IDIS
DOI 10.1007/s12394-009-0022-6

Minimal credential disclosure in trust negotiations

Federica Paci · David Bauer · Elisa Bertino · Douglas M. Blough · Anna Squicciarini · Aditi Gupta

Received: 31 March 2009 / Accepted: 26 July 2009

Abstract The secure release of identity attributes is a key enabler for electronic business interactions. Users should have the maximum control possible over the release of their identity attributes and should state under which conditions these attributes can be disclosed. Moreover, users should disclose only the identity attributes that are actually required for the transactions at hand. In this paper we present an approach for the controlled release of identity attributes that addresses such requirements. The approach is based on the integration of trust negotiation and

F. Paci (✉)
Department of Information Engineering and Computer Science, University of Trento, Povo, Trento, Italy
e-mail: paci@disi.unitn.it

D. Bauer
Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: gte810u@mail.gatech.edu

E. Bertino
Cs Department, Purdue University, West Lafayette, IN 47907, USA
e-mail: bertino@cs.purdue.edu

D. M. Blough
Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: doug.blough@ece.gatech.edu

A. Squicciarini
College of Information Sciences and Technologies, The Pennsylvania State University, University Park, PA 16802, USA
e-mail: acs20@psu.edu

A. Gupta
Cs Department, Purdue University, West Lafayette, IN 47907, USA

Published online: 01 October 2009

minimal credential disclosure techniques. Trust negotiation supports selective and incremental disclosure of identity attributes, while minimal credential disclosure guarantees that only the attributes necessary to complete the on-line interactions are disclosed.

Keywords Identity attributes · Micro-claims · Minimal disclosure

Introduction

As activities such as shopping, discussion, entertainment and business collaboration are increasingly being conducted on Internet, digital identity management systems have become fundamental to underpinning accountability in business relationships, controlling the customization of the user experience, protecting privacy, and adhering to regulatory controls. Digital identities consist of data referred to as *identity attributes* that describe an individual, such as the social security number, the date of birth, and the country of origin. An identity attribute consists of an attribute name, also called identity tag, and a value.

Managing identity attributes raises a number of challenges. On one hand, identity attributes need to be shared to speed up and facilitate authentication of users and access control, which is often based as policies expressed as conditions against identity attributes. On the other hand, they need to be protected as they may convey sensitive information about an individual and can be a target of attacks like identity theft. Key requirements related to digital identity management systems are correctness, integrity and confidentiality of identity attributes. Correctness is important to assess identity attributes validity. Integrity requires data not to be tampered. Confidentiality deals with the protection of sensitive identity information from unauthorized disclosure. Identity information should only be accessible by the authorized recipients. It is therefore essential to provide mechanisms for confidential release of individuals' attributes. Users should also have control over the release of their identity attributes and be able to state the conditions for attributes disclosure. Users should be able to disclose only the identity attributes actually needed for authentication and access control according to the *least disclosure principle* (Bishop 2004). In fact, users are often required to reveal more information than necessary to complete digital transactions.

In this paper we present an approach for the controlled release of identity attributes, based on the integration of trust negotiation and minimal credential disclosure techniques, which addresses the above requirements. Trust negotiation supports the selective and incremental disclosure of identity attributes while minimal credential disclosure supports the least disclosure principle.

The paper is organized as follows. Next section gives an overview of our approach for the secure release of identity attributes based on the integration of the Trust-*X* negotiation framework (Bertino et al. 2004) and the minimal credential disclosure technique by Bauer et al. (2008). We then summarize the negotiation language and the negotiation protocol supported by Trust-*X*. Next, we introduce the notion of micro-claims, followed by the presentation of the negotiation protocol based on the use of the minimal disclosure technique. We proceed with presenting

the experimental results. The final sections outline related work and conclude the paper.

Approach technical overview

Trust negotiation is an approach for establishing trust in open distributed systems. Trust negotiation establishes trust online between two negotiating parties through bilateral credential disclosure. Digital credentials are assertions stating one or more identity attributes of a given subject, referred to as the owner, certified by trusted third parties called certification authorities (CAs). A key aspect of trust negotiation is that sensitive credentials may be protected by disclosure policies specified by credential owners. Disclosure policies state the conditions under which credentials can be released during a negotiation. Conditions are expressed as constraints against the credentials possessed by the interacting parties and the identity attributes encoded in the credentials. Therefore, trust negotiation supports the selective and incremental disclosure of identity attributes but does not guarantee compliance with the least disclosure principle. In conventional trust negotiations, users may exchange credentials that also contain identity attributes not actually requested by the counterpart's disclosure policies.

An interesting approach supporting fine-grained disclosure of identity attributes has been proposed by Bauer et al. (2008). Such approach is based on the notion of micro-claims, that is, predicates specified in terms of identity attributes. Examples of birth-date-related micro-claims are "Age≥18", and "Age≥21". The set of all micro-claims derived from the identity attributes of a user is collected in a unique credential, represented as a Merkle hash tree (Merkle 1987) in which the leaf nodes represent the micro-claims. The use of such Merkle hash tree credentials makes it possible to dynamically specify an arbitrary subset of micro-claims for a given interaction without disclosing the other micro-claims. This approach also assures integrity of the micro-claims because the credential is released and signed by a CA. The major drawback of such approach is that it requires the users to determine which micro-claims to release for a given transaction, and thus it needs to be integrated with a tool able to automatically select, on behalf of the user, the micro-claims to release.

Therefore, the integration of trust negotiation and minimal credential disclosure is the key to a solution to the problem of identity attribute disclosure addressing all requirements we have outlined. To date, however, there is no system combining those two approaches. In this paper, we make a step towards the development of such system and discuss how to extend Trust-*X* with the minimal credential disclosure technique. The resulting system allows users to disclose only the identity attributes necessary to satisfy the counterpart disclosure policies. Rather than sending all the credentials requested by the counterpart policies, clients, on behalf of their users, send only one credential that contains the micro-claims about the identity attributes specified in the counterpart's disclosure policies. The integration of the two approaches is not trivial. Because verifying whether clients have certain properties is based on proving that they own certain micro-claims, we had to extend the Trust-*X* negotiation language in order to specify disclosure policies protecting the

release of single attributes rather than credentials, and expressing conditions against these attributes. Moreover, because clients should be able to automatically select the micro-claims that satisfy the counterpart disclosure policies, we had to devise different strategies to match the micro-claims with the attributes in the disclosure policies and to select only the micro-claims that satisfy the policies.

