CERIAS Tech Report 2004-73

PRIVACY-PRESERVING TRUST NEGOTIATION

by E.Bertino, E.Ferrari, A.Squicciarini

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

Privacy-preserving trust negotiations

E. Bertino[1], E. Ferrari[2], A. Squicciarini[3]
[1]CERIAS and Computer Science Department
Purdue University, West Lafayette, IN, USA
bertino@cerias.purdue.edu
[2]Dipartimento della Cultura, Politiche e Informazione
Universita' degli Studi dell'Insubria, Como
elena.ferrari@uninsubria.it
[3]Dipartimento di Informatica e Comunicazione
Universita' degli Studi di Milano
squicciarini@dico.unimi.it

Abstract

Trust negotiation is a promising approach for establishing trust in open systems, where sensitive interactions may often occur between entities with no prior knowledge of each other. Although several proposals today exist of systems for the management of trust negotiation none of them addresses in a comprehensive way the problem of privacy preservation. Privacy is today one of the major concerns of users exchanging information through the Web and thus we believe that trust negotiation systems must effectively address privacy issues to be widely acceptable. For these reasons, in this paper we investigate privacy in the context of trust negotiations. More precisely, we propose a set of privacy preserving features to be included in any trust negotiation system, such as the support for the P3P standard, as well as different formats to encode credentials.

1 Introduction

The huge recent increase in web-based applications carried out on the Internet has been accompanied by an exponential amount of data exchanged and collected by the interacting entities. As the amount of exchanged information exponentially grows, privacy [21] has emerged as one of the most crucial and challenging issues. Current researchers are thus focusing on devising both systems for supporting on line resource sharing [14, 22] and on technologies for preserving user privacy in a standardized and automated manner [1].
Privacy is defined as "the right of individuals to determine when, how and to what extent information about them is communicated to others".[1] The most significant proposal for supporting privacy over Internet is the Platform for Privacy Preferences - P3P [7]. The designers of P3P have also developed a preference language, called APPEL [8], to allow users to express their privacy preferences, thus enabling automated matching of privacy preferences against P3P policies.
Privacy issues are particularly crucial when dealing with trust management. Today, among the various approaches that can be adopted for exchanging resources and services on the web, a promising model is represented by trust negotiations [22]. All existing trust negotiation systems are based on the disclosure of certain amount of sensitive information, usually conveyed by *digital credentials*, required to establish trust. However, although several efficient and powerful negotiation systems

[1]This definition is by Alan Westin, Professor of Public Law and Government, Columbia University.

have been developed so far [5, 9, 18, 26, 23], none of them provides a comprehensive solution to protect privacy during the negotiation process. In particular, none of them supports P3P policies for expressing user privacy requirements and preferences in a standard way. Our belief is that trust negotiation has the potentiality for being widely used to establish trust during on-line negotiations, once it is complemented with privacy-preserving techniques. Starting from an analysis of the most relevant privacy pitfalls in trust negotiation, in this paper we revise all the key aspects of a trust negotiation, which are crucial in order to efficiently and effectively preserve privacy. Our philosophy is to exploit, whenever possible, existing standard privacy technologies, such as P3P and APPEL. The main innovative features we propose in this paper are the support for different credential formats, each of which provides a different degree of privacy protection, the notion of *context* associated with a policy, which allows one to both express privacy policies, and convey information which can be used to protect disclosure policies, and the integration of P3P policies at various steps of the negotiation. The work reported in this paper is built on top of a negotiation system named Trust-$\mathcal{X}$ previously proposed by us [3]. However, the extensions we propose in this paper are major extensions since our previous proposal, as the other existing negotiation systems, does not address privacy nor it does support any of the above mentioned features.
A work related to our is the work by Seamons et. al [18] which explores issues concerning support of sensitive policies, based on the use of of hierarchies in policy definitions. However, they do not address the issue of privacy policy support. The work by Winslett et. al, [26], provides a unified scheme to model resource protection, including policies. It currently represents one of the most significant proposals in the negotiation research area, and it is the approach that most influenced our work. However, [26] does not deal with the issue of supporting privacy policies, neither it defines an ad hoc policy language. Finally, [19] provides an overview of some of the privacy problems that may arise during a negotiation. However, it does not provide a comprehensive solution to such problems, rather it mainly deals with protection of sensitive policies.
The remainder of this paper is organized as follows. Next section presents the main privacy pitfalls compromising trust negotiations, and outlines possible remedies. Section 3 overviews the Trust-$\mathcal{X}$ framework. Section 4 presents the Trust-$\mathcal{X}$ privacy language. Section 5 deals with a privacy based approach to trust negotiation. Section 6 presents Trust-$\mathcal{X}$ architecture, whereas Section 7 concludes the paper and outlines future research directions.

2 Privacy pitfalls and solutions in trust negotiations

One of the major concerns users have in adopting negotiation systems is that trust negotiation does not control or safeguard personal information once it has been disclosed. Nothing is usually specified about the use of the information disclosed during a negotiation. A possible solution to this problem is to integrate the negotiation system with the P3P platform [7]. The P3P platform can be used for stating how the personal information collected through credentials disclosure during on line transactions will be managed by the receiver. Another potential vulnerability of trust negotiation arises because of the common strategy of postponing actual credential disclosure. Indeed, during the policy evaluation phase, privacy can be compromised in several ways, since there are no guarantees about counterpart honesty until the end of the process. Policy disclosure can be used to determine the value of sensitive attributes without the credential ever being disclosed. Furthermore, during policy exchange it is not possible to determine whether a party is lying or not until the credentials are actually disclosed. Indeed, when a request for a sensitive credential is sent the counterpart typically replies by sending a counter request for the credentials necessary to disclose the credential originally requested. Thus, the receiver can infer that the counterpart can satisfy the request, obtaining clues about the possession of sensitive credentials, even if it never actually obtains the credential. As a result, a mendacious subject may practically gather the counterpart

