $User$ knows the value a registered with SP_{reg} to retrieve the key k.

1: $User \rightarrow SP_{prov}$: msg $\{msg = SP_{reg}$ has my (uid) commitment for secret for $tag\text{-}of\text{-}a\}, T_1$

2: $SP_{prov} \leftrightarrow SP_{reg}$: c {retrieves $c = g^{-a} mod\ p$ corresponding to $tag\text{-}of\text{-}a$ and $user\text{-}id$}

3: $User \rightarrow SP_{prov}$: $d = g^r mod\ p$ {User selects r from $[1..q]$}, uid, T_2

4: $SP_{prov} \rightarrow User$: e $\{SP_{prov}$ selects e from $[1..2^t]\}$

5: $User \rightarrow SP_{prov}$: $y = r + ea\ mod\ q$, $y' = r + ea + x\ mod\ q$, $\{x$ is a random such that $k = d(g^x - 1)\}$, uid, T_3

6: SP_{prov} : *verifies* $d = g^y * c^e mod\ p$ and *evaluates key* $\{(g^{y'} * c^e) - d\} mod\ p = k$

The ZKP and symmetric key sharing is given in Protocol 2. There are three entities involved in this protocol; namely the user, the SP from which the user wants service, denoted as SP_{prov}, and the SP which registered the SIT attributes of the user, denoted as SP_{reg}). In step 1 the user lets the SP_{prov} know that SP_{reg} has his/her committed values. Then in step 2 SP_{prov} confirms this claim with SP_{reg} and gets the required commitments corresponding to the SIT attributes as proofs of identity and required SIT attributes. Similar to Schnorr's protocol in steps 3,4 and 5 the user generates a proof depending on the random challenge sent by SP_{prov}. The main difference is in step 5 where the user calculates y' depending on the random symmetric key k. Here x is randomly chosen so that the resultant key is also random. For the standard symmetric ciphers there is a short list of weak keys which are avoided at this step. The user also generates y to prove it knows the value of the commitment like the original proof in Protocol 1. In step 6, SP_{prov} verifies the claim and retrieves the symmetric key generated by the user in step 5. SP_{prov} can get the key k only if knows the value of the secret commitment stored in SP_{reg}. This protocol is repeated in order to obtain a symmetric key as a proof of knowledge for each SIT attribute required.

Once the symmetric keys are generated and shared, the user creates the final message. If there are n SIT attributes required as proofs of identity and m SIT attributes required for satisfying the requested service policy, then Protocol 2 is run $N = n + m$ times resulting in N symmetric keys. The final message is encrypted N times in a nested manner. The required information is revealed only in the inner-most encrypted portion. As a clarifying example we show in Figure 4 the format of final message that $Alice$ would give to SP_{prov} in Example 1. Through 2 iterations of Protocol 2, the keys K_{SSN} and K_{CCN} are retrieved. M_0 is the main message sent to SP_{prov} which is encrypted with the SP's public key. By reading this message SP_{prov} knows it has to use K_{SSN} to open one layer of encryption thus retrieving the deciphered version of M_1. It now knows it has to use K_{CCN} to get the final value, that is, CCN. SP_{prov} can also confirm at this point that this corresponds to the value committed to SP_{reg} which it received during the last run of Protocol 2.

4.4 Identifying Duplicates of SIT attributes

During registration it is very important to verify if the proposed commitments of sensitive attributes are already registered in the federation. Such verification is to prevent a malicious user from registering stolen SIT attribute values with his/her credentials. Therefore to prevent duplicates the responsible SP should check with all other SP's if the proposed commitment of a sensitive attribute is already present. Such a check can be executed incrementally or via a broadcast and it can be very expensive. We therefore propose the use of distributed hash tables (DHT)[18, 6] for this purpose.

A DHT has no central server and partitions a key space among *n* servers. Each distributed server has partial list of where data is stored in the system and the keys are mapped uniformly to the servers according to set rules. A *"lookup"* algorithm is required to locate data given the key for that data. There are two main functions for handling the data, namely $put(key, data)$ and $get(key)$.