Before describing how the integration is realized, we give an overview of Trust-X and the minimal credential disclosure approach.

Trust negotiation

Trust-*X* is a comprehensive XML-based framework for trust negotiations for peer-to-peer environments. In what follows, we give an overview of *X -TNL* (Bertino et al. 2004), the negotiation language supported by Trust-*X* and then of the Trust-*X* negotiation protocol.

Trust-X language

X -TNL is the XML-based language supported by Trust-X. *X -TNL* credentials are the means to convey information about the profile of the parties involved in the negotiation. A credential is a set of identity attributes of a party issued by a CA. All credentials associated with a party are collected into a unique XML document, referred to as *X- Profile*. The disclosure policies state the conditions under which a resource or a credential can be released during a negotiation. Conditions are expressed as constraints on the attribute credentials. Each party adopts its own Trust-*X* set of disclosure policies to protect its own credentials, policies, and resources. Like credentials, disclosure policies are encoded using XML. Two building blocks for specifying disclosure policies are *terms* and *R-Terms*. A term is an expression of form *P (C)* where *P* is a credential type and *C* is a (possibly empty) list of conditions on the attributes encoded in credentials of type *P*. The credential type *P* can be unspecified (and denoted by a variable), so to express constraints on the counterpart properties without specifying from which types of credential such properties should be obtained from. The receiver of the policy can choose which credentials to send as a proof of policy satisfiability. R-Terms are expressions of the form *ResName (attrset)* where *ResName* denotes a resource name and *attrset* denotes a set of attributes, specifying relevant characteristics of the resource. Examples of resources are a credential, a file or a Web service. Disclosure policies can thus assume one of the following forms:

1) $R \leftarrow T_1, T_2, ., T_n, n \geq 1$, where $T_1, T_2, ., T_n$ are terms and R is an R-Term identifying the name of the target resource.
2) $R \leftarrow$ DELIV. A rule of this form is called delivery rule, meaning that R can be delivered as is.

A disclosure policy is *satisfied* if the stated credentials are disclosed to the policy sender and the policy conditions (if any) evaluated as true, according to the specific credential content. A delivery rule implies that the resource *R* is ready to be released, and no specific requirement has to be satisfied.

EXAMPLE 1 The following are examples of disclosure policies:

- Driving License(State = Indiana)←PictureID(Age >18)
- PictureID←BMV(State = Indiana)

The first policy states that in order to obtain a Driving License credential in Indiana the user must disclose a picture id that proves he is older than 18. The second policy says that in order to release his PictureID, the user wants to see a credential proving that the counterpart is an authorized Bureau of Motor Vehicle (BMV) branch of Indiana.

The Trust-X negotiation protocol

In Trust-*X*, the negotiation is performed in two main phases: the policy *evaluation phase* and the *credential exchange phase*. The policy evaluation phase, the key phase of Trust-*X*, consists of a bilateral and ordered policy exchange. The goal is to determine a sequence of credentials, called *trust sequence*, satisfying the disclosure policies of both parties. During each interaction, one of the two parties sends a set of disclosure policies to the other. The receiving party verifies whether its *X-Profile* satisfies the conditions stated by the policies, and whether there are local policies regulating the disclosure of the credentials requested by the policies sent by the other party. If this is the case, the receiving party sends to the other party the disclosure policies protecting the credentials requested by the other party. Otherwise, the receiver informs the other party that it does not possess the requested credentials. The counterpart then sends an alternative policy, if any, or ends the process. The interplay goes on until one or more potential trust sequences are determined, that is, whenever both parties determine one or more sets of policies that can be satisfied for all the involved resources. To maintain the progress of a negotiation and help detecting a potential trust sequence a tree structure is used.

Once the parties have agreed on a trust sequence, the credential exchange phase begins. Each party discloses its credentials, following the order given by the trust sequence, eventually retrieving those credentials that are not immediately available through credentials chains. Upon receiving a credential, the counterpart verifies the satisfaction of the associated policies, checks for revocation and validity dates, and authenticates the ownership (for credentials). The receiving party then replies with an acknowledgment, and asks for the subsequent credential in the sequence, if any. Otherwise, a credential belonging to the subsequent set of credentials in the trust sequence is sent. The process ends with the disclosure of the requested resource or, if any unforeseen event happens, an interruption.

A minimal credential disclosure approach

Users are often required to reveal more information than necessary when authorizing digital transactions. For example, they might be required to supply their birth date in order to prove that they are at least 18. If such information falls into the wrong hands, there is a potential for misuse.

Information such as birth date is often used as a secondary identifier by service providers and, therefore, revealing such information unnecessarily can increase the risk of identity fraud. The minimum disclosure credential concept assumes that users' personal information is stored as a set of "micro-claims". Examples of birth-date-related micro-claims are "Age >18", "Age >21", "Age >25", etc. Similar micro-claims can be constructed for almost any category of personal information. Once micro-claims are constructed, minimum disclosure credentials allow users to reveal, in a manner easily verifiable by receiving parties, the minimum set of micro-claims necessary for a particular transaction. This allows users to minimize the amount of their personal information that is revealed to, and maintained by, parties with which they interact.