profile by falsely declaring possession of credentials. An ideal system should be able to prevent information leakage, without interfering with the negotiation process. A possible solution, proposed by Winsborough et al. in [23], is that of introducing the notion of attribute acknowledgment policies, in which a participant establishes for those attributes that she considers sensitive, whether or not she satisfies those attributes. However, the negotiation process results quite cumbersome and requires a user to specify an amount of policies, larger than her real necessity. An alternative solution is to disclose credentials as soon as a corresponding policy has been satisfied before the end of the policy evaluation phase. However, this strategy may result in unnecessary credential disclosures, as well as needless rounds of negotiation when failure is inevitable. A more promising approach is thus that of adopting a different perspective. Rather than introducing a policy for each sensitive attribute or jeopardizing private information by immediately disclosing credentials, policy expressiveness may be improved, by giving the user the possibility of specifying key information for driving the negotiation, while sending the policy. Finally, another issue related with credentials arises because of sensitive attributes.(e.g., age, credit rating). A credential may contain several sensitive attributes, and very often just a subset of them is required to satisfy a counterpart policy. However, when a credential is exchanged, the receiver anyway gathers all the information contained in the credential. Although some proposals exist [6, 12] for encoding digital credentials, no widely accepted standard exists for their representation which allows the possibility of a partial disclosure.

3 Trust-$\mathcal{X}$ overview

This section briefly summarizes the main features of the Trust-$\mathcal{X}$ [3] system which is used as the reference system throughout the paper. Trust-$\mathcal{X}$ is a comprehensive framework for trust negotiation, providing both a language for encoding policies and certificates, and a system architecture. The language, named $\mathcal{X}$-TNL, is XML based language and is specifically conceived for the specification of both certificates and policies. Trust-$\mathcal{X}$ certificates convey information about the parties and can be either credentials or declarations. Digital credentials are assertions describing one or more properties of a given subject, certified by trusted third parties. Declarations, by contrast, are self-credentials issued by their owner, collecting auxiliary information that do not need to be certified (such as for instance specific preferences) but may help in better customizing the negotiation. Since declarations are complementary information to be optionally exchanged and do not present particular privacy requirements, in what follows we focus only on credentials. Such certificates are collected into $\mathcal{X}$-Profiles, which are associated with each Trust-$\mathcal{X}$ entity.
Protection needs for the release of a resource are expressed by *disclosure policies*. A resource can be either a service, a credential, or any kind of data that need to be protected. Disclosure policies regulate the disclosure of a resource by imposing conditions on the credentials the requesting party should possess. Disclosure policies for a resource can be gradually made known to the counterpart according to the degree of trust established, in order to ensure a better protection of the sensitive information exchanged. Trust-$\mathcal{X}$ also comprises an architecture for negotiation management, which is symmetric and peer-to-peer. A Trust-$\mathcal{X}$ negotiation consists of a set of phases to be sequentially executed. The idea is thus to disclose policies at first, in order to limit credential release, and then disclose only those credentials that are necessary for the success of negotiation. The key phase of a Trust-$\mathcal{X}$ negotiation is the policy evaluation phase, which consists of a bilateral and ordered policy exchange. The goal is to determine a sequence of credentials, called *trust sequence*, satisfying disclosure policies of both parties. More precisely, each time a disclosure policy is received the steps to be executed and the corresponding Trust-$\mathcal{X}$ modules involved are the followings:

- The party determines if the policy can be satisfied by any of the possessed credentials, querying the $\mathcal{X}$-Profile.

- The *compliance checker* checks in the policy base the protection needs associated with the credentials, if any.
- If a set of credentials and associated policies are actually found, the set of counter policies are extracted by the *policy base* and then sent to the counterpart.

Once a trust sequence has been determined, the credential exchange phase is executed. Each time a credential is received, the local compliance checker module checks local policy satisfaction and verifies at runtime the validity and ownership of the remote credentials. Functions required to carry out credential disclosure include verification of credential contents, checking for revocation, authentication of ownership. More details on the Trust-$\mathcal{X}$ system can be found in [3].

4 A privacy preserving specification language

In what follows we present the solutions we have devised to protect privacy during negotiations by proposing some extensions to the conventional languages for expressing policies and credentials. More precisely, we first propose an extension of the standard credential format to deal with privacy issues. Then, we introduce the notion of *context* associated with a policy, which can be used to attach privacy rules to a policy, as well as to protect policy disclosure and to speed up the negotiation process.

4.1 Private credentials

During trust negotiations credentials play a key role, in that they represent the means to prove parties properties required to establish trust. Credentials must thus be unforgeable and verifiable. Typically, a digital credential contains a set of attributes specified using name/value pairs. The credential is signed using the issuer's private key and can be verified using the issuer's public key. Our system supports two types of credential schemes, each reflecting different user requirements, such as the possibility of gradually establishing trust, preventing party lies, or the maximization of privacy protection of the exchanged attributes. The first one is called *basic format* and represents the standard approach for credential encoding, that is, a digitally signed document containing a set of subject properties. The second proposal is called *privacy enhanced* format, and is based on a credential template able to allow disclosure of the credential in two different steps, keeping the sensitive content of the credential secret until the end of the negotiation. The proposed format also supports partial disclosure of credentials, to protect the privacy of sensitive attributes.
In the next sections we first illustrate the technique used to support partial attribute disclosure. Then, we present the privacy enhanced format, since this represents a novelty in the state of the art. We do not further elaborate on the basic format, since it has been already presented in [4].