For our purposes the DHT is used as follows. The unique identifier or the key for the DHT is the commitment $c = g^{-a}$ as given in Protocol 1. The key space therefore has the range $[0..p-1]$[6] and the data is the tuple $(user\text{-}ID, TAG, Type\text{-}of\text{-}Registration)$. Because a federation is a closed system, there is an inherent trust amongst the SP's; therefore we can use the SP's for storing and retrieving the values [17]. During the formation of the federation a range of key values are given to each SP which will be responsible for storing the given keys and the corresponding data values. This is in addition to the SP with which the user had registered in the first place. When a user wants to register his/her attributes with an SP, that SP ex-

[6]All the system parameters are consistent with Protocol 1 and 2.

ecutes the $lookup$ algorithm to find if any duplicate is present. If a duplicate is present then an alarm is raised that trigger a procedure that determines the compromise of the sensitive attribute. Otherwise, first the SP registers the users commitments. Following that, depending on the range the committed value belongs to, the appropriate SP is given this key and the corresponding data. This replication adds to the robustness and security of the DHT's. As a result, we claim that by using the DHT's we can prevent duplication of registered sensitive identifiers, which is crucial to ensure effective protection against identity theft.

5. ANALYSIS AND DISCUSSION

We now analyze the security and complexity of the SIT protocols. In particular we assess the identity theft protection in the presence of malicious parties and the communication costs. Before evaluating the above measures we present an interesting paradox based on the desired properties for identity theft protection in the federation. The two required properties are as follows:

Property 1: *Identity Hiding.* Given $f(x)$, it is infeasible to compute the value of x.

Property 2: *Duplicate Detection.* Given $f(x)$ and $f(y)$ identify the case when $x = y$.

The first property is required so that the registrar SP, who stores the committed values of the user is not able to compute the actual secret value. If that SP could compute the values the process of registration could be simplified greatly to store all the values in clear. Then, if this SP is compromised so are all the SIT attributes. The second property is required to prevent duplicates of sensitive identifier commitments in the federation. This requirement is needed to prevent a malicious user from registering stolen attributes with his/her identifier. It is interesting to observe that property 2 enables one to launch a brute force dictionary attack. This is a realistic attack for most identifiers. For instance a SSN is composed of 9 digits, therefore it has $10^9 < 2^{30}$ possible SSN's. Listing 2^{30} possible strings for a 30 bit value is not difficult for an adversary with moderate computational resources. It is not obvious how padding or randomization can be added in a useful manner. We however suggest that the typical sensitive identifiers should be appended with other related information. For example a credit card number can be appended with expiry date and name resulting in a longer length of this sensitive information. The above paradox, however, is an open problem which may be solvable using innovative cryptographic techniques and security models.

5.1 Identity Theft Protection in The Presence of Malicious Parties

We now prove some relevant properties of the SIT attribute registration and the SIT attribute usage protocol. As illustrated in Example 1 there are mainly three kinds of entities: the user; the registrar SP_{reg}; and the SP_{prov} which will be providing the service. We assume that the SP_{reg} is semi-honest.

In the following two theorems we prove the correctness and confidentiality properties of the registration protocol. By correctness we mean that an honest user can execute the protocol successfully achieving the specified results. By confidentiality we mean privacy preservation of the registered user attributes.

Theorem 1 *Let U be a user and let $Attr$ be the set of attributes U wishes to protect. Protocol 1 ensures registration attributes $Attr$ in identity-protected, even in the presence of malicious users.*

PROOF. We prove that a malicious user cannot compromise an honest user SIT attribute registration. Two possible cases arise: i) U registers a set of attributes *before* a malicious user tries to re-register a subset of those attributes with his/her actual attributes instead, ii) U registers a set of attributes *after* a malicious user has registered U's stolen attributes.