Credential design

The core of a minimum disclosure credential is a Merkle hash tree structure, which is similar to a redactable signature (Johnson et al. 2002). A redactable signature scheme is such that the signature can be verified even when parts of the document being signed are hidden (redacted). In the credentials that use this structure, hashes of micro-claims are represented as leaves in a Merkle hash tree, as shown in Fig. 1. The credential consists of two parts: a *public part* and a *private part*. The public part of the credential is a certificate. The certificate holds information about the issuer, the certification chain for the issuer, the type of certificate, the date range over which the certificate is valid, the user's public key, and a signature of the root node of a Merkle hash tree. The certificate should not hold any data about the user directly,

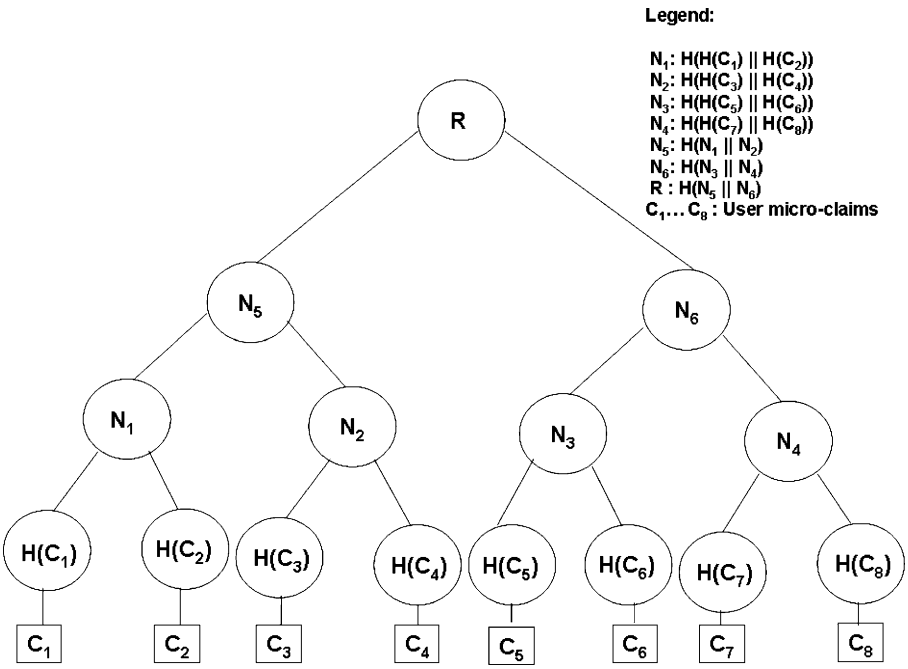

Fig. 1 Simple Merkle hash tree with eight claims

even data as common as a name. As per standard operation, the certificate will be signed by a certificate authority. The private part of the credential consists of a private key and a Merkle hash tree, whereby all of the leaf nodes are attributes of, or micro-claims about, the identity of the user, who is the credential holder. A user who wishes to assert some micro-claims supplies the certificate, the values of the asserted claims and their positions in the hash tree, and additional hash tree intermediate values necessary to reconstruct the root hash value, which the receiving party validates against the root hash value in the certificate.

The basic minimum disclosure credential scheme has been extended to allow claims from multiple authorities to be combined within a single credential, while still allowing a user to supply an arbitrary subset of the claims from all authorities (Bauer et al. 2008). This allows users to have multiple "identity providers" that supply claims only related to their particular domain.

In order to support credentials that contain claims issued by different identity providers, referred to as *combined credentials*, the hash tree structure contains a subtree for each identity provider. The root of the subtree has a third child node that represents the hash of the public certificate for the nodes in the subtree. We use combined credentials in our trust negotiation approach with minimal disclosure.

Protocols for the credential

In order to create a credential by a single identity provider, four steps are required. First, the user and the identity provider agree on a list of claims. Next, the hash tree for the claims is generated. Third, the identity provider verifies that the user possesses the private key. Finally, the identity provider generates and signs the public certificate. When generating the hash tree, random padding must be added to the claims before they are hashed, as discussed in (Bauer et al. 2008). With a single identity provider, the tree will always be balanced, bounding the number of interior nodes to the number of claims. The number of hashes needed to generate the tree is therefore bounded to twice the number of claims-generally an insignificant amount of computation time.

The protocol for using the credential follows conventional PKI certificate usage. To show a set of claims from the credential, the owner provides the claims and the intermediate node values and path information necessary for the receiving party to verify each of the claims. If any of the claims are in a subtree certified by a different identity provider, the accompanying certificates must be included with the set of claims. The receiving party verifies that the claims are in the hash tree specified (via the root hash) in the certificate part of the credential. The receiving party also verifies the signature on the certificate part of the credential. In order to verify that the sending party is the holder of the credential, the receiving party also verifies that the sending party possesses the private key which matches the public key claimed by the credential. This can be done by standard methods, such as challenge/response or as part of a secret key agreement operation, as long as the key verification is tied to the specific claims being asserted by the sending party. In addition, to cryptographic verifications, the receiving party must confirm that it trusts the identity providers to assert the claims in the credential. For example, an assertion by the Department of Motor Vehicles that an individual is a licensed lawyer should not be trusted.

Trust negotiation with minimal credential disclosure

None of the existing trust negotiation software systems provide support to minimal integration disclosure. Therefore, in order to integrate such techniques into trust negotiations, it is necessary to extend the language to specify negotiation policies and the negotiation protocol itself. If we consider Trust-X, the negotiation language must be extended with the specification of disclosure policies for identity attributes. At the negotiation protocol level challenging issues are how to match the attribute names in the disclosure policies' terms with the attribute names in the micro-claims and how to select the micro-claims that make true the terms. The negotiating parties might use different vocabularies and hence the terms in the policies and the micro-claims might have syntactic and/or semantic differences in attribute names and categorical values.

Identity attribute disclosure policies

Trust negotiations are carried out on the basis of disclosure policies that regulate the release of sensitive credentials, while the approach for minimal credential disclosure is based on micro-claims that express conditions on user's identity attributes not related to a particular credential type. As such, the *X -TNL* language must be extended in order to specify disclosure policies that protect the release of attributes rather than credentials. We thus extend the disclosure policy language with expressions of the form $A \leftarrow AT_1, \ldots, AT_n$ where A is an attribute name, and terms AT_i are attribute conditions of the form *a op expr*. *a* denotes an attribute name, *op* is a comparison operator, such as $=$, $\leq$, $\geq$, $<$, $>$, and *expr* denotes a constant or a variable name.

Micro-claims matching

Users may belong to different domains, and each domain may use a different vocabulary to denote identity attribute names. Thus, when a user sends its disclosure policies to the counterpart during a trust negotiation, the counterpart may not be able to understand which micro-claims he/she has to provide to satisfy the user's policy. Therefore, we need a technique for matching the identity attribute names in disclosure policies with the names in the users' micro-claims.

The matching technique depends on the types of variations in identity attribute names, which can be classified as follows:

- *Syntactic variations* refer to the use of different character combinations to denote the same term. An example is the use of "CreditCard" and "Credit_Card".
- *Terminological variations* refer to the use of different terms to denote the same concept. An example of terminological variation is the use of the synonyms "Credit Card" and "Charge Card".
- *Semantic variations* are related to the use of different concepts in different domains, each characterized by a different ontology, for denoting the same term.