4.1.1 Protected attribute credentials

An interesting approach to maximize privacy protection is to selectively disclose attributes within a credential, so that only the needed subset of properties is made available to the recipient of the credential. The best system currently available to allow partial disclosure of credentials relies on the use of the bit commitment technique [15], which enables users to commit a value without revealing it. By exploiting this technique within digital credentials it is possible to actually send credentials by revealing only the minimal set of attributes required during the negotiation. Although the idea of selectively disclosing credential attributes is not new [6, 16], this technique has never been thoroughly explored, especially in trust negotiations. The only work on this topic is from Jarvis [10]. The work focuses on selective disclosure of credentials during negotiations and provides a prototype

```
<HEADER>
<Student_Card credID='12ab', CredType= Student Card >
<Issuer HREF='http://www.ItalyCountry.com'
Title=KTHUniversity_Repository/>
<expiration_date> may 12th 2002   </expiration_date>
</HEADER>
<CONTENT>
    <name>
    <Fname> Olivia </Fname>
      XXXXXXXXXXXXXXXXX
    </name>
XXXXXXXXXXXXXXXXXXXXXX
    <faculty> history </faculty>
    <badge_number> 328454</badge_number>
</CONTENT>
</Student_Card>
```

Figure 1: Example of protected attributes in a privacy enhanced credential

implementation. Our focus, differently from [10], is to deeply analyze the impact of protected attribute credentials on trust negotiations, and devise new strategies to allow interoperability between users adopting various credential formats. Further, instead of using the bit-commitment technique we adopt a multi-bit hash commitment technique for attribute encoding, as the length of attributes will likely be longer than one bit. The general protocol followed to issue credentials with protected attributes is briefly summarized in what follows. A credential requester first generates the set of attribute values for the credential. In order to create a credential with protected attributes the requester has first to create the corresponding private values to be used in place of the sensitive ones. Given a sensitive attribute `a` with value `va` the operations needed for its protection are: 1) generate a random string `r`; 2) compute `p=va|r`, that is, the concatenation of `va` with `r`; 3) compute `v=one_way(p)`, generated by invoking a hash function on `p`. These operations are performed for all the attributes of the credential that need to be selectively protected. Once the credential is ready, it is submitted to the credential authority, which verifies its content and the corresponding values, and finally signs it. During a negotiation, the credential can be sent by keeping secret the content of protected attributes. The disclosure of private attributes is executed by sending the counterpart both `va`, the original value, and `r`, the random value, so that the receiver can compute `va` using the same hash function and verify the attribute validity. The remaining sensitive attributes of a credential that are not relevant for the negotiation be left hidden, and never be disclosed to the counterpart.

4.1.2 Privacy enhanced credentials

The basic Trust-$\mathcal{X}$ credential is an XML document, digitally signed by the issuer, according to the standard defined by W3C for XML Signatures [27]. A credential is an instance of a *credential type*, which is a DTD (Document Type Definition) used as a template for credentials having a similar structure. Although the content of a credential is determined by the corresponding type, each credential, beyond the specific language used for its encoding, must always convey some general reference information about the corresponding credential type, the issuer, and its temporal validity. This set of information are crucial for proving that the credential, besides its specific content, is a signed and valid digital document issued by an entity reputed trusted.
The credential format we have devised, named *privacy enhanced credential template*, captures this reference information into a specific portion of the document, named *header*, which is kept separated from the private content of the credential, contained into a different portion of the document, referred to as the *content*. Further, credential content is structured by using the technique presented in the previous section, so that partial disclosure of attributes can also be achieved. This way of

structuring credentials enables negotiating parties to adopt new strategies to gradually establish trust. Indeed, the header and content can be disclosed at different times during a negotiation. For instance, one possible strategy is to disclose the header to prove credential possession as the credential is involved into a negotiation, and keep the credential content secret until its disclosure at the end of the whole process. An alternative approach can be that of requiring attributes to be disclosed as soon as they are requested by a policy. Then, header disclosure can be immediately followed by disclosure of the required attributes and corresponding random values.
Although Trust-$\mathcal{X}$ provides an XML-based encoding of credentials, a privacy enhanced credential template is a language independent way of encoding credentials. Thus, in what follows we give a logic definition, abstracting from the specific Trust-$\mathcal{X}$ syntax. Formally, a credential type ct can be represented as a pair $< n_{ct}, p_{ct} >$, where n_{ct} is the name of the credential type, and p_{ct} is the set of corresponding attribute specifications. Each attribute specification contains the name and the domain. A privacy enhanced credential template can be modeled as follows.

Definition 4.1 (**Privacy enhanced credential template**). Let $ct =< n_{ct}, p_{ct} >$ be a credential type, a *privacy enhanced credential* template for ct is a pair $<$ *header-template*, *content-template* $>$ where:

- *header-template* is a set containing the following attribute specifications:
  1) `credID`, specifying the credential identifier;
  2) `CredType`, identifying the type of the credential;
  3) `Expiration`, specifying the credential expiring date;
  4) `IssueRep`, denoting the unique address of the issuer's server.[2]
- *content-template* is a list collecting attribute specifications p_{ct}. □

In the following definition and throughout the paper, we denote with $sign(doc)$ and $hash(doc)$ the signature and the hash value computed over document *doc*.
We are now ready to define a privacy enhanced credential.

Definition 4.2 (**Privacy enhanced credential**). Let $p_{ct} =<$*header-template,content-template*$>$ be a privacy enhanced credential template, and let k be the number of attributes collected in *content-template*. A privacy enhanced credential *pc* instance of p_{ct} is a tuple $<$ *header, content (sign(header|content)* where:

- For each attribute specification p in *header-template*, *header* contains a pair (*p-name*, *p-value*), where *p-name* is the name of the attribute specified by p, and *p-value* is a value compatible with its domain;

- For each attribute specification p in *content-template*, *content* contains $hash($*p-name*$|$*p-value*$|$*random*$)$, where *p-name* is the name of the attribute specified by p, *p-value* is a value compatible with its domain, and *random* is a random string of bits;

- *sign(header|content)* is the signature computed over the header concatenated with the credential content. □

Figure 1 shows an example of privacy enhanced credential, encoded using an XML compliant language.
A credential proof is a particular state of a privacy enhanced credential, where the header is plain and the content is hidden, while the signature over the whole document can be verified. More formally, the *credential proof* of a *privacy enhanced credential* is defined as follows.[3]

[2] Identified by a URI [25].

[3] In the definition and throughout the paper we use the dot notation to denote the component of a tuple. Thus, given a tuple t we use $t.c$ to denote the value of the component c of tuple t.

Definition 4.3 (**Credential proof**). Let c be a privacy enhanced credential, and let k be the number of attributes in c. A credential proof for c is a tuple: $\langle$ *c.header, hash(c.p-name*$_1$*|c.p-value*$_1$*|random*1*), ..., hash(c.p-name*$_k$*|c.p-value*$_k$*|randomk), sign((c.header, hash(c.p-name*$_1$*|c.p-value*$_1$*|random*1*), ..., hash(c.p-name*$_k$*|c.p-value*$_k$*|randomk))*$\rangle$. □

In what follows, when a credential proof for a credential *cred* has been disclosed, we say that *cred is proven*. Indeed, the credential receiver has an immediate proof of the credential, and a sufficient level of trust can be reached to advance the negotiation. During the policy evaluation phase, when a credential is requested, the credential proof can be safely released as the corresponding policy is satisfied, before having found a trust sequence. In this way the receiver party is ensured that the other party possesses the requesting credential, even if it cannot immediately access its content, unless explicitly required by parties adopted strategies. The protocol for proving a credential can be sketched as follows:

`Requester::` `Request` *cred*
`Cred_onwner::` `Send` $\langle$*cred.header*, *hash*(*cred.p-name*$_1$*|cred.p-value*$_1$*|random*1), ..., `hash(`*p-name*$_k$*|p-value*$_k$*|randomk*`))`, `sign((`*cred.header*, `hash(`*cred.p-name*$_1$*|cred.p-value*$_1$*|random*1), ..., `hash(`*cred.p-name*$_k$*|cred.p-value*$_k$*|randomk*$\rangle$
`Requester::` `Check` *cred.header*
`Requester::` `Verifies sign((`*cred.header*, *hash*(*cred.p-name*$_1$*|cred.p-value*$_1$*|random*1), ..., *hash*(*cred.p-name*$_k$*|cred.p-value*$_k$*|randomk*)

The header of each credential is thus sent as plain text, whereas properties names and values are replaced with hash values. Hash values are generated for each attribute of the credential as introduced in Section 4.1.1. By using the issuer reference contained in the header, the credential receiver can verify credential ownership and validity. For example, during an on-line purchase, if a credit card is asked, one can directly verify by the bank whether the credential is still valid and whether the owner is actually the party who presented it, without knowing the credit card number until the success of the negotiation is not certain.
At the end of the policy evaluation phase, when a trust sequence has been found, all or a subset of random values are disclosed, allowing the receiver to verify credential attributes actual content. If the verification process fails, the receiver can eventually notify this to the issuer and abort the negotiation.
The protocol to reveal attribute content, say attribute *name_j*, is as follows:

`Cred_owner::` `Send` *name_j*, *value_j*, *random_j*
`Requester::` `Compute` $H = hash(value\_j|random\_j)$
`Requester::` `Verifies hash(`*name_j*|*value_j*|*random_j*`)` `with` H

Finally, it is worth to note that a credential proof is a powerful means to address those scenarios where a party is actually interested in verifying credential possession and not really in attribute credential values. As the header collects information proving credential possession header disclosure can immediately satisfies the policy without further exchanges. Typical scenarios are those requiring id cards proving membership to institutions like companies, libraries, gyms and so on.

4.2 Policy language

Besides credentials, trust negotiation relies on disclosure policies, specifying trust requirements that must be satisfied in order to access the requested resource. Beyond the specific formalism adopted, disclosure policies are often modeled as expressions specifying two types of information: the target resource for which the policy is specified, and the credentials to be disclosed, eventually specifying

conditions on them. Next definition formalizes this concept, according to the Trust-$\mathcal{X}$ syntax. The target resource is denoted as R, whereas the requested credentials are denoted by means of *terms*. A term specifies a credential name and eventually conditions against some of its attribute values.

Definition 4.4 (Rule) [3] A rule is an expression of one of the following forms:
1)$\mathbf{R} \leftarrow \mathcal{T}_\mathbf{1}, \mathcal{T}_\mathbf{2}, ., \mathcal{T}_\mathbf{n}$, n$\geq$ 1, where $\mathcal{T}_\mathbf{1}, \mathcal{T}_\mathbf{2}, ., \mathcal{T}_\mathbf{n}$ are terms and **R** is the name of the target resource.
2) $\mathbf{R} \leftarrow$ **DELIV**. This kind of policies are called *delivery policies.* □

A delivery policy states that no further information is requested for disclosing the requested resource. In what follows we assume each rule to be uniquely identified by an id.

Example 1 *Alice is a student at KTH university wishing to obtain a loan for her university studies. NBG Bank offers special Student loans for promising students. Furthermore, NBG has an on-line service, called HelpStudent, to submit loan applications.*
To complete the application HelpStudent adopts a policy p_1 requiring a credit card to pay for the application fee. The corresponding rule is the following: $r = loan \leftarrow CreditCard$.

Although this specification captures all the basic information required to carry on a negotiation, it is not expressive enough to specify other crucial information that may be associated with a policy, such as, for instance, its usage, its prerequisites, or the privacy policies for the requested credentials. For this reason, in this paper we enhance the policy language by adding a set of information referred to as *policy context*. The goal is to integrate the basic rule defining a policy with a structured set of information to be used during trust negotiation process. Before formally defining a policy we thus need to introduce the notion of policy context.

Definition 4.5 (Policy context). Let $R \leftarrow \mathcal{T}_1, \mathcal{T}_2, ., \mathcal{T}_n$ or $R \leftarrow DELIV$ be a rule, and let **rid** be its identifier. A **context** for **rid** is a pair: $<$ **pol_prec_set**, **priv** $>$ where:
– **pol_prec_set** $= \{p_1, \ldots, p_k\}$ is a possibly empty set of policy identifiers,[4] named policy precondition set, where $\forall\, p \in$ **pol_prec_set**, the corresponding *rule* is of the form $R \leftarrow \mathcal{T'}_1, \mathcal{T'}_2, ., \mathcal{T'}_k$;
– **priv** is a privacy policy;
All the components of a context are optional. □

The context of a policy is thus a set of information to be associated with a given rule. More precisely, the *pol_prec_set* component is a set of policy identifiers such that at least one of the policy needs to be satisfied before the disclosure of the policy with which the precondition set is associated. The idea of policy preconditions is to protect sensitive policies by introducing an order in policy disclosure. In particular, policies belonging to a precondition set are related to the same resource required by the rule within which they are associated. This aspect has been extensively investigated by us in [3] and thus we do not further elaborate on it. By contrast, the *priv* component is a new feature of a context and denotes a P3P privacy policy. The task of privacy policies is thus to complement disclosure policies, specifying whether the information conveyed by the credentials will be collected and/or used. Privacy policies may also specify the management of the portions of credential content (if any), not explicitly requested by the associated policy but anyway obtained as part of the credential. Indeed, upon receiving a credential, unless protected attributes are used, the recipient obtains the whole credential, and not only the attributes requested to satisfy the policy. We further reason on P3P policy encoding in Section 6. We are now ready to formally define a disclosure policy.

Definition 4.6 (Disclosure policy). A disclosure policy is a pair: $<$ **rid**, **context** $>$ where: **rid** is a rule identifier, and **context** is the optional context associated with the rule. □

[4]We assume that each policy is identified by a unique identifier.

```
<POLICY xmlns="http://www.w3.org/2000/P3PV1>
....
    <STATEMENT>
        <DATA-GROUP>
          <DATA ref="http://www.TrustX.repos.credtype#idCard.name">
          <DATA ref="http://www.TrustX.repos.credtype#idCard.lastname">
          <DATA ref="http://www.TrustX.repos.credtype#idCard.birthdate">
          <DATA ref="http://www.TrustX.repos.credtype#idCard.key">
              <CATEGORIES>
              <purchase/>
              </CATEGORIES>
          </DATA>
        </DATA-GROUP>
      <ACCESS> <contact_and_other> </ACCESS>
      <PURPOSE resolution-type="independent">
        <current/>
        <develop />
      </PURPOSE>
      <RECIPIENT> <ours/><same/></RECIPIENT>
      <RETENTION> <stated-purpose/> </RETENTION>
    </STATEMENT>
    <STATEMENT>
          <DATA-GROUP>
              <DATA ref="http://www.TrustX.repos.credtype#idCard.street">
              <DATA ref="http://www.TrustX.repos.credtype#idCard.stateprov">
              <DATA ref="http://www.TrustX.repos.credtype#idCard.postalCode">
              <DATA ref="http://www.TrustX.repos.credtype#idCard.country">
                <CATEGORIES><purchase/> </CATEGORIES>
          </DATA>
          </DATA-GROUP>
        <ACCESS> <contact_and_other> </ACCESS>
          <PURPOSE>
            <contact/>
            <individual-decision>
        </PURPOSE>
        <RECIPIENT> <ours/></RECIPIENT>
          <RETENTION> <business-practices/> </RETENTION>
    </STATEMENT>
</POLICY>
```

Figure 2: Example of fine grained privacy policy

Example 2 *Suppose HelpStudent asks the applicant the ID Card after receiving the Credit Card, to check card ownership and collect applicant address information. Moreover, suppose that HelpStudent maintains a database of customers personal data. In particular, it collects applicant ID Card to proceed the loan application. Thus, the rule associated with the disclosure policy requiring ID Card will be* $(loan \leftarrow idcard)$*, whereas a possible context is* $(\{p_1\}, priv)$*, where* p_1 *is the id of the policy of Example 1, and priv is the privacy policy informing user about id card management. The associated P3P policy is shown in Figure 2.*

We denote with the term *Policy Base* ($\mathcal{PB}$) the encoding of all the disclosure policies associated with a party. Next section shows how the privacy policy component of a policy context can be exploited to negotiate resources.

5 Privacy preserving trust negotiations

In the following sections we show how the features presented so far, i.e., privacy enhanced credentials and contexts associated with a policy, can be used to carry on a privacy preserving trust negotiation.

5.1 Using privacy enhanced credentials in trust negotiation

The following examples show how privacy enhanced credentials may be used to successfully complete a negotiation strongly protecting privacy, and simultaneously helping to deal with situations which would cause a negotiation failure adopting a traditional trust negotiation protocol.
Suppose Alice is a patient of the *Health Clinic* and wants to buy the drugs she needs by an on-line pharmacy. Suppose that the pharmacy is allowed to sell this kind of drugs Alice needs either to doctors or by prescription of doctors working at the Health Clinic. Further, the pharmacy also needs to obtain Alice patient card issued by the clinic and a valid credit card, to complete the transaction. On Alice side, suppose that she is willing to disclose the requested credentials only if the pharmacy presents a credential proving pharmacy affiliation with the hospital since the patient id-card conveys sensitive information about Alice health. The corresponding rule will be: $Patient\_Card() \leftarrow Health\_Clin\_Aff()$. If the clinic is willing to disclose its affiliation only to clinic patients and/or doctors, then the adopted rule will be the following: $Health\_Clin\_Aff() \leftarrow Patient\_Card()$. In a traditional negotiation, such rules will create a *negotiation deadlock*, thus causing negotiation failure, even if both parties possess the requested properties and associated credentials. Such deadlock may be avoided by using privacy enhanced credentials. During the policy evaluation phase parties may prove each other credential possession without actually revealing credential content until having received all the requested credential proofs.
Selective disclosure of a credential can also be used to strengthen parties privacy protection. As a simple example, consider the scenario of a customer purchasing books from an on-line store. Suppose that the on-line store requires a credit and id card for both verifying credit card ownership and retrieving customer home address, in order to ship the book to the customer personal address. If the customer does not want to show his/her remaining information conveyed in the id-card (like the date and place of birth) he/she can send the id-card hiding all the sensitive information not strictly required by the on line store, without failing the process. Finally, note also that in such context the customer may choose whether to immediately send the requested attributes or wait until the policy evaluation has been completed and the requested credentials have been sent.

5.2 Privacy policies in Trust-$\mathcal{X}$ negotiations

A Trust-$\mathcal{X}$ negotiation is organized according to two main phases, the former devoted to policy exchange, and the latter to credentials disclosure. The key phase of the process is the policy evaluation phase where trust requirements are exchanged to determine possible sequences of credentials for successfully completing the negotiation. In addition, on top of the policy evaluation phase, Trust-$\mathcal{X}$ provides an *introductory phase*, to exchange preliminary information whose main goal is to reach an agreement on privacy requirements. Indeed, besides the specific way of approaching trust during negotiations, privacy concerns are a common feature of each Trust-$\mathcal{X}$ negotiation. The key component of the introductory phase is the *privacy agreement* sub-phase. The specific aim of the privacy agreement sub-phase is to reach a preliminary agreement on data collection and use before starting sensitive information exchange. The agreement, due to the mutual exchange of information characterizing a negotiation, is thus reached by communicating to the counterpart both privacy practices and preferences, using coarse grained P3P policies and privacy preferences rules. We assume user preferences are expressed using the APPEL [8] language, although other languages can be used as well (e.g., [2]). By specifying high level privacy policies, parties can communicate the types of data they will collect without having to enumerate every individual data. This type of policies can be implemented using P3P syntax by describing data using the <dynamic><miscdata/><dynamic> element and the categories to which the information to be exchanged belongs to.
Once a prior agreement on data management is reached, parties can enter into the core of the pro-

```
<POLICY xmlns="http://www.w3.org/2000/P3PV1>

......
    <STATEMENT>
        <DATA-GROUP>
          <DATA ref="#dynamic.misc.data" >
              <CATEGORIES>
              <purchase/> <uniqueid/> <state/>
              </CATEGORIES>
          </DATA>
        </DATA-GROUP>
      <ACCESS> <contact_and_other> </ACCESS>
        <!-- Use (purpose)-->
      <PURPOSE resolution-type="independent">
        <current/>
        <develop />
      </PURPOSE>
      <RECIPIENT> <ours/><same/></RECIPIENT>
      <RETENTION> <stated-purpose/> </RETENTION>
    </STATEMENT>

    <STATEMENT>
          <DATA-GROUP>
              <DATA ref="#dynamic.misc.data" >
              <DATA ref="#user.home-info.online.postal" >
                <CATEGORIES><purchase/> </CATEGORIES>
          </DATA>
          </DATA-GROUP>
        <ACCESS> <contact_and_other> </ACCESS>
          <PURPOSE>
            <contact-required="opt-in"/>
            <individual-decision="opt-in"/>
        </PURPOSE>
        <RECIPIENT> <ours/></RECIPIENT>
          <RETENTION> <business-practices /> </RETENTION>
    </STATEMENT>
</POLICY>
```

Figure 3: Example of coarse grained P3P policy

cess and start the policy evaluation phase. Note that under this approach the subsequent phases of a negotiation can be carried out without worrying about data collection and use. However, since parties are not aware of the actual counterpart requirements, resources requiring ad hoc privacy policies might be involved while evaluating parties policies. To cope with this possibility each policy context may contain in the optional *priv* field a P3P fine grained privacy policy. If desired by the parties, each policy can be sent accompanied by the related P3P policy, specifying how the information collected will be managed and for which purpose. Each time a P3P policy is received the receiver has to evaluate first the disclosure policy compliance and then check whether his/her privacy preferences comply with the P3P policy. Similar to privacy policies, privacy preference rules can also express either coarse grained preferences to be exchanged in the agreement phase, or more fined grained preferences associated with credentials reputed privacy sensitive, and be exchanged as the credentials are involved in the process.

Example 3 *With reference to the loan scenario of Example 1, consider now two different privacy policies. The first P3P policy is an example of a coarse grained privacy policy to be exchanged during the privacy agreement phase. Suppose that, to accept students application, HelpStudent needs to obtain certain information by the applicants, and store it for a week to check user compatibility with bank loan policies. However, HelpStudent also adopts a privacy policy stating that NBG offers personalized loan recommendations, for which it needs to collect customer personal information. The resulting P3P policy to be matched against the applicant privacy preferences is shown in Figure*

3. The first STATEMENT says that customer personal information and miscellaneous purchase data (e.g., credit card number, type of loan required) will be used for completing the loan transaction. The second STATEMENT allows HelpStudent to use miscellaneous data for creating personalized recommendations and email them to the customer.
The second privacy policy, reported in Figure 2, is an example of policy referring to a credential, (see Example 2). The policy states privacy practices related to credentials of type `IdCard`*. Such policy is sent together with policies requiring user ID Card, and inform the user about the credential receiver intended use to collect user contact information to complete the electronic transaction.*

6 Trust-$\mathcal{X}$ architecture

Trust-$\mathcal{X}$ is composed by several components, sketched in Figure 4. As shown, the main components of the system are a *Policy Base*, storing disclosure policies, the $\mathcal{X}$-*Profile* associated with the party, a *Tree Manager*, managing the negotiation tree, and a *Compliance Checker*, testing policy satisfaction and generating request replies. The compliance checker also checks local policy satisfaction and verifies at runtime the validity and ownership of remote credentials. The goals of the system components are essentially to support policy and credential exchange and to test whether a policy is satisfied. In addition to those basic elements several modules for the management of privacy policies are also included. Current implementations of privacy systems [17] usually have two components deploying P3P. Web sites install privacy policies and reference files, at their sites, using tools like [11, 13]. Then, as users browse sites, their preferences are checked against site policies. This simple schema is not adequate in our context, in that we are dealing with both user and server sides. Indeed, each Trust-$\mathcal{X}$ entity acts during a negotiation as a server requesting personal data as well as a user disclosing personal information. The framework should thus support both sides during negotiations. In what follows we illustrate users and server modules separately. Then, we merge these two visions into a unique framework and show how they fit together. In presenting the modules we focus on P3P policies to be exchanged during the policy evaluation phase. We recall that (cfr. Section 5) P3P policies are also exchanged during the policy agreement phase. However, such policies are coarse grained since they refer to the whole negotiation and not to a specific credential. As such, they can be specified and evaluated using standard mechanisms adopted by web sites.
For a policy sender, the system should support the following actions:

1. mapping credential schema into data schema usable for privacy policy specification;
2. specifying privacy policies about these data using P3P;
3. providing agents supporting privacy policies among negotiations.

Creation of P3P policies can be executed off-line before negotiations start. The process is sketched on top of Figure 4 (dashed lines). As shown, the module in charge of encoding P3P is the `Policy wizard`. Given a disclosure policy *dp*, the module extracts the corresponding credential schema[5] (see Figure 4, arrow 1) $C_1, .. C_n$ required by *dp* from the `credential schema repository`. This module is implemented as a credential chain tool, to retrieve credential schemes from public issuer repositories and by a local cache storing the most widely used schemes.
Once the required schemes are retrieved, credentials content is considered, to identify data to be collected. Credentials content can be analyzed under two different perspectives. If the information to be collected is a set of properties and the credential actually represents only the envelope to transmit these data then the policy can be specified as a conventional P3P policy, that is, using built in data schemes and categories provided by the standard, without referring to the particular

[5] We use the term schema and type interchangeably, in this context.

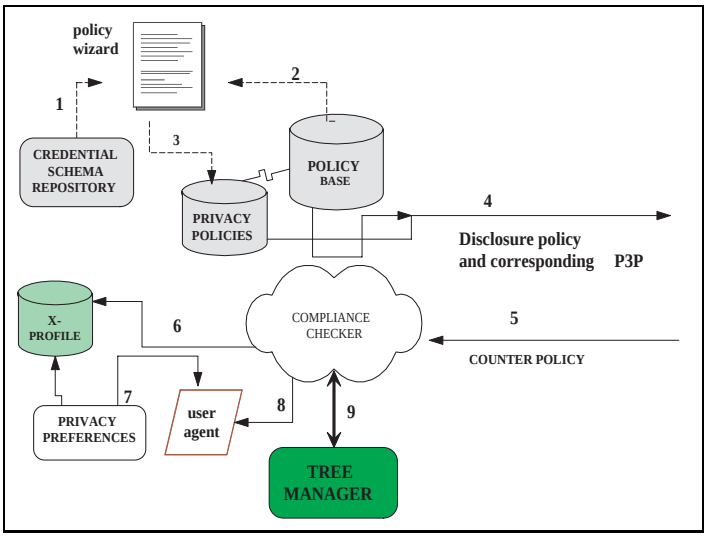

Figure 4: Trust-$\mathcal{X}$ framework. Dashed lines denote off-line operations. Arrow labels denote the flow of the operations.

credential collecting the requested attributes. By contrast, if the key information is the credential itself, then the policy should refer not only to the attributes in the credential but also to the credential itself. For instance, if a web server wants to cache a whole credential to create a data base collecting customers data, it has to refer to the specific certificate, specifying its ID and issuer public key. In such cases it is mandatory to extend P3P data schema to encode the data structure underlying the credential. Privacy policies are encoded according to version 1.1. of P3P [24], that provides a new format for expressing P3P data schema in a simpler way than the previous one. The new format uses the XML Schema Definition (XSD) format which can be validated against an XML schema. Since Trust-$\mathcal{X}$ credentials are defined in terms of DTDs it is possible to encode the policy directly referring to the schema corresponding to a credential, by simply translating the DTD with XSLT [28] in XSD. An example of P3P policy referring to a credential schema is shown in Figure 2.

Once the corresponding data schema has been encoded the policy creation wizard can complete policy encoding, specifying data management (for which purpose the data will be collected, for how long..and so forth), defined according to the local privacy practice. The privacy policy is finally linked to the corresponding disclosure policy.

On the other side, the policy receiver must be equipped with tools for describing privacy preferences and matching policy against privacy preferences. Tools for describing privacy preferences can be implemented as ad-hoc policy editors, able to encode user preferences into a set of rules to be used to evaluate remote P3P policies. Policy matching, by contrast, can be executed by a user agent integrating the compliance checker module. The behavior of the user agent is modeled by function *Privacy_matching*(), reported in Figure 5. The function is invoked each time a disclosure policy is received, even if it is not actually accompanied by a specific P3P policy. If a privacy policy is attached to the disclosure policy, policy check is performed between the privacy policy and the preference rules of the receiving party, with respect to the credentials requested by the disclosure policy with which the privacy policy is associated. If no privacy policy is associated with the disclosure policy, then the preference rules are checked against the privacy policies exchanged during privacy agreement phase, denoted as *Priv_nego*, in Figure 5. Similarly, if no privacy policy is associated with the disclosure policy but a preference rule has been specified for the credentials requested by

the policy, then the preference rule is checked against the coarse grained privacy policy exchanged during the privacy agreement phase.

```
Function Privacy_matching(priv)
Input :
    priv: a privacy policy or an empty message;
Output :
    Accept(priv) or Refuse(priv)
Precondition:
    Priv_nego is the remote P3P policy exchanged in the privacy agreement phase
    Gen_priv_pref are the local privacy preference rules exchanged in the privacy agreement phase
begin
  if priv ≠ ∅      %matching privacy preferences
      Let p_c be the policy to which priv is associated
      Let Spec_rules be the set of specific local preference rules not belonging to Gen_priv_pref;
      if ∃rule ∈ Spec_rules for credentials requested in p_c then
      % Match is a function checking compliance between a privacy rule and a p3p policy
        if Match(rule, priv)=TRUE then
        % there exists a specific rule for the requested credentials
          Return(Accept(priv))
        else Return(Refuse(priv))
      else
        if Match(priv_pref, priv)=TRUE then
          Return (Accept(priv))
        else Return(Refuse(priv))
  else      %  p_c.priv = ∅
      Let Spec_rules be the set of specific local preference rules in p_c credentials not belonging to Gen_priv_pref;
      if ∃rule ∈ Spec_rules for credentials requested in p_c then
        if Match(rule, Priv_nego)=TRUE then
        % there exists a specific rule for the requested credentials
          Return (Accept(priv))
        else Return(Refuse(priv))
end
```

Figure 5: **Function** Privacy_matching()

Modules characterizing the two sides are merged into a unique framework, and are complementary to each other. The first set of modules is used to integrate conventional disclosure policies sent to the counterpart with specific privacy policies. The latter, by contrast, acts as a further filter when remote policies asking for credentials conveying personal information are received.
As new features are added to Trust-$\mathcal{X}$, an increasing computational effort is required to carry out the process. For example, if a disclosure policy is sent together with a privacy policy, the compliance checker has actually to make one or two additional operations (arrow 7) in order to check if the P3P policy fires. However, due to the simplicity underlying the P3P platform the time required to perform this kind of checking is minimal, and it does not really impact system performance. Moreover, it is expected that in most cases the overload caused by P3P policy exchange will be actually very limited and confined in the privacy agreement phase. Indeed, during the subsequent phase only additional privacy policies not covered by the previous ones may be exchanged, if required by the parties.

7 Conclusions

In this paper we have presented a system for trust negotiation specifically designed for preserving privacy during a negotiation. The system provides support for P3P policies, that can be exchanged at various steps of the negotiation, and for different credential formats, providing different degrees of privacy protection. We are currently developing a suite of strategies to carry on a negotiation,

that exploit and extend the notion of context associated with a policy, to allow one to trade-off among efficiency, robustness, and privacy requirements. Additional future work includes an implementation of both the proposed system and the credential formats. Other extensions include the development of mechanisms and modules to semi-automatically design privacy policies to be associated with disclosure policies. Additionally, we plan to fully support P3P 1.1, once the new version will be standardized.

References

[1] R. Agraval, J. Kiernan, R. Srikant, X. Yu. Implementing P3P using database technology. *19th International Conference on Data Engineering*, Bangalore, India, March 2003.

[2] R. Agraval, J. Kiernan, R. Srikant, Y. Xu. An X-Path based preference language for P3P. *Twelfth International World Wide Web Conference*, Budapest, Hungary, May 2003.

[3] E. Bertino, E. Ferrari, A.C. Squicciarini. Trust-$\mathcal{X}$: a Peer to Peer Framework for Trust Establishment. To appear in *IEEE Transaction on Knowledge and Data Engineering (TKDE)*

[4] E. Bertino, E. Ferrari, A.C. Squicciarini. $\mathcal{X}$-TNL: an XML based language for Trust negotiation. *Fourth IEEE International Workshop on Policies for Distribuited Systems and Networks*, Como, June 2003.

[5] P. Bonatti, P. Samarati. Regulating Access Services and Information Release on the Web. *7th ACM Conference on Computer and Communications Security*, Athens, Greece, November 2000.

[6] S. A. Brands, Rethinking Public Key Infrastructure and Digital Credentials. *MIT Press*, Cambridge, Massachusetts 2000.

[7] L.Cranor, M. Langheirich, M. Marchiori. The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. *W3C Reccomandation*, April 2002. Available at `http://www.w3.org/P3P/brochure.html`

[8] L.Cranor, M. Langheirich, M. Marchiori. A P3P Preference Exchange Language 1.0 (APPEL1.0) *W3C Working Draft*, April 2002.

[9] A. Herzberg, Mihaeli, Y. Mass, D. Naor, and Y. Ravid. Access Control Meets Public Infrastructure, Or: Assigning Roles to Strangers. *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2000.

[10] R. D. Jarvis. Selective Disclosure of Credential Content during Trust Negotiation. *Master of Science Thesis, Brigham Young University*, Provo, Utah, April 2003.

[11] IBM Tivoli Privacy Wizard
Available at: `http://www.tivoli.resource_center/maximize/privacy/wizard_code.html`

[12] A. Herzberg, Y. Mass, Relying Party Credentials Framework *RSA Conference,* San Francisco, CA, April 2001.

[13] JRC P3P Resource Centre: `http://p3p.jrc.it`

[14] N. Li, John C. Mitchell and W. H. Winsborough. Design of a Role-Based Trust Management Framework. *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2002

[15] M. Naor Bit commitment Using Pseudorandomness. *Advances in Criptology- 89 Lecture Notes in Computer Science, Vol. 435* Springer-Verlag, New York, 1990.

[16] P. Persiano and I. Visconti. User Privacy Issues Regarding Certificates and the TLS Protocol *Proceedings of the ACM Conference on Computer and Communication Security*, Athens, Greece, 2000.

[17] References for P3P implementation `http://www.w3org/P3P/implementations`

[18] K.E. Seamons, M. Winslett, T. Yu. Limiting the disclosure of Access Control Policies during Automated Trust Negotiation. *Network and Distributed System Security Simposium*, San Diego, CA, February 2001.

[19] K.E. Seamons, M. Winslett, T. Yu. Protecting Privacy During On Line Trust Negotiation. *2nd Workshop on Privacy Enhancing Technologies* San Francisco, CA, April 2002.

[20] W. Stallings. Cryptography and Network Security: Principles and Practice, second edition *Prentice Hall, 1999*

[21] A. F. Westin. Privacy and Freedom. *Atheneum, New York, 1967.*

[22] M. Winslett et. al. Negotiating Trust on The Web. *IEEE Internet Computing, Vol. 6, No 6*, pp. 30-37, November 2002.

[23] W. H. Winsborough, N. Li. Towards Practical Automated Trust Negotiation. *IEEE 3rd Intl. Workshop on Policies for Distributed Systems and Networks*, Monterey, CA, 2002.

[24] P3P- The Platform for Privacy Preferences, Version 1.1
Available at: `http://www.w3.org/P3P/1.1/`

[25] Uniform Resource Identifiers Naming and Addressing: URIs, URLs, ...
Available at: `http://www.w3.org/addressing`

[26] T. Yu, M. Winslett. A Unified Scheme for Resource Protection in Automated Trust Negotiation. *2003 IEEE Symposium on Security and Privacy*, Oakland, CA, 2003.

[27] World Wide Web Consortium Extensible Markup Language (XML) 1.0, 1998.
Available at: `http://www.w3.org/TR/REC-xml`

[28] XSL Transformations (XSLT). Version 1.0 W3C Recommendation, 1999.
Available at: `http://www.w3.org/TR/xslt`