

Policy Migration for Sensitive Credentials in Trust Negotiation

Ting Yu
Department of Computer Science
North Carolina State University
yu@csc.ncsu.edu

Marianne Winslett
Department of Computer Science
University of Illinois at Urbana-Champaign
winslett@cs.uiuc.edu

ABSTRACT

Trust negotiation is an approach to establishing trust between strangers through the bilateral, iterative disclosure of digital credentials. Under automated trust negotiation, access control policies are associated with sensitive credentials to control under what circumstances those credentials can be disclosed. Ideally, the information in a user's sensitive credential should not be known by others unless the corresponding policy is satisfied. However, the original model for user interaction in trust negotiation has pitfalls which can be easily exploited to infer one's private information, even if access control policies are strictly enforced. To preserve one's privacy, a more flexible interaction model for trust negotiation is required. On the other hand, it is also desirable for two parties to be able to establish trust whenever possible. There is potentially a conflict between privacy preservation and the assurance of a successful trust negotiation. In this paper, we identify the situation where sensitive information can be inferred through observing one's behavior in trust negotiation. Then we propose *policy migration* as one approach to preventing such inference. Compared to previously proposed approaches, policy migration has a low management overhead, and provides a nice balance between inference prevention and guarantees of success in trust establishment. We also discuss the limitations of policy migration, and possible directions for more comprehensive solutions.

Categories and Subject Descriptors

K.6.5 [Management of Computing and Information Systems]: Security and Protection

General Terms

Security

Keywords

trust negotiation, policy migration

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WPES'03, October 30, 2003, Washington, DC, USA.
Copyright 2003 ACM 1-58113-776-1/03/0010 ...\$5.00.

1. INTRODUCTION

Global competitive pressure and the possibility of severe security breaches are forcing organizations and individuals to develop the ability to rapidly form relationships and cooperate with one another to solve urgent problems. Such cooperation often involves unanticipated resource sharing across organizational boundaries. As disparate groups attempt to collaborate to conduct sensitive processes and detect and respond to problems, their efforts to provide rapid and efficient response are hindered by traditional approaches to access control in decentralized systems, which are based on subject identity and are administered centrally. Organizations and individuals require nimble security facilities that will enable them to rapidly and efficiently access each others' resources and integrate the information provided by the resources, while offering specific privacy guarantees.

Automated trust negotiation (ATN) is a new approach to access control and authentication in open, flexible systems such as those described above. ATN enables open computing by assigning an access control policy to each resource that is to be made accessible to "outsiders". An access control policy describes the *properties* of the parties allowed to access that resource, in contrast to the traditional approach of listing their identities. In ATN, one's properties are demonstrated through the use of digital credentials, the digital analogues of paper credentials that people carry in their wallets: digitally signed assertions by a credential issuer about the credential owner. Digital credentials often contain sensitive information about the credential owner. Thus their disclosure is also protected by access control policies.

A trust negotiation is triggered when one party requests to access a resource owned by another party. Since each party may have policies that the other needs to satisfy, trust is established incrementally through bilateral disclosures of credentials and requests for credentials. The purpose of trust negotiation is to find a credential disclosure sequence (C_1, \dots, C_k, R) , where R is the service or resource to which access was originally requested, such that when credential C_i is disclosed, its access control policy has been satisfied by credentials disclosed by the other party.

In trust negotiation, each party adopts a *trust negotiation strategy*, which controls the exact content of the messages exchanged with others, i.e., which credentials to disclose, when to disclose them, and when to terminate a negotiation. A simple strategy for establishing trust is to disclose every credential whose access control policy has been satisfied by credentials received from the other. This approach results in needless credential disclosures, even though the

other party is authorized to receive them. To remedy this problem, typically, each party also discloses access control policies so that they can focus on those credentials actually needed to advance the negotiation.

The following online transaction example shows what a trust negotiation process may look like. Suppose Alice is a graduate student who wants to buy textbooks at the beginning of a semester. She surfs the web and finds a good deal from Bob, who is the owner of a small online book store. Bob offers a special discount to graduate students whose GPA is over 3.0. The negotiation process may look as follows.

Step 1 Alice requests the special discount for this transaction.

Step 2 Bob replies with the policy for the discount, requesting Alice to show 1) her student ID to prove that she is a graduate student; and 2) her official transcript to prove that her GPA is over 3.0.

Step 3 Alice does not mind showing her student ID to others. But her official transcript contains some quite sensitive information. Besides disclosing her student ID, she also tells Bob that in order to see her official transcript, Bob must prove that his online book store belongs to the Better Business Bureau.

Step 4 Fortunately, Bob does have a Better Business Bureau membership card. And he is willing to tell anybody about this fact. So he discloses his BBB membership card to Alice.

Step 5 At this point, Alice knows that she can trust Bob and discloses her official transcript to Bob.

Step 6 Now Bob has received both Alice’s student ID and her official transcript. After verifying she is a graduate student and her GPA is over 3.0, Bob gives Alice the special discount for this transaction.

Note that in practice the above process is conducted by security agents who interact with each other on behalf of users. A user only needs to specify access control policies for credentials and other resources. The actual trust negotiation process is fully automated and transparent to users.

Access control policies play a central role in protecting one’s privacy. Ideally, Alice’s sensitive information should not be known by Bob unless Bob has satisfied the corresponding access control policy. As we mentioned above, each credential is associated with an access control policy, specifying under what circumstances that credential can be disclosed. The underlying assumption is that if a credential is not disclosed, information in that credential will not be known by others. However, depending on the way two parties interact with each other, one’s private information may flow to others in various forms. Though credential disclosures are the major form of information flow, controlling them alone is far from sufficient for providing strong protection for users’ private information.