In case i), after the honest user has successfully registered his/her SIT attributes, a malicious user will not be able to re-register those attributes with his/her own SSO ID. This is ensured by the duplicate detection mechanism given in Section 4.4. Here, we assume that the exact value of the committed sensitive attribute is important for the validity of the attribute. Therefore when the adversary attempts to re-register the commitment will look identical to the one sent by the honest user. The registrar service provider detects this duplicate and hence denies the registration.

In case ii), attributes are obviously SIT attributes only after they are registered. As such, it is required that users be aware of their sensitive identifiers and register them in a timely fashion. If the user tries to register with any SP in the federation, like in case (i), a duplicate is detected and a physical verification is requested by the SP. In this case the user can give a valid in-person verification as discussed in Section 4.2.1 and then re-register the values successfully. The thesis thus holds. □

Theorem 2 *The SIT attribute registration protocol satisfies confidentiality.*

PROOF. We prove that the actual sensitive values of the registered commitments is not revealed even to the registrar SP_{reg}. The attribute registration prevents guessing the values stored with it. This is directly related to the *hiding* property of Schnorr's ZKP protocol. The key assumption is that the Discrete Log Problem [19][7] is hard for a polynomially bound adversary. If the length of the committed attribute is more than sixteen bits, then by current standards it is infeasible for an adversary to launch a dictionary attack to guess the value of the committed value or compute the discrete log. The actual secret of the commitment thus remains confidential. □

Corollary 2 Registrar SP cannot infer *other* related attributes based on the ones stored with it.

PROOF. (Sketch) It has been shown in [31] how combining attribute information about a user can help infer his/her other not directly disclosed attributes. Because the actual values of the attributes remain confidential even to SP_{reg}, we see that it is not possible to infer other attribute information from the

[7]Given a multiplicative group $(G, *)$, an element g in G having order n and an element y in the subgroup generated by g, we have to find the unique integer x such that $g^x mod\ n = y$. Here x is the discrete logarithm $log_g y$.

set of committed values. The only values given in clear are the $tags$ associated with the given commitments. These $tags$ are required to be generic so that they do not leak information about the corresponding secret attribute. For example instead of having a tag *Purdue-Student-ID* the tag should be *Affiliated-Institution-ID*. Here the latter generic tag does not have the specific identifiers like *Purdue* and *Student*. There is no other information in clear which could leak information; therefore inferring information about the user is hard. □

The next two theorems prove the correctness and confidentiality of the SIT attribute usage protocol. By correctness of SIT attribute usage protocol we mean that the proofs of identity can be used successfully to prove ownership of the attributes required by the SP_{prov}. Only after this proof does SP_{prov} get the required attributes. Correctness of the SIT attribute usage protocol ensures mitigation of identity theft. By confidentiality we mean that the protocol is privacy-preserving with respect to the user attributes such that none of the SP's learn more information than required about the user.

Theorem 3 *Let U be a user and SP_{prov} be the service provider U is interacting with. SIT attribute usage protocol is correct if and only if at least one of the interacting parties is semi-honest.*

PROOF. To prove that SIT attribute usage protocol is secure we need consider two cases: i) U is malicious and SP_{prov} is semi-honest.
ii) SP_{prov} is malicious and U is semi-honest.

Case i). A malicious user cannot provide the commitment corresponding the the different proofs of identity required by the SP_{prov}'s service policy. We assume that not all the sensitive SIT attributes are compromised. Due to incorrect commitments the symmetric key exchange in Protocol 2 fails. Referring to the protocol, the malicious user could easily articulate y' using the equation: $\{(g^{y'} * c^e) - d\} mod\ p = k$ used by the SP_{prov} in step 6 to exchange a symmetric key successfully. However, y can only be verified if the user knows the secret using the original ZKP. Therefore this protocol is secure against a malicious user.

Case ii). In our context a malicious SP_{prov} is a SP wishing to use SIT attributes of the user *without* verifying the proofs identity. This not possible because of the format of the final message disclosed by the user (see Figure 4). Protocol 2 for key exchange is successful if and only if the committed value is verified correctly with SP_{reg}, for each such attribute. The messages are encoded in a nested manner such that only after decrypting with the keys corresponding to the proofs of identity can the SP_{prov} retrieve the required user attributes. This forces the SP_{prov} to follow the protocol and prevent misuse of an honest user SIT attributes. □

It is important to notice that in Protocol 2 we require at least one party between the user and the service providing SP is semi-honest during the message exchanges. The worst case arises when the registrar and another SP in the federation collude. In this case, we cannot prevent the leak of information such that the registrar collects clear attributes from $SP'_{prov}s$. However, we will explore approaches to detect this misbehavior in our future work.

Theorem 4 *The SIT attribute usage protocol satisfies confidentiality.*

PROOF. SP_{prov} is not required to learn any information about the sensitive attributes used as proofs of identity in the SIT attribute usage protocol. Only the tag corresponding to the identity proof is given to SP_{prov} to query SP_{reg} and retrieve the corresponding commitment. This tag as specified earlier has to be generic to avoid leaking any secret information. If SP_{prov} can get the value of the secret identifier in the commitment, then it would be equivalent to solving the DLP problem. This contradicts our assumption that DLP is hard. Therefore information about the SIT attributes as proofs of identity remains confidential to the SP_{prov} type SP's. □

In addition, the illustrated protocols are secure against man-in-the-middle and replay attacks. This is because of four main reasons. First, any message sent from the user to the SP is encrypted with the public key of the SP. Second, an explicit naming convention [1] is used by including the SSO ID of the sender. Third, timestamps are used to maintain freshness of the messages. Finally, the challenges sent by the SP are random[8] and cannot be predicted. For the final message if an adversary could successfully replay this message then it could essentially use the SIT attribute with the attached proofs of identity. This is not possible because the symmetric keys are generated in response to random challenges sent by the SP and the proofs of identity. Interestingly, in this manner even the SP_{prov} which successfully retrieves the user's SIT attributes as required cannot maliciously use them with any other SP. The timestamps in the final message also prevents timing and replay attacks. Note that timestamps can be replaced by counters or nounces, as suitable for the federation environment.

5.2 Complexity Analysis

The complexity cost is estimated in terms of the number and sizes of messages exchanged among SP's and users. In the registration phase the Protocol 1 is executed for each registered SIT attribute. The number of messages exchanged for each iteration is four. The sizes of these messages are in $log\ (p)$ or $log\ (q)$ depending of the modulus. For Protocol 2 in the attribute usage protocol, the number of messages exchanged is five. Furthermore, let n be the number of proofs of identity required to gain assurance regarding the validity of a user, and m be the number of required attributes. Then the number of times Protocol 2 has to be run is $N = n + m$. The sizes of most messages are of the same order of the sizes of the messages exchanged during the registration phase. The size of the final message after all iterations of Protocol 2 are executed is proportional to N, and the number of nested encrypted messages is N. If symmetric cipher AES is used in the CBC mode [3], then the size of each nested block is at least 128 bits. As one cipher block is added with each encryption the size of the

[8]Note that these random challenges can be made non-interactive assuming a random oracle model [4] but this is outside the scope of the paper.

final message is approximately $128 \times N$ bits. The registrar SP_{reg} acts like a database of information, and thus it does not represent a bottleneck in the system. To enhance efficiency, Protocol 1's multiple attributes registration can be executed in parallel because the commitments are independent of each other. Similarly Protocol 2's multiple symmetric key retrieval can also be made parallel and according to any order.

6. RELATED WORK

In this section we first explore the most relevant federated digital identity management initiatives and then solutions to the identity theft problem in federations. In the corporate world there are several emerging standards for identity federation like Liberty Alliance and WS-Federation. Because the projects are very similar we describe the former in more detail below. Liberty Alliance [12] is based on SAML and provides open standards for SSO with decentralized authentication. SSO allows a user to sign-on once at a Liberty-enabled site in order to be seamlessly signed-on when navigating to another Liberty-enabled site without the need to authenticate again. This group of Liberty-enabled sites is a part of what is called a *circle of trust*, which is a federation of SP's and identity providers having business relationships based on the Liberty architecture. The identity provider is a Liberty-enabled entity that creates, maintains and manages identity information of users and gives this information to the SP's. The users authenticate themselves to an identity provider in the federation and other SP's obtain authentication information of the user from this identity providers. As compared to Liberty Alliance which uses PKI for user authentication, we show how we can also leverage the SSO ID for establishing from simple to complex digital user attributes. This adds privacy, flexibility and usability to the identity system. In addition our specific identity theft protection protocols can prove valuable when used in the Liberty identity federation framework.

Shibboleth [14] is an initiative by universities member of Internet2 [13]. The goal of such initiative is to develop and deploy new middleware technologies that can facilitate inter-institutional collaboration and access to digital content. It uses the concept of federation of user attributes. When a user at an institution tries to use a resource at another, Shibboleth sends attributes about the user to the remote destination, rather than making the user log in to that destination. The receiver can check whether the attributes satisfy its own policies. An identity provider in the Shibboleth architecture has all the user attributes and user privacy preferences which are taken into account when it has to supply user identity information to other members of the federation. Our approach differs with respect to Shibboleth in that we do not rely on a central identity provider providing all user attributes. User attributes in our framework are distributed within the different federation members, each of which can effectively be an identity provider. We also provide a mechanism by using which a user can get certified attributes from the federation members and use them to obtain further certified information.

Concerning the problem of identity theft, Liberty Alliance (LA), Shibboleth project and other organizations like Better Business Bureau and Federal Trade Commission have initiated some efforts aiming at educating consumers and preventing identity theft. A LA paper [9] points out that having SSO in federations helps reduce ID theft by reducing the number of login names and passwords which might be related to other user attribute information. The paper also discusses how attribute sharing in a federation inherently prevents user attributes theft *"by controlling the scope of access to participating websites, by enabling consent-driven, secure, cross-domain transmission of a users personal information."* LA mitigates ID theft by having the organizations in the federation adopt superior standards of security, distributing information to avoid single point of failure, by having access control on these attributes based on user preferences, and having coordinated response to incidents and frauds. However to the best of our knowledge no identity provision protocols which mitigate ID theft have been developed dealing with the case when the members of the federation are not completely trusted. Solutions dealing with such case would provide protection against insider threat or when a service provider is compromised. Our solution not only exploits the advantages of a federation, as the general usage case, but extends it even further with the concept of SIT attributes and SIT attribute usage.

RSA Laboratories' product Nightingale [20] implements a secret-splitting technology, which is designed to be integrated into application software as a server module for the back-end of any network. Secret splitting is a cryptographic technique that breaks a piece of data into two components. Learning one of these components reveals no information about the original data. Using secret-splitting the sensitive data is cryptographically distributed across two locations - the Nightingale module/server and an application server thus avoiding a single point of failure. The secret data can be of three types: (1) user authentication data like SSN, passwords; (2) business data like customer records and their CCN; (3) the cryptographic keys themself. Nightingale can thus be used to mitigate identity theft by making it hard to retrieve the stored user information. Interestingly, the secret splitting can be used with the solution proposed in this paper to split the committed values of user identity attributes. Such an approach may provide even better identity theft protection which we will explore as a part of our future work.

7. CONCLUSION

In this paper we have proposed a flexible and privacy-preserving approach that allow a user to establish a unique identifier and then proceed to establish other complex identity attributes in a federation. Our approach relaxes the dependence on PKI for user authentication which is currently a bottleneck for many trust management solutions. We also presented a novel solution to the problem of identity theft based on cryptographic techniques. In the paper we have also analyzed the security and complexity of the proposed protocols. The analysis also gives the assumptions and properties that are required in order to implement the given identity theft protection solution and an open paradox which still needs to be addressed. Our future work includes moving from a context characterized by the semi-honest paradigm to a context where malicious SP's can collude with each other and the development of techniques to

detect misbehavior. We will also explore how other cryptographic techniques can be integrated with the one presented in this paper. In essence we believe that our approach encourages the development of innovative cryptographic techniques to address security and privacy problems in digital identity management.

8. REFERENCES

[1] M. Abadi and R. Needham. Prudent engineering practice for cryptographic protocols. *IEEE Trans. Softw. Eng.*, 22(1):6–15, 1996.

[2] H. Abelson and L. Lessig. Digital identity in cyberspace. In *White Paper Submitted for 6.805/Law of Cyberspace: Social Protocols*, 1998.

[3] AES. http://csrc.nist.gov/cryptotoolkit/aes/.

[4] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[5] E. Bertino, E. Ferrari, and A. C. Squicciarini. Trust-χ: A Peer-to-Peer Framework for Trust Establishment. In *IEEE Transactions on Knowledge and Data Engineering*, pages 827– 842. IEEE, July 2004.

[6] J. Byers, J. Considine, and M. Mitzenmacher. Simple load balancing for distributed hash tables, 2002.

[7] X. Chen, F. Zhang, and K. Kim. A new id-based group signature scheme from bilinear pairings, 2003.

[8] A. Durand. Federated identity and pki collide.

[9] W. Duserick and F. Investments. Whitepaper on liberty protocol and identity theft. In *Liberty Alliance Project*, 2004.

[10] U. Fiege, A. Fiat, and A. Shamir. Zero knowledge proofs of identity. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 210–217, New York, NY, USA, 1987. ACM Press.

[11] F. T. C. for the Consumer. Id theft: What it's all about.

[12] Identity-Management. Liberty alliance project. http://www.projectliberty.org.

[13] Internet2. http://www.internet2.edu/.

[14] Internet2. Shibboleth. http://shibboleth.internet2.edu.

[15] K. Klingestein. Emergence of identity service providers, 2002.

[16] I. T. Management. http://www.identitytheftmanagement.com/.

[17] G. S. Manku. Balanced binary trees for id management and load balance in distributed hash tables. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 197–205, New York, NY, USA, 2004. ACM Press.

[18] G. S. Manku. *Dipsea: a modular distributed hash table*. PhD thesis, 2004. Adviser-Rajeev Motwani.

[19] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Handbook of applied cryptography, 2001.

[20] R. L. Nightingale. http://www.rsasecurity.com/rsalabs/node.asp?id=2424.

[21] E. Norlin and A. Durand. Whitepaper on towards federated identity management. In *Ping Identity Corporation*, 2002.

[22] N. R. C. of the National Academies. *Who Goes There? Authentication Through the Lens of Privacy*. The National Academies Press, Washington, D.C., 2003.

[23] A. Pashalidis and C. J. Mitchell. Impostor: A single sign-on system for use from untrusted devices.

[24] C. P. Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO '89: Proceedings on Advances in cryptology*, pages 239–252, New York, NY, USA, 1989. Springer-Verlag New York, Inc.

[25] R. SEMANK. Internet single sign-on systems.

[26] A. B. Spantzel, A. C. Squicciarini, and E. Bertino. Integrating federated digital identity management and trust negotiation. In *review IEEE Security and Privacy Magazine*, 2005.

[27] Thwate. Documentation required to request a thawte certificate, http://kb.thawte.com/esupport/thawte/esupport.asp?id=vs1515.

[28] Thwate. http://www.thwate.com.

[29] Verisign. http://www.verisign.com.

[30] A. Whitten and J. D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *8th USENIX Security Symposium*, 1999.

[31] D. Woodruff and J. Staddon. Private inference control. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 188–197, New York, NY, USA, 2004. ACM Press.