Syntactic variations can be identified by using look-up tables enumerating the possible ways in which the same term can be written by using different character

combinations. Terminological variations can be identified by means of dictionaries or thesaurus such as WordNet (Fellbaum 1998). Dictionaries are typically used to retrieve all the synonyms of a given term. Semantic variations can be determined by using ontology matching techniques (Choi et al. 2006; Jarvis 2003). Ontology matching supports the comparison of not only the name of an attribute but also its semantics by considering the relations defined in the ontology with which the attribute is associated.

To detect all the possible variations in identity attribute names listed in disclosure policies and in micro-claims, we adopt a matching technique based on look up tables, dictionaries, and ontology mapping. To enable such matching approach, we assume that users have an ontology describing their domain and the identity attributes specified in their disclosure policies and micro-claims correspond to a concept in such ontology. Moreover, each user maintains a look up table containing alternative character combinations and a table recording sets of synonyms for (some of) the identity attribute names referred in his/her disclosure policies and micro-claims.

The matching protocol takes place when a user receives a disclosure policy from the counterpart and notifies the counterpart that is not able to understand which identity attributes he/she has to provide to satisfy the policy. Thus, the counterpart sends a set of alternative character combinations and a set of synonyms for the identity attributes listed in its disclosure policy. The party thus presents the user with predefined naming alternatives. If by using the counterpart's information, the user is not able to match its identity attributes with the ones by the counterpart, it notifies the counterpart. Thus, the counterpart tries to match the concepts corresponding to the identity attributes the user is not able to provide with concepts of the user's ontology. If no matches are found, the negotiation fails.

Micro-claims selection strategies

Once the match between attribute names in the disclosure policies' terms and the ones in the micro-claims is performed, it is important to select the micro-claims that satisfy the terms. Since there might be multiple micro-claims for the same attribute name, different strategies can be adopted. A naive strategy is to use string matching to select the micro-claims that satisfy a policy term. This strategy is very restrictive because if there are no micro-claims that textually match a policy term, the term cannot be satisfied even though there are micro-claims that logically match or imply the condition specified in the term. For example, assume that the birthday of a user is 04–15-1978 and hence the user is 30 years old. Suppose that the user has to prove that his age is greater than 18, that is, he has to prove the disclosure policy term "Age >18". Suppose that the following micro-claims are recorded in his Merkle hash credential: "Age >16", "Age >20", "Age >28", and "Age <45". It is clear that under the string matching strategy, no one of those micro-claims matches the policy term, even though both the micro-claims "Age >20" and "Age >28" logically imply the condition "Age >18".

To address such issue, the most intuitive strategy is the one that we refer to as *tight bind strategy*. Under such strategy, the selected micro-claim is the one that logically implies the term in the disclosure policy and that is closer to the actual value assumed by the attribute from which the micro-claim is derived. Though this strategy is intuitive, it makes it possible for the service provider (or any party

receiving the micro-claim) to obtain a tighter bound on the actual value of the attribute. In the scenario described above, where a user has to prove to be older than 18, the tight bind strategy would select the micro-claim "Age >28". If a malicious party knows that tight bind strategy is adopted, it can reduce the uncertainty about the actual age of the user. An alternative strategy, that we adopt, is to select the micro-claim which, not only logically implies the policy term, but also has the farthest value from the actual value of the attribute. Such strategy, that we refer to as *loose bind strategy*, provides a less tight bound about the actual value of the attributes and thus provides a higher uncertainty about the actual values of the attribute, as compared to the tight bind strategy. According to this strategy, if a user has to prove to be older than 18 in the scenario described above, the loose bind strategy will select the micro-claim "Age >20".

Note that both the tight bind and the loose bind strategy can also be adopted to select micro-claims about identity attributes that assume non numerical values. However, in order to apply such strategies, the attributes must be organized into some hierarchical form. Organizing the data into hierarchies allows for generalization. Generalization is accomplished by assigning a disclosed value that is more general than the original attribute value. For example, a user who has to prove he is resident in Indiana can make the address information less specific by omitting the street and city and revealing just the zip code. We assume that domain D^i is a partially ordered set $(D^i, \angle)$ whose largest element corresponds to the non-generalized attribute value and the smallest element is the most generalized value which is the suppressed value. The domain D^i contains multiple generalization levels denoted by the pedex. An attribute generalized to $D_1{}^i$ is more general than an attribute generalized to $D_2{}^i \angle D_1{}^i$, which implies that $D_2{}^i$ discloses more information than $D_1{}^i$. Using this basic structure, the user can decide which strategy to apply, by selecting the adequate level of generalization. For example, a user who has to prove he/she is resident in Indiana, if he/she decides to apply the loose bound strategy, would select a micro-claim about the zip code, while he/she discloses all micro-claims about its address if adopting the tight bind strategy.

Users can decide to use both the strategies alternatively during a negotiation process. Using always the same strategy for the selection of micro-claims may lead to information leakage because the strategy can be identified. Once the strategy is identified, the attacker may infer valuable information about the identity attributes being hidden.

For example, consider the case of a user who needs to prove he is older than 18. If the user decides to adopt tight-bind strategy, and he is 20, he will select Age >18 micro-claim, rather than the micro-claim Age <28. By looking at previous negotiation steps, and possibly correlating micro-claims that are semantically related, the attacker may guess that the user is adopting a tight-bind strategy. Hence, he can guess that the user is likely closer to 18 than 28, and most likely younger than 25.

If the attacker does not know the criteria according to which the predicates are selected, even these simple guesses are harder to make. By alternating different strategies, the user's strategy selection is hidden, and therefore it's harder for an attacker to infer the actual value of identity attributes.

Negotiation protocol with minimal credential disclosure

The adoption of the minimal credential disclosure in Trust-*X* requires extending the profiles of parties to store Merkle hash tree credentials.

We thus extend users *X-Profiles* to include the credentials according to the Merkle hash tree structure, with micro-claims about users identity attributes and all the credentials that certify that the users have these identity attributes. At the beginning of the negotiation users select the strategy to select the micro-claims (loose bind strategy or tight bind strategy) . For each term AT_i in the disclosure policy $A \leftarrow AT_1, ..., AT_n$ received by the counterpart, a party computes the set *Claim_Set* of micro-claims name in AT_i and that make true AT_i. The match between attributes names in the AT_i terms and the attributes in the micro-claims is executed by using the approach described in the previous section. If *Claim_Set* is empty, the policy cannot be satisfied by the party, and a notification message is sent to the counterpart. Otherwise, the party checks if there are policies protecting the release of the attribute corresponding to term AT_i. If such policies exist, the party sends them to the counterpart. At the end of the policy evaluation phase, when the two parties agree on a trust sequence *Trust_Seq* of terms that must be satisfied, they do not exchange and verify one by one the credentials, as it typically occurs in Trust-*X* negotiations. For each term AT_i in the *Trust_Seq,* the user has to satisfy, the Merkle hash tree credentials containing the micro-claims that match the attribute names in AT_i and imply AT_i are selected. Once selected the micro-claims that match and satisfy the terms in counterpart's disclosure policies, the Merkle hash tree credentials containing such claims are combined together. Thus, only the public part of the combined credential, and the information to enable the verification of counterpart disclosure policies are sent to the counterpart. This information includes the list of micro-claims that match the terms in counterpart disclosure policies, the intermediate node values and path information necessary for the receiving party to verify all the micro-claims. The receiving party verifies that the micro-claims are in the hash tree by computing the hash value of the root and checking if the root hash in the certificate matches the computed hash value. The receiving party also verifies the signature on the root value in the certificate part of the credential. In order to verify that the counterpart is the holder of the credential, the party also verifies that the counterpart owns the private key which matches the public key claimed by the credential. If the counterpart micro-claims verification is successful, the algorithm checks if the signature affixed on the root hash in certificate is valid. If the signature is valid, the negotiation between the two parties is successful, otherwise the negotiation fails.

EXAMPLE 2 Consider a simple negotiation where a party receives the disclosure policy DrivingLicense←State = Indiana, Age >18 that requires the party to prove that he is resident in Indiana and that he is older than 18. The party has to demonstrate that he owns a set of micro-claims that satisfy the terms State = Indiana and Age >18. Assume that the party owns the Merkle hash tree credential in Fig. 2. The credential contains the micro-claims State = Indiana, ZipCode = 47906, City = West Lafayette, Address = Brown Street, and Age >16, Age >20, Age >28, and Age <45. Assume that the identity attributes about the location of the user are ordered based on their level of generalization as follows: ZipCode = 47906 < Address =

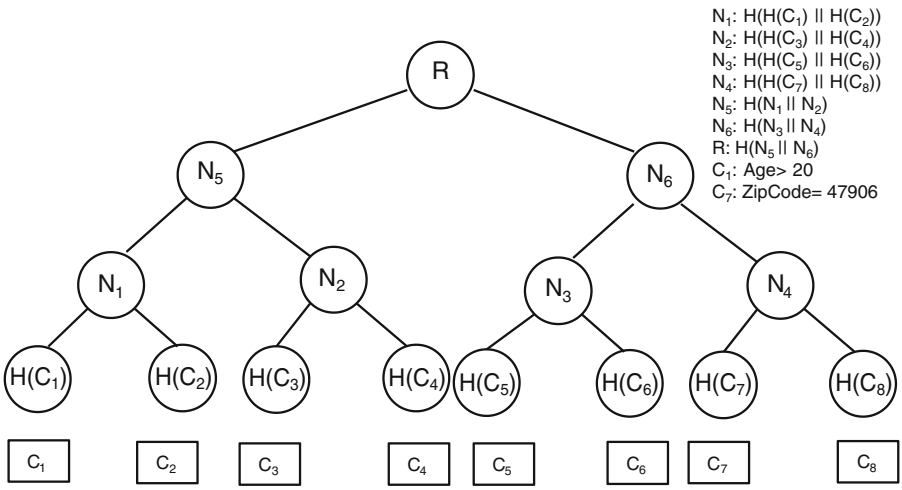

Fig. 2 Example of Merkle hash tree

Brown Street < City = West Lafayette < State = Indiana. Assume that the party decides to adopt the loose bind strategy. Thus, according to loose bind strategy the micro-claims that match the terms State = Indiana and Age >18 in the disclosure policy are ZipCode = 47906 and Age >20 respectively.

The party, sends the certificate and the following information to prove his claims:

- *for the claim Stateof Residence = Indiana, the authentication path is H (C_7), H (C_8), N_3 , N_4, N_5, N_6 , R and the node set is{H (C_7), H (C_8), N_3, N_5}*
- *for the claim Age >25, the authentication path is H(C_1),H(C_2), N_1, N_2, N_5, N_6, R, and the node set is {H(C_1),H(C_2), N_2, N_6}.*

The counterparty computes the root hash performing the following steps:

1. *computes the hash of H(C_1) || H(C_2) that is N_1*
2. *computes the hash of N_1 || N_2 that is N_5*
3. *computes the root value R as the hash value of N_5 || N_6*

If the value computed matches the value in the certificate sent by party, the verification succeeds.

Implementation and experimental evaluation

In this section we present, first, the performance of the minimal credential disclosure approach and then we present the performance of the Trust-*X* negotiation protocol with minimal credential disclosure.

Table 1 Comparison of the public-key operations for Merkle hash tree, Brute force and Brands credential schemes for 1 micro-claim to be verified

	Operations
MHT Credential	
Prover	Tree reduction
	Prove ownership (1 slow modular exponentiation)
Verifier	Hash computations
	Verify certificate (1 fast modular exponentiation)
	Verify ownership (1 fast modular exponentiation)
Brute Force	
Prover	Prove ownership (1 slow modular exponentiation)
Verifier	Verify certificates (1 fast modular exponentiation)
	Verify ownership (1 fast modular exponentiation)
Brand's Credential	
Prover	Prove credential + ownership (1 slow modular exponentiation)
Verifier	Verify credential + ownership (1 slow modular exponentiation)

Performance of micro-claims' verification based on Merkle hash tree credentials

Our minimal disclosure credential design uses mainly one-way hashes and minimizes the number of public/private key operations (large modular exponentiations). This allows our system to be faster than the digital credential schemes of (Brands et al. 2007) and (Camenisch and Lysyanskaya 2001), which require more of these expensive, modular exponentiation operations. Table 1 reports a summary of the public/key operations required by the Merkle hash tree based credential scheme and the one by Brands.

We performed two sets of performance experiments dealing with the credentials alone, focusing on the computational burden placed on servers, which is likely to be the performance bottleneck in a system. In the first set, we evaluated credential verification time vs. number of claims. In the second set, we implemented and deployed an actual client-server system in Emulab (White et al. 2002), and evaluated server throughput. These experiments included communication costs and additional client-side and server-side computations costs in addition to credential verification time, and have multiple clients accessing the service simultaneously, to provide a realistic assessment of performance. In all experiments, we compared our hash tree approach against Brands' credentials and the brute force solution of embedding each claim in a separate certificate. For Brands' credentials, we used the parameters from (Brands et al. 2007) ($|p|=1{,}600$, $|q|=256$, $|s|=160$), which is roughly equivalent in security level to our credential. Brands' credential has the advantage over our approach that it can prove properties of claims, reducing its need for micro-claims. At this stage, we do not attempt to quantify that advantage in these tests.

The first set of experiments was performed on an Intel Core 2 Duo E6600 running at 2.4 GHz using Sun's server JVM version 1.6.0. In all cases the operations were

Table 2 Time efficiency of hash tree and certificate

Operation	Time (μ-sec)
Verify Tree (SHA-256)	
1 claim of 2,048	31
20 claims of 2,048	100
All 2,048 claims	8,000
Verify Certificate (RSA, 1,536 bit)	
1 certificate	620
20 certificates	12,000
2,048 certificates	1,300,000
Verify Brands	
1 claim	38,000
20 claims	280,000
2,048 claims	26,000,000

run repeatedly before timing, so as to force the JVM to fully optimize the operations. We used X.509 certificates for the public key certificate part of the credential. The certificate was signed using RSA, and the CA's key was 1,536 bits. Results are shown in Table 2.

The total time to verify one hash tree credential is equal to the tree verification time plus the time to verify one certificate. The hash tree used in these experiments contains 2,048 claims. From Table 2, we see that verifying one claim with the hash tree approach is approximately 30 times faster than verifying one claim in a Brands' credential (2*620 μs+31 μs compared with 38,000 μs), including proving ownership. For 20 claims, the hash tree approach is approximately 210 times faster than Brands', and with 2,048 claims, the speedup factor is approximately 2,800. Speedup compared to the brute force approach of one certificate per claim approaches linear with the number of claims, e.g. approximately 50 times faster with 100 claims.

The hash tree credential also requires less computation for the prover than Brands' credential. Brands' credential requires almost the same time for the prover (client) as the verifier, for a modest number of claims. More precisely, the client requires about 26 ms to show a credential with one claim and about 270 ms to show a credential with 20 claims. Our credential requires one RSA private key operation (about 20 ms) and a tree reduction (about 5 ms, for showing 20 claims out of 2,048) on the part of the client.

In the Emulab experiments, there was a variable number of clients and one server, which accepted connections from the clients and verified one credential per client request. Each client and server instance was run on a separate 3 GHz Pentium Xeon. Each client generated verification requests one after another, waiting for one response and then immediately generating a new request. All parameters were the same as in the first set of experiments. Here, our primary performance measure is the server throughput (number of verifications done per second). The results are shown in Fig. 3.

The figure shows that throughput with Brands' credentials saturates at only a few clients, while the hash tree credentials can handle up to about 15 clients before saturating. The peak throughput for the hash tree credentials is approximately 210 verifications per second, while the peak throughput with Brands' credential having ten claims is approximately five operations per second and with 20 claims it is only about two

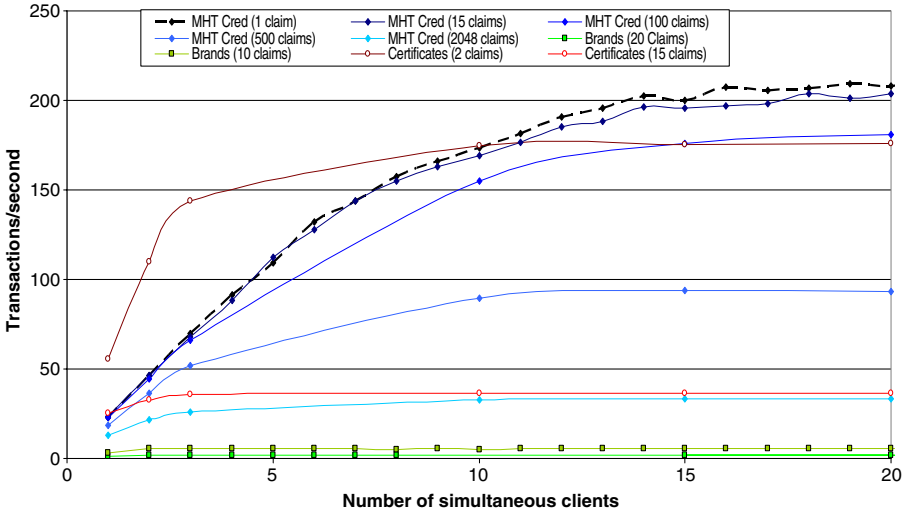

Fig. 3 Credential verification throughput vs. number of clients

operations per second. Adding claims to the hash tree credential has only a modest impact on performance. With 15 claims, the peak throughput is still over 200 verifications per second and even with 2,048 claims, the throughput is still around 33 verifications per second. The figure also shows that throughput for the brute force approach with one certificate per claim drops off rapidly as the number of claims increases.

Performance of the negotiation protocol with minimal credential disclosure

To evaluate the impact of adopting a minimal credential approach in trust negotiation, we have integrated into the Trust-*X* prototype (Squicciarini et al. 2007) the minimal credential disclosure approach by Bauer et al.

We have measured the time to perform credential exchange phase according to the complexity of the disclosure policies. By complexity of a policy, we mean the number of terms that are required to the counterpart for a single resource. We distinguish between single and multiple-term policies. Policy complexity influences the number and the length of alternative trust sequences according to which the two parties can complete the negotiation. We measure the total execution time of credential exchange phase for both negotiation parties, denoted as client and server respectively, and the execution type to combine the credentials to be sent to the counterpart and the time to verify the combined credentials in the following cases:

1. varying the number of single-term policies from five to fifteen scaling of a factor equal to five
2. varying the number of credentials conditions in a multi-term policy from five to fifteen scaling of a factor equal to five

We have performed our experiments on a Pentium 4 PC with 2.00-GHz processor and with 512 MB of RAM, under Microsoft Windows XP. The same working load of the system was ensured in all the experiments. In addition, for each test case,

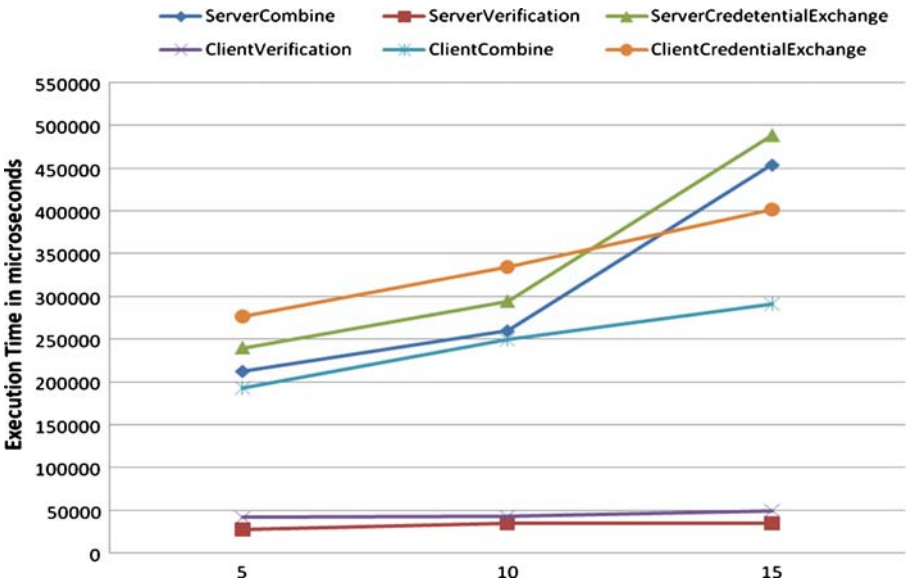

Fig. 4 Credential exchange execution time for single term policies

eleven trials have been executed, and the average of the results obtained from the last ten trials has been computed, excluding the results of the first test. The performance has been measured in terms of CPU time (in microseconds).

Figure 4 shows the experimental results for test case one. The total execution time of both server (represented in green) and client (represented in orange) increases with the increase of the number of single term policies exchanged during the

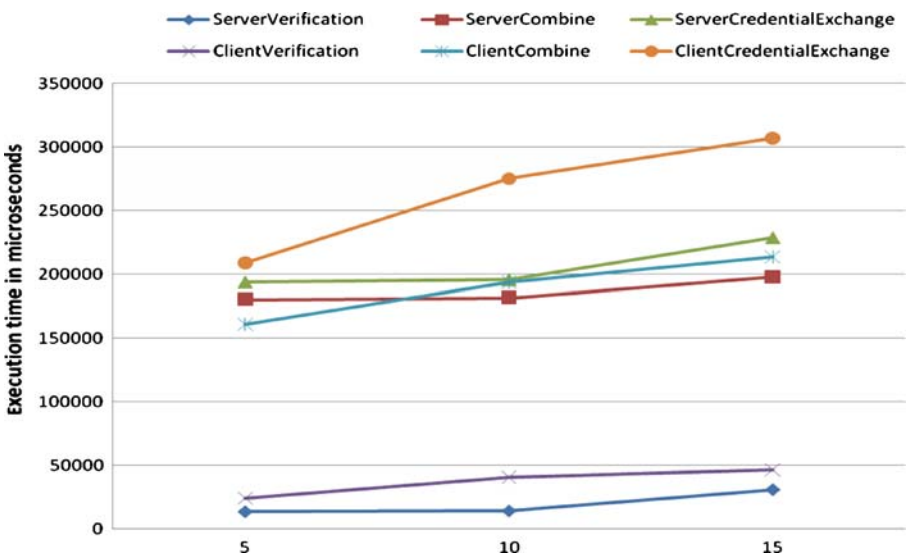

Fig. 5 Credential exchange execution time for multi term policies

negotiation. This result is due to the fact that the number of credentials to be combined and sent to the counterpart increases with the number of policies. However, the time to verify the combined credentials is almost constant.

Figure 5 shows the experimental results for the second test case. The execution time for credential exchange increases with the number of attribute conditions that needs to be verified by the two parties because the time to generate combined credentials increases.

Related work

Trust negotiation has been recognized as an interesting and challenging research area to explore, and has been extensively investigated in recent years. As a result, several techniques and prototypes have been developed (Herzberg et al. 2000; Winsborough et al. 2000). Work related to privacy in the context of trust negotiation systems has focused on the protection of sensitive policies and credentials. Yu et al. (2003) have developed a unified scheme, known as Unipro, to model resource protection, which applies to both the actual resources to be protected and to the policies. Such scheme, however, is based on a notion of protection which is closer to the notion of access control and as such it is not able to support anonymization. Our work, instead, has the goal of providing stronger privacy to individuals through the use of minimal credential disclosure techniques.

Two significant approaches dealing with selective disclosure of attributes are by Holt et al. (2003) and by Li et al. (2003). Holt's work focuses on hidden credential features, and on how to improve performance of hidden credentials constructed from identity-based crypto systems which satisfy credential indistinguishability. The notion of credential indistinguishability, adopted in such an approach, is very different from ours. The key idea by Holt et al. is to make credentials indistinguishable to any recipient which does not possess either of the credentials corresponding to P and P0, where P and P0 are elements of the set of possible single-credential policies. Li et al. have proposed a scheme called Oblivious Signature Based Envelope (OSBE) (Li et al. 2003). OSBEs are similar to Hidden Credentials in that the ability to read a message is contingent on having been issued the required secret.

Several privacy-enabled identity management systems have been based on the notion of anonymous credential (Camenisch and Herreweghen 2002; Chaum 1985). In anonymous credential systems, organizations know the users only by pseudonyms. Different pseudonyms of the same user cannot be linked. Yet, an organization can issue a credential to a pseudonym, and the corresponding user can prove possession of this credential to another organization (who knows him by a different pseudonym), without revealing anything more than the fact that she owns such a credential.

Stefan Brands developed a different form of digital credential, in which a user has a single public credential, but that credential is pseudo-anonymous, even to the issuer (Brands 2000). The credential holds attributes that the user can selectively prove to a service provider. Repeated showings of the same credential are linkable, both if shown to the same or different service providers. However, since the credential is issued blind by the identity provider, the effect is that a user has one

global pseudonym. The credential can be reissued easily, allowing the user to change the global pseudonym, as permitted by the identity provider.

We have shown that verifying minimum-disclosure credentials of the type we use herein is approximately three orders of magnitude faster than verifying Brands' credentials. The reason is that the minimum-disclosure credentials require mainly hash computations and only one RSA public-key operation to verify the signature on the certificate, whereas Brands' credential requires a large number of big modular-exponentiation operations.

Camenisch et al., have proposed and implemented yet another form of digital credential, or more precisely, yet more forms (Camenisch and Lysyanskaya 2001; Camenisch and Herreweghen 2002; Camenisch et al. 2005). IBM's Idemix system is based on Camenisch et al.'s work (Camenisch and Herreweghen 2002).

We emphasize that the approaches by Brands and Camenisch et al., achieve much stronger anonymity properties than are possible with our credentials. Minimum disclosure credentials provide auditability in that the certificate authority can tie the credential to a specific user. The lack of strong anonymity in the minimum disclosure approach has the additional benefit of much higher performance in verifying credentials.

Conclusions

In this paper we have presented an approach for the controlled release of identity attributes based on the integration of trust negotiation and minimal credential disclosure techniques. The minimal credential disclosure approach guarantees that during the negotiation process a party discloses only the attributes necessary to complete the interactions with other parties. As part of future work, we plan to investigate how to leverage micro-claims in the context of federations. A service provider, which has successfully verified a set of counterpart's micro-claims during a negotiation process, can release a signed assertion listing the counterpart's micro-claims and specifying how the micro-claims have been obtained and verified. Such an assertion can be used during subsequent negotiations in the federation. We also plan to investigate privacy-preserving strategies for the generation and selection of micro-claims. Micro-claims could be generated for example based on anonymization criteria, like k-anonymity.

Acknowledgements This material is based in part upon work supported by the National Science Foundation under the Grant CNS-CT-0716252ITR and upon work supported by the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001, under the auspices of the Institute for Information Infrastructure Protection (I3P) research program. The I3P is managed by Dartmouth College. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security, the I3P, or Dartmouth College.

References

Bauer D, Blough D, Cash D. Minimal information disclosure with efficiently verifiable credentials. Proceedings of 4th Workshop on Digital Identity Management, 2008.

Bertino E, Ferrari E, Squicciarini AC. Trust-X- a peer to peer framework for trust establisment. IEEE Transactions on Knowledge and Data Engineering (TKDE). TKDE. 2004;16(7):827–42.
Bishop M. Introduction to computer security. Boston: Addison-Wesley; 2004.
Brands S. Rethinking public key infrastructures and digital certificates; building in privacy. Cambridge Massachusetts: MIT Press; 2000.
Brands S, Demuynck L, Decker BD. A practical system for globally revoking the unlinkable pseudonyms of unknown users. In 12th Australasian Conference on Information Security and Privacy, 2007.
Camenisch J, Herreweghen EV. Design and implementation of the Idemix anonymous credential system. Proceedings of the 9th ACM Conference on Computer and Communications Security; 2002.
Camenisch J, Lysyanskaya A. An efficient system for non-transferable anonymous credentials with optional anonymity revocation, Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology; 2001.
Camenisch J, Hohenberger S, Lysyanskaya A. Compact e-cash. In: Cramer R, editor. Advances in cryptology—EUROCRYPT '05, Vol. 3494 of lecture notes in Computer Science; 2005. p. 302–21.
Chaum D. Security without identification: transaction systems to make big brother obsolete. Communications of the ACM, 28, 1985.
Choi N, Song IY, Han H. A survey on ontology mapping. SIGMOD Record. 2006;35(3):34–41.
Fellbaum C. Wordnet: an electronic lexical database. Cambridge: MIT; 1998.
Herzberg A, Mihaeli J, et al. Access control meets public key infrastructure, or: assigning roles to strangers. In Proceedings IEEE Symposium on Security and Privacy, Oakland, CA; 2000.
Holt J, Bradshaw R, Seamons KE, Orman H. Hidden credentials. In Proceedings of 2nd ACM Workshop on Privacy in the Electronic Society, Washington, DC; 2003.
Jarvis RD. Selective disclosure of credential content during Trust Negotiation, Master of Science Thesis, BrighamYoung University, Provo, Utah; 2003.
Johnson R, Molnar D, Song DX, Wagner D. Homomorphic signature schemes. Topics in Cryptology—CTRSA 2002, vol. 2271. Springer-Verlag; 2002, p. 244–62.
Li N, Du W, Boneh D. Oblivious signature-based envelope. In Proceedings of 22nd ACM Symposium on Principles of Distributed Computing (PODC 2003), Boston, Massachusetts; 2003.
Merkle R. A digital signature based on a conventional encryption function. In Proceedings of Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology; 1987, p. 369–78.
Squicciarini A, Bertino E, Ferrari E, Paci F, Thuraisingham B. PP-trust-X: a system for privacy preserving trust negotiations. ACM Transactions on Information Systems Security. TISSEC. 2007;10(3):1–50.
White B, Lepreau J, Stoller L, Ricci R, Guruprasad S, Newbold M, et al. An integrated experimental environment for distributed systems and networks. In Proc. of the Fifth Symposium on Operating Systems Design and Implementation, Boston, MA; 2002, p. 255–70.
Winsborough WH, Seamons KE, Jones V. Automated trust negotiation. DARPA Information Survivability Conference and Exposition, Volume I, Los Alamitos, CA, USA: IEEE Press; 2000, p. 88–102.
Yu T, Winslett M. A unified scheme for resource protection in automated trust negotiation. In Proceedings of IEEE Symposium on Security and Privacy, Oakland, CA; 2003.