Let us revisit the online transaction example. Once Bob tells Alice his access control policy for getting the discount, normally Alice will check her own credentials and see whether the policy can be satisfied. If she can, since her official transcript is sensitive, she will respond with her policy for that credential, as happened in step 3. On the other hand, if she cannot satisfy Bob’s policy, a natural behavior of Alice is

to terminate the negotiation since it does no good to waste time on a negotiation that will definitely fail. However, by merely observing Alice’s response to the policy, Bob can easily infer whether her GPA is over 3.0, even though her official transcript is not disclosed yet.

The main reason for such information flow is the overly strong semantics of the current interaction model of trust negotiation. Note that when disclosed, an access control policy \mathcal{P} is in effect a query, asking whether the other party possesses the right credentials to satisfy \mathcal{P} . And one’s behavior thereafter serves as a “yes/no” answer to the query, which is not protected by access control policies at all.

From the above analysis, it seems we have two possible approaches to this problem — assigning a weaker semantics to users’ behavior or using additional policies to control when to make a response. In this paper, we identify desiderata for inference prevention in trust negotiation. Then we analyze the pros and cons of existing approaches according to the desiderata. Finally, we introduce policy migration as our approach and analyze its advantages and limitations. The basic idea of policy migration is to integrate sensitive credentials’ policies into the policies of other credentials that are relevant to the negotiation but are not sensitive. Such an approach makes it hard for an attacker to automatically distinguish the behavior of a party who possesses sensitive credentials from that of those who do not. On the other hand, the policies for sensitive credentials are still implicitly enforced during trust negotiation. Thus, users’ access control safeguards are not compromised.

The rest of the paper is organized as follows. In section 2, we discuss different types of inferences that may happen in trust negotiation. Desiderata for inference prevention are also presented in this section. In section 3, we describe several proposed approaches from the literature. Policy migration is presented in section 4. Section 5 concludes this paper and discusses directions for future work on privacy preservation in trust negotiation.

2. PRIVACY VULNERABILITY IN TRUST NEGOTIATION

In [23], Seamons et al. identified two types of privacy vulnerability. The first type is called *attribute-sensitive credentials*, and the transcript vulnerability shown in section 1 is of this type. Suppose Alice’s credential has an attribute (e.g., birth-date, address or social security number) whose value is considered sensitive. In the original interaction model for trust negotiation, if Bob’s access control policy \mathcal{P} specifies a constraint on that sensitive attribute, then depending on whether Alice issues a counter request in response to \mathcal{P} , Bob can infer whether Alice owns a credential satisfying that constraint. Through a series of probes similar to binary search, this vulnerability can be exploited to quickly determine the exact value of some sensitive attribute such as one’s age or credit rating. If Alice considers an attribute to be sensitive, then given a policy that places a constraint \mathcal{P} on the acceptable values of that attribute, in general Alice’s response to the disclosure of \mathcal{P} should be considered sensitive, whether or not the value of her attribute can satisfy \mathcal{P} . Therefore we call this *symmetric sensitivity*. Another property of sensitive attributes is that only the value of the attribute is considered sensitive. Alice does not mind Bob knowing that her driver’s license has a *birth_date* attribute.

The second type of vulnerability, identified in [23] and [26] independently, is called *possession-sensitive credentials*, i.e., the fact that Alice has or does not have a certain credential is considered sensitive. As an extreme case, suppose an attacker Eve wants to find out who works for the CIA, which is considered very sensitive information. She may set up a web page claiming to offer a very attractive deal. When Alice wants to get the deal, Eve replies with an access control policy asking Alice to show a credential issued by the CIA to her. If Alice responds with a policy (even a very restrictive one), then Eve will infer that Alice is connected to the CIA, because otherwise, Alice would just terminate the negotiation and leave.

Sometimes, the fact that Alice does not possess a credential may also be sensitive. For example, before a cautious patient gets advice from an online medical clinic, she may want to verify that the doctor in the clinic does have a valid medical practice license issued by a state board of medical practice. Any decent clinic will be more than willing to show the license. If the clinic chooses to terminate the negotiation, the patient can infer with high confidence that the clinic is phony.

Note that the sensitivity shown in the above two examples is not symmetric. If a person is not connected to the CIA, she will be unlikely to mind others knowing that (after all, many people do not want to have anything to do with covert agencies). In this case, non-possession is typical, natural and not sensitive. In the second example, a qualified doctor should be willing to show her medical license to anybody who wants to see it. Thus possessing such a credential is not sensitive at all. Due to this property, we also call sensitivity in this situation *asymmetric sensitivity*.

Seamons et al. [23] proposed the use of dynamic policy graphs to protect attribute-sensitive credentials. Their approach can be further generalized as a policy filtering mechanism, the details of which can be found in [28]. In the rest of this paper, we will address only the problem of possession-sensitive credentials.

As mentioned in section 1, there are two basic approaches to providing stronger protection for users' privacy in trust negotiation — assigning a weaker semantics to users' behavior or using additional access control policies to control when to make a response. However, no matter what our approach is, privacy protection is not the only concern. Recall that the primary goal of ATN is to establish trust between strangers. There is potentially a conflict between success in trust establishment and privacy preservation. For example, one may be extremely cautious and act in a very passive way — only disclosing credentials when their policies are satisfied and never responding with any counter requests. This approach guarantees that the above vulnerabilities cannot be exploited. But meanwhile, it may result in unnecessary credential disclosures (as did the simple strategy in section 1) or premature termination of a potentially successful negotiation, missing valuable business opportunities. On the other hand, the negotiation protocols proposed in [30] guarantee successful trust establishment whenever possible, but are subject to information leaks. It is desirable to find a balance between these requirements. To do so, we have identified the following desiderata for privacy protection in trust negotiation.

1. The privacy protection scheme should be **easy to deploy** in existing trust negotiation systems. The man-

agement overhead imposed on human users should be as low as possible. Ideally, a human user should only have to specify access control policies and what credentials are considered attribute-sensitive or possession-sensitive. With such information, there should be a general algorithm that dictates what a security agent needs to do to protect those sensitive credentials.

2. When possible, the privacy protection scheme should **avoid relying on pre-arranged coordination of a large group of parties**. For example, one way to protect possession-sensitive credentials is to let a large group of parties agree to respond in the same way when asked to show a certain credential, no matter whether they think that credential is possession-sensitive or not. For example, suppose that at the time a California driver's license is issued, the recipient is given standard default access control and ack policies (discussed below) for use with the license. Few parties will bother to customize or replace the default policies. Thus if Alice is sensitive about the fact that she does not possess a California driver's license, she may be able to hide her lack of possession from many parties by adopting those standard policies. However, in other cases, such as the CIA credential example used earlier, no such standard policies will be available. In such cases, the use of a protection approach that depends on coordinating the actions of a large group of parties will have to rely on the deployment of a new supporting infrastructure, which can require significant management effort, especially in widely decentralized environments. We will discuss this point further in section 3.
3. **Avoidance of unnecessary failure** of trust negotiations should also be considered when a privacy protection scheme is adopted. For example, we should avoid those approaches where a trust negotiation always fails whenever an attribute-sensitive or possession-sensitive credential is involved.

3. RELATED WORK

A number of trust negotiation systems and supporting middleware have been proposed and/or implemented in a variety of contexts (e.g., [6, 7, 12, 13, 17, 16]). Information leakage during trust negotiation is studied in [23, 25, 26] and we will discuss this work in detail.

In [23], non-response is proposed as a way to protect possession-sensitive credentials. For example, suppose Alice works for the CIA. If Bob requires that Alice show a credential issued by the CIA before she can access Bob's service, Alice will not respond to that request, as if she does not possess such a credential. She can disclose the credential later when Bob happens to disclose the right credentials that satisfy the policy for disclosing her CIA credential. This approach is easy to deploy in trust negotiation and does not rely on the cooperation of a large group of other users. But clearly a potentially successful negotiation may fail because of Alice's conservative response. In practice, such "terminate and leave" behavior is probably what CIA agents are trained to do, in order to avoid the risk of being detected. On the other hand, unilateral departure will also prevent parties with less critically sensitive credentials from accessing their desired resource using trust negotiation.

Similarly, for credentials where non-possession is sensitive, Seamons et al. [23] proposed to act as if one has that credential. For example, if Alice’s driver’s license has been revoked due to driving under the influence of alcohol, she may assign a policy \mathcal{P} to a “virtual” driver’s license. When Bob requires Alice to show her driver’s license, Alice responds with policy \mathcal{P} . Only after \mathcal{P} is satisfied by Bob will Alice tell Bob that she actually does not have a valid driver’s license. Therefore, the policy \mathcal{P} actually specifies under what circumstances Alice can tell others the fact she does not own a driver’s license. This approach works fine in this example, because typically even if an individual has a valid driver’s license, she will still have a policy to control to whom it can be disclosed. But for credentials that most people are willing to show to anybody (e.g., a company’s Better Business Bureau membership certificate), this approach is not effective. The fact that there is a policy associated with Alice’s BBB certificate strongly indicates that Alice actually does not belong to the BBB.

Another approach [26, 25] is based on the principle, “Discuss sensitive topics only with appropriate opponents”. Their framework focused on protecting one’s roles (which are also called attributes in their work). For example, being an CIA agent is a role. Therefore, protecting that a user has a CIA agent role in their framework is the analogue to protecting the fact that a user has an agent credential issued by the CIA in our framework. In their approach, for each sensitive credential, no matter whether Alice possesses it or not, she has an acknowledgment policy (an ack policy). Alice only acknowledges whether she has a credential or not after the corresponding ack policy is satisfied.

The management overhead of this approach can be high. For each sensitive credential Alice possesses, she now needs an access control policy for its disclosure and also an ack policy to determine with whom she can discuss this credential. And it is widely known that designing and maintaining good policies is a hard task, especially for individual users. Further, as for the approaches in [23], ack policies are not effective for protecting credentials that most people do not have or are willing to show freely. To make it work for those credentials, we have to organize a large group of users who agree to act in the same way to protect the interests of a small portion of the people.

In fact, if a CIA credential is the only credential Bob requires from Alice, then without the pre-arranged support of a large group of users, there is no way that Alice can protect the fact that she possesses one without jeopardizing the success of her negotiation. The reason is that people without a CIA agent credential will simply admit it and terminate the negotiation. Any other actions Alice takes will directly reveal the fact that she does possess the credential.

However, if a CIA credential is not the only required credential, can Alice protect her possession-sensitive credentials without jeopardizing the success of a negotiation, even without the cooperation of a large group of users? That is the question we will address next.

4. A LIGHT-WEIGHT APPROACH TO PROTECTING POSSESSION-SENSITIVE CREDENTIALS

Since we are dealing with possession-sensitive credentials, whose internal contents are irrelevant for our purposes, in

this section we model credentials and resources as uninterpreted symbols. Each of these resources has exactly one access control policy \mathcal{P} , where \mathcal{P} is a Boolean expression involving only credentials C_1, \dots, C_k that the other party may possess; Boolean constants *true* and *false*; the Boolean operators \wedge and \vee , and parentheses as needed. In evaluating the expression \mathcal{P} , C_i evaluates to true (is “satisfied”) if and only if the other party has disclosed credential C_i . We use $C \leftarrow \mathcal{P}$ to denote that C ’s policy is \mathcal{P} . We can distinguish between local and remote resources (by renaming propositional symbols as necessary), so that it is always clear whether a propositional symbol in a policy refers to a local or non-local credential for a party.

Resource C is *unlocked* if its access control policy is satisfied by the set of credentials disclosed by the other party. A resource C is *unprotected* if its policy is always satisfied, (i.e., C ’s policy is equivalent to *true*). The *denial policy* $C \leftarrow \text{false}$ means that the party does not possess C or is not willing to disclose it under any circumstances. A party implicitly has a denial policy for each credential it does not possess. If the disclosure of a set \mathcal{C} of credentials satisfies resource R ’s policy, then we say \mathcal{C} is a *solution* for R . Further, if none of \mathcal{C} ’s proper subsets is a solution for R , we say \mathcal{C} is a *minimal solution* for R .

To begin the discussion, we assume that Alice has only one possession-sensitive or non-possession sensitive credential, denoted C_* . For simplicity, we further assume that C_* only appears in the policy of R , the resource that Alice originally tried to access. We will revisit these assumptions at the end of the section.

If C_* is a possession-sensitive credential of Alice’s, then we consider the following scenario: Alice has $C_* \leftarrow \mathcal{P}$, where \mathcal{P} is not logically equivalent to *true*, while all the other users do not possess C_* , i.e., they have $C_* \leftarrow \text{false}$. Note that in practice, Alice may not be the only person that possesses C_* . However, since we assume that there is no cooperation of a large group of users, it is the same as if Alice is on her own.

Similarly, if C_* is a non-possession sensitive credential of Alice, then Alice has $C_* \leftarrow \text{false}$, and a second policy \mathcal{P} to control to whom she is willing to disclose this denial policy for C_* , which suggests to others that she does not possess C_* . Meanwhile, all the other users have $C_* \leftarrow \text{true}$.

4.1 The TrustBuilder-Relevant Strategy and the Disclosure Tree Strategy

The intuition behind policy migration is to make Alice’s behavior to some extent indistinguishable from that of other people. Since we assume that there is no pre-arranged cooperation from a large group of users, it is necessary for us to identify some of the common behaviors among ordinary users, so that Alice can imitate them. Such behaviors are largely determined by users’ negotiation strategies. In this section, we introduce two negotiation strategies that will be used in subsequent sections to show how Alice can avoid leaking information through her behavior.

As mentioned in section 1, each party’s security agent adopts a trust negotiation strategy, which determines what credentials and policies to disclose in the next message and when to terminate a negotiation. Formally, we model a strategy as a function, whose inputs are the set \mathcal{S} of all the credentials and policies disclosed so far, the resource R to which access was originally requested, and the set \mathcal{L} of a

party’s local policies and credentials. Its output is a message m , which is the subset of \mathcal{L} to disclose in the next message, subject to $m \cap \mathcal{S} = \emptyset$ (i.e., no duplicate disclosures can be made). In particular, when $m = \emptyset$, it is called a *failure* message, which means that a party terminates the negotiation, for whatever reason.

Since the Internet is a freewheeling place, each party should have autonomy in choosing whatever strategies meet their requirements. Given two strategies f_1 and f_2 , we say they are *interoperable* with each other if, by adopting them respectively, Alice and Bob can always establish trust whenever their access control policies allow. A maximal set of interoperable strategies, called the disclosure tree strategy (DTS) family, is identified in [30]. Next we present two strategies in the DTS family that will be useful when we discuss protection for possession-sensitive and non-possession sensitive credentials.

The first strategy is called the *TrustBuilder-Relevant Strategy*, which works as follows. After Alice receives a message from Bob, Alice identifies the set m of all the unlocked credentials and policies that are syntactically relevant to R and have not been disclosed yet. A credential C is *syntactically relevant* to resource R iff C appears in R ’s policy, or C appears in the policy of a credential C' that is syntactically relevant to R . If m is still empty, then Alice puts one or more new syntactically relevant denial policy disclosures into m , if she has any.¹ Then Alice sends m to Bob as her next message. Alice terminates the negotiation if she cannot make any new syntactically relevant disclosures. Note that a party does not need to disclose any denial policies unless no other relevant disclosures can be made.

The TrustBuilder-Relevant Strategy is simple and easy to deploy because it does not maintain any internal structures to trace the dependency relationship between credentials. On the other hand, a party using this strategy may make unnecessary disclosures.

For example, suppose Alice requests access to Bob’s resource R and Bob has $R \leftarrow A1 \wedge A2$. Further, suppose Alice has $A1 \leftarrow B1$ and $A2 \leftarrow false$. Clearly, after receiving R ’s policy from Bob, we can see that Alice will not be able to gain access to R since she does not possess credential $A2$. However, by using the TrustBuilder-Relevant Strategy, Alice will disclose $A1 \leftarrow B1$ as her next message instead of terminating the negotiation. In other words, though theoretically there is no hope to establish trust, Alice may still continue the negotiation until she cannot make any safe disclosures. Due to its simplicity, we expect that the TrustBuilder-Relevant Strategy will be popular for use in trust negotiation, in spite of its potentially irrelevant disclosures. It is especially suitable for clients with limited computational resources, e.g., PDAs and smart cards.

A different strategy of considerable theoretical interest is the Disclosure Tree Strategy, which is the most cautious strategy in the DTS family. This strategy carefully maintains internal structures to capture the current status of a trust negotiation, which will help to determine which disclosures are absolutely necessary to advance the negotiation.

¹The TrustBuilder-Relevant Strategy presented in [30] requires a party to disclose all the relevant unlocked credentials and policies, including denial policies. We slightly modify the definition of the strategy in this paper. It is easy to prove that the modified version of the TrustBuilder-Relevant Strategy still belongs to the DTS family.

After receiving a message from the other party, it chooses as its next message a minimal set of disclosures that will allow the other party to continue the negotiation. If there are several minimal sets satisfying the above condition, a user may choose the one that fits her needs best.

For example, suppose $R \leftarrow A1 \vee A2$ and Alice has $A1 \leftarrow B1$ and $A2 \leftarrow true$. When receiving R ’s policy from Bob, there are two minimal sets of disclosures that Alice can make. She can disclose $A2$ if she wants to gain access to R immediately. Or she may disclose $A1 \leftarrow B1$ if she wants to extract more information from Bob. Because the Disclosure Tree Strategy always makes minimal information disclosures during a trust negotiation, it exemplifies the potential actions of cautious users who have ample computational resources.

The key observation here is that, when using the Disclosure Tree Strategy, Alice may choose not to disclose a syntactically relevant credential even though it is unlocked during a trust negotiation.

4.2 Protection of Possession-Sensitive Credentials

Suppose C_* is Alice’s possession-sensitive credential, and Carol is a user who does not possess C_* . Consider the following two examples:

Example 1 Let Bob, the server, have $R \leftarrow A1 \wedge C_*$. Suppose Alice has $A1 \leftarrow true$ and $C_* \leftarrow B1$, while Carol has $A1 \leftarrow B1$ and $C_* \leftarrow false$. Further, suppose Carol adopts the TrustBuilder-Relevant Strategy. For Alice, considering R ’s policy and the fact that $A1 \leftarrow true$, she may *dynamically migrate* the policy of C_* to $A1$ and make $A1$ ’s policy become $A1 \leftarrow B1$. Clearly, if Bob can satisfy this augmented policy of $A1$, then it is equivalent to satisfying the policy of C_* . Therefore, by disclosing $A1 \leftarrow B1$ to Bob, Alice preserves the chance to establish trust (to some extent, as we will see later). Meanwhile, her response is exactly the same as that of Carol. As described in section 4.1, since Carol adopts the TrustBuilder-Relevant Strategy, after receiving R ’s policy, she will reply with $A1 \leftarrow B1$, even though theoretically trust cannot be established. Thus, Bob cannot tell by observing her behavior whether Alice possesses C_* .

Example 2 Let $R \leftarrow A1 \vee C_*$. Suppose Alice has $A1 \leftarrow B1$ and $C_* \leftarrow B2$, while Carol has $A1 \leftarrow B1 \vee B2$ and $C_* \leftarrow false$. Because Carol adopts the TrustBuilder-Relevant Strategy, she will reply with $A1 \leftarrow B1 \vee B2$. For Alice, she may migrate the policy of C_* to $A1$ and make $A1 \leftarrow B1 \vee B2$. As in Example 1, disclosing $A1$ ’s augmented policy to Bob does not reduce the chance to have a successful negotiation. Meanwhile, Alice’s response is exactly the same as Carol’s, and Carol does not possess C_* .

The above two examples reflect the basic idea of our *policy migration* approach to protecting possession sensitive credentials:

1. Integrate the policy of a possession-sensitive credential into that of another relevant credential. And, meanwhile,
2. Preserve the potential success of the current trust negotiation. In other words, if theoretically Alice and

Bob can establish trust, then this must still be true after applying policy migration. More precisely, if there is a safe disclosure sequence for R under the original policies, then there must also be a safe disclosure sequence for R under the modified version of the policies. Further, if a successful trust negotiation is not possible, then after policy migration, both parties must eventually recognize the failure.

The first point allows Alice to “hide” the policy of C_* through policy migration and never directly disclose the policy of C_* , so that Bob cannot distinguish Alice’s response from that of other users. This is how Alice’s privacy is protected. The second point says that migration of the policy of C_* should not compromise the success or failure of a trust negotiation. Note that after we migrate C_* ’s policy to another credential C , C ’s disclosure is still controlled by its original policy \mathcal{P}_C . Its augmented policy is only for the purpose of disclosure to Bob so that C_* ’s policy does not need to be disclosed.

In the first example above, after the policy migration, i.e., after conjoining C_* ’s policy with that of $A1$, the potential success of the trust negotiation is not always preserved. For example, suppose Bob has policy $B1 \leftarrow A1$. Then there exists a safe disclosure sequence $(A1, B1, C_*, R)$. However, after Alice discloses $A1 \leftarrow B1$, from Bob’s point of view, he will find a circular dependency between $A1$ and $B1$, which may make him believe that there is no hope to establish trust, and terminate the negotiation prematurely.

On the other hand, the policy migration in the second example does preserve the potential success of the trust negotiation. The intuition is that, by changing $A1$ ’s policy to $A1 \leftarrow B1 \vee B2$, from Bob’s point of view, the policy has been weakened. Thus, if there is a safe disclosure sequence containing $A1$ in the original policies, we can still find such a sequence when we have $A1 \leftarrow B1 \vee B2$. Further, if there is a safe disclosure sequence containing C_* but not $A1$, then $B2$ must appear in the sequence. Thus, after Alice’s policy migration, Bob will see a safe disclosure sequence for $A1$. By following this sequence, after $B2$ is disclosed, Alice can disclose C_* instead of $A1$, which will also satisfy R ’s policy.

Formally, we have the following problem: Given $R \leftarrow \mathcal{P}$ and the policies for the credentials C_1, \dots, C_k, C_* that occur in \mathcal{P} , is there a way to assign new policies to C_1, \dots, C_k , such that by only disclosing to the other party these new policies, but not the policy of C_* , the potential success of a trust negotiation is preserved? We have the following theorem.

THEOREM 4.1. *Suppose C_* is possession-sensitive for Alice. Given $R \leftarrow \mathcal{P}$, let $C_1 \vee \dots \vee C_t$ be the disjunctive normal form of \mathcal{P} , where each C_i , $1 \leq i \leq t$, is a conjunction of credentials. If there exists C_i such that C_* does not appear in C_i , then there is a policy migration for C_* that will preserve the potential success of the trust negotiation.*

We call a disjunct an *independent disjunct* if C_* does not appear in it. The construction of the policy migration is as follows. For each credential C , we use \mathcal{P}_C to represent its policy. Given Bob’s policy \mathcal{P} , let \mathcal{P}' be a policy obtained by replacing each credential C in \mathcal{P} by \mathcal{P}_C . We call \mathcal{P}' the *mirror equivalent* of \mathcal{P} . Intuitively, if the credentials disclosed by Bob can satisfy \mathcal{P}' , then Alice will be able to disclose credentials that satisfy \mathcal{P} . (This intuition can also be used as the basis of another kind of negotiation strategy, as discussed in section 4.4.)

Without loss of generality, suppose disjuncts C_1, \dots, C_{i-1} involve C_* while C_i, \dots, C_t do not. Let \mathcal{P}' be the mirror equivalent of $C_1 \vee \dots \vee C_{i-1}$. We call \mathcal{P}' the *migrating component*. For each credential C appearing in C_i , we change its policy to $C \leftarrow \mathcal{P}_C \vee \mathcal{P}'$. We call $\mathcal{P}_C \vee \mathcal{P}'$ the *covering policy* of C , because it helps us to preserve the intent of the original policy of C_* . For C_* , its covering policy is *false*. For all the other credentials, their covering policies are the same as their original ones. We want to emphasize that the covering policy of C is not used to control C ’s actual disclosure. Alice’s disclosure of credential C is still controlled by her original policy \mathcal{P}_C .

As an example, suppose $\mathcal{P} = (A1 \wedge C_*) \vee (A2 \wedge A3)$. Then the covering policies will be $A1 \leftarrow \mathcal{P}_{A1}$, $C_* \leftarrow false$, $A2 \leftarrow (\mathcal{P}_{A2}) \vee ((\mathcal{P}_{A1}) \wedge (\mathcal{P}_{C_*}))$ and $A3 \leftarrow (\mathcal{P}_{A3}) \vee ((\mathcal{P}_{A1}) \wedge (\mathcal{P}_{C_*}))$.

Let $\mathcal{P}_{original}$ be the mirror equivalent of \mathcal{P} when the original policies of credential C_1, \dots, C_n are used, and let $\mathcal{P}_{covering}$ be the mirror equivalent of \mathcal{P} when the covering policies are used. In the above example, $\mathcal{P}_{original} = (\mathcal{P}_{A1} \wedge \mathcal{P}_{C_*}) \vee (\mathcal{P}_{A2} \wedge \mathcal{P}_{A3})$ and $\mathcal{P}_{covering} = (\mathcal{P}_{A2} \vee (\mathcal{P}_{A1} \wedge \mathcal{P}_{C_*})) \wedge (\mathcal{P}_{A3} \vee (\mathcal{P}_{A1} \wedge \mathcal{P}_{C_*}))$. It is not hard to prove that after the above policy migration, $\mathcal{P}_{original} \equiv \mathcal{P}_{covering}$.

After policy migration, Alice simply discloses to Bob the covering policies of all the credentials appearing in R ’s policy except that of C_* . Such a response is exactly the same as what Carol, who does not possess C_* , will do when the TrustBuilder-Relevant strategy is used. Therefore, by observing Alice’s response, Bob cannot tell whether Alice possesses C_* or not. Meanwhile, the way that we construct the policy migration guarantees that the potential success of trust negotiation is preserved, as will be proved below. Therefore, we will not miss any chance to establish trust.

PROOF. We need to prove that the above policy migration preserves the potential success of a trust negotiation, i.e., there exists a safe disclosure sequence under the original policies if and only if there also exists a safe disclosure sequence under the covering policies.

Suppose there exists a safe disclosure sequence $G = (C_1, \dots, C_k = R)$ according to the original policies. Without loss of generality, we assume G is a *minimal sequence*, i.e., by removing any credentials from the sequence, the resulting sequence is not a safe disclosure sequence any more. If C_* does not appear in G , then G is also a safe disclosure sequence according to the covering policies, because for any credential $C \neq C_*$, C ’s original policy entails its covering policy. Otherwise, if $C_i = C_*$ is in G , since we assume C_* only appears in R ’s policy, we can move C_* to the position right before R and the sequence is still safe. Thus, we assume $C_{k-1} = C_*$. Since G is a minimal sequence, before R is disclosed, a minimal solution for R ’s policy that contains C_* must have been disclosed. According to the way covering policies are constructed, before C_* is disclosed, the covering policies of those credentials appearing in the independent disjunct C_i are satisfied by $\{C_1, \dots, C_{k-2}\}$. Therefore, we can replace C_* in G with the credentials in C_i if they are not in the sequence yet, which will give us a safe disclosure sequence under the covering policies.

On the other hand, suppose there exists a minimal safe disclosure sequence G according to the covering policies. Then we need to prove that there also exists a safe disclosure sequence according to the original policies. Before R is disclosed, a minimal solution $\{C'_1, \dots, C'_q\}$ for R ’s policy must have been disclosed in the sequence. For each C'_j ,

$1 \leq j \leq q$, whose covering policy is not the same as its original policy, its covering policy is $\mathcal{P}_{C'_j} \vee \mathcal{P}'$, where \mathcal{P}' is the migrating component and $\mathcal{P}_{C'_j}$ is the original policy of C'_j . When C'_j is disclosed in the sequence, if $\mathcal{P}_{C'_j}$ is satisfied, then we still keep C'_j in the sequence. Otherwise, the migrating component \mathcal{P}' must have been satisfied by credentials disclosed before C'_j . According to the way \mathcal{P}' is constructed, there must be at least one conjunct C_l of R 's policy where C_* appears in C_l and all the policies of credentials occurring in C_l are satisfied by credentials disclosed before C'_j . Suppose $C_j = A1 \wedge \dots \wedge At \wedge C_*$. Then, the sequence $(C'_1, \dots, C'_{i-1}, A1, \dots, At, C_*, R)$, is a safe disclosure sequence according to the original policies. \square

What happens if C_* appears in every minimal solution for R 's policy? Example 1 shows a possible way — migrating the policy of C_* to credentials in the same minimal solution through conjunction. In detail, for each minimal solution $\{C_1, \dots, C_k, C_*\}$, Alice chooses one credential C_i and makes its covering policy $C_i \leftarrow \mathcal{P}_{C_i} \wedge \mathcal{P}_{C_*}$, while the covering policies of credentials that are never chosen are the same as their original ones. Alice discloses all the other credentials' covering policies to Bob but not that of C_* ($C_* \leftarrow false$).

However, this approach cannot always preserve the potential success of a trust negotiation. Intuitively, by putting the policy of C_* in conjunction with a credential C 's policy and disclosing it to Bob, the policy of C is strengthened from Bob's point of view. Thus, as is possible in Example 1, Bob may find that there is no hope to establish trust according to the policies received from Alice, while he would conclude the opposite according to the original policies. In this situation, there is essentially a tradeoff between privacy protection and trust establishment. In practice, to minimize the change to the potential success of a trust negotiation, we should migrate the policy of C_* to the policies of as few credentials as possible. Although potential success will not always be preserved, still the parties are more likely to establish trust under this approach than they would be if Alice simply tells Bob $C_* \leftarrow false$ when Bob asks for her possession-sensitive credentials.

PROPOSITION 4.1. *Let C_* be a possession-sensitive credential and suppose $R \leftarrow A1 \wedge C_*$. Suppose there exists a safe disclosure sequence $(C_1, \dots, C_i = A1, C_{i+1} = C_*, C_{i+2} = R)$. Then after migrating the policy of C_* to $A1$ such that $A1 \leftarrow \mathcal{P}_{A1} \wedge \mathcal{P}_{C_*}$ and $C_* \leftarrow false$, there is still a safe disclosure sequence for $A1$.*

PROOF. Since $A1$ and C_* are both Alice's credentials, $A1$ is not in any minimal solution for the policy of C_* . Therefore, the set of credentials $\{C_1, \dots, C_{i-1}\}$ is a solution for both C_* and $A1$. Thus, it is also a solution for $\mathcal{P}_{C_*} \wedge \mathcal{P}_{A1}$. So the sequence $(C_1, \dots, C_i = A1)$ is a safe disclosure sequence leading to $A1$'s disclosure according to the new policy of $A1$. \square

During a trust negotiation, after Alice discloses the covering policy of $A1$ to Bob, once the covering policy is satisfied by Bob, Alice can disclose both $A1$ and C_* to Bob and gain access to R consequently.

PROPOSITION 4.2. *Let $A1, \dots, At, C_*$ be a minimal solution for R 's policy, where C_* is a possession-sensitive credential. Suppose there exists a safe disclosure sequence*

$(C_1, \dots, C_i = C_, C_{i+1} = Aq, C_{i+2} = R)$. Then after migrating the policy of C_* to produce $Aq \leftarrow \mathcal{P}_{Aq} \wedge \mathcal{P}_{C_*}$, there is still a safe disclosure sequence for R according to the new policies. \square*

The proof of proposition 4.2 is similar to that of proposition 4.1, and we omit it here.

When there is more than one possession-sensitive credential in R 's policy, as long as there is a disjunct in R 's policy that does not involve any possession-sensitive credential, we can still preserve the potential success of a trust negotiation through policy migration by using a similar approach. In detail, let C_1, \dots, C_k be disjuncts involving at least one possession-sensitive credential and let C_{k+1} be independent of all possession-sensitive credentials. Then the migrating component \mathcal{P}' should be the mirror equivalent of $C_1 \wedge \dots \wedge C_k$. For each credential C in C_{k+1} , we set its covering policy to be $C \leftarrow \mathcal{P}_C \vee \mathcal{P}'$. The proof of preservation of potential success is similar to that of theorem 4.1.

Similarly, the possession-sensitive credential C_* does not have to appear in R 's policy. The theorems presented in this section will still hold whenever Alice's possession-sensitive credentials occur in only one policy \mathcal{P} that is relevant to the negotiation. In that case, Alice needs to find an independent disjunct in \mathcal{P} to migrate C_* 's policy to, subject to the condition that at least one credential in that disjunct has not already had its policy disclosed. If no such credentials are available, then policy migration cannot be used.

We will discuss the situation where possession-sensitive credentials appear in multiple policies, possibly spread across both parties, in a later section.

4.3 Protection of Non-Possession Sensitive Credentials

Recall that if C_* is a non-possession sensitive credential for Alice, then we have the following scenario: Alice has $C_* \leftarrow false$ and a policy \mathcal{P}_{C_*} to control to whom she is willing to disclose the denial policy for C_* . In other words, Alice will admit the fact that she does not possess C_* only if Bob has satisfied \mathcal{P}_{C_*} . Meanwhile, in the worst case, all the other users have $C \leftarrow true$. Clearly, if C_* is the only credential required to gain access to R , Alice cannot effectively protect the fact that she does not possess C_* , because all other users will disclose C_* immediately. Any other action Alice takes will reveal this sensitive fact. (Alice may intentionally unplug her connection with the Internet and make Bob believe that the negotiation is interrupted because of a failure of the network. Such a physical approach is out of the scope of this paper.)

When there are more credentials involved in R 's policy, the protection of non-possession sensitive credentials is very similar to that of possession-sensitive credentials. We try to migrate policy \mathcal{P}_{C_*} to the policies of other credentials so that Alice will behave the same as certain users who do have C_* and adopt the Disclosure Tree Strategy. Let us look at the following examples.

Suppose $R \leftarrow A1 \wedge C_*$. Obviously, since Alice does not have C_* , there is no way that trust can be established. Alice can make the covering policy of $A1$ be $A1 \leftarrow \mathcal{P}_{C_*}$ and disclose it to Bob. From Alice's response, Bob cannot infer that Alice does not possess C_* , since a user adopting the Disclosure Tree Strategy may make the same response. If Bob cannot satisfy policy \mathcal{P}_{C_*} , Alice will naturally terminate the negotiation. Otherwise, Alice can tell Bob that

she does not possess C_* , since Bob is qualified to know the truth. In either case, Bob cannot infer the sensitive fact from Alice’s behavior.

On the other hand, suppose $R \leftarrow A1 \vee C_*$. Alice can migrate \mathcal{P}_{C_*} to the policy of $A1$ such that $A1 \leftarrow \mathcal{P}_{A1} \vee \mathcal{P}_{C_*}$, and disclose it to Bob. Note that Bob cannot infer that Alice does not have C_* based on the fact that Alice does not disclose C_* , because a party adopting the Disclosure Tree Strategy will not necessarily disclose every unlocked and relevant credential. Later on, if Bob has satisfied \mathcal{P}_{A1} , then Alice may disclose $A1$ and gain access to R thereafter. If Bob has satisfied \mathcal{P}_{C_*} , similar to the above example, Alice may tell Bob that she does not possess C_* . The problem comes when Bob indicates that he can satisfy neither \mathcal{P}_{A1} nor \mathcal{P}_{C_*} . Now the situation boils down to the same as when C_* is the only credential required to gain access to R , a case that cannot be effectively handled in the current approach to trust negotiation.

In general, we have the following theorem.

THEOREM 4.2. *Suppose C_* is non-possession sensitive for Alice. If $\{C_*\}$ is not a minimal solution for Bob’s policy for R , then there is a policy migration for \mathcal{P}_{C_*} that satisfies the following conditions:*

1. *If there is a safe disclosure sequence for R under the original policies, then there is also one under the covering policies.*
2. *If there is a safe disclosure sequence $G = (C_1, \dots, C_k, R)$ under the covering policies, then either there is also one under the original policies, or \mathcal{P}_{C_*} is satisfied by $\{C_1, \dots, C_k\}$. In other words, after C_k is disclosed in the sequence, it is safe for Alice to tell Bob that she does not possess C_* .*

PROOF. Suppose $R \leftarrow \mathcal{P}$. Let $C_1 \vee \dots \vee C_t$ be the disjunctive normal form of \mathcal{P} , where each C_i , $i = 1, \dots, t$, is a disjunct. The construction of the policy migration is as follows. For each C_i that involves C_* , we choose one credential $C \neq C_*$ from C_i and make $C \leftarrow \mathcal{P}_C \vee \mathcal{P}_{C_*}$ the covering policy for C . For all the other credentials, their covering policies are the same as their original ones. Let $\mathcal{P}_{original}$ be the mirror equivalent of \mathcal{P} when the original policies of Alice’s credentials are used and let $\mathcal{P}_{covering}$ be the mirror equivalent of \mathcal{P} when the covering policies are used. It is easy to see that $\mathcal{P}_{covering} \equiv \mathcal{P}_{original} \vee \mathcal{P}_{C_*}$.

Clearly, for any credential $C \neq C_*$, its covering policy is weaker than or the same as its original policy, i.e., if C ’s original policy can be satisfied by Bob, so can its covering policy. Therefore, if there is a safe disclosure sequence G according to Alice’s original policies, then after the above policy migration, G is also a safe disclosure sequence according to the covering policies.

On the other hand, suppose that according to the disclosed covering policies, there exists a safe disclosure sequence G . Since Alice never discloses the policy of C_* , G does not involve C_* . Let C_i be a disjunct of R ’s policy that was satisfied before R is disclosed in G . If the original policy of every credential C in C_i is satisfied before C is disclosed, then G is also a safe disclosure sequence according to the original policies of Alice. Otherwise, according to the way that covering policies are constructed, before R is disclosed, \mathcal{P}_{C_*} must be satisfied. \square

When there is more than one non-possession sensitive credential in R ’s policy, as long as there is no disjunct in the disjunctive normal form of R ’s policy that contains only non-possession sensitive credentials, we can still handle it in a similar way. We will omit the details of this extension in this paper.

Similarly, the possession-sensitive credential C_* does not have to appear in R ’s policy. The theorems presented in this section will still hold whenever Alice’s possession-sensitive credentials occur in only one policy that is relevant to the negotiation (as long as there is a suitable disjunct available). However, as discussed above, it is possible that the migration will not preserve the potential success of the negotiation.

4.4 Discussion

One potential alternative for protecting possession-sensitive credentials is to require each party to always disclose the mirror equivalent of R ’s policy. That is, after Bob discloses R ’s policy \mathcal{P} , Alice only discloses \mathcal{P} ’s mirror equivalent \mathcal{P}' , without telling Bob the policy of each individual credential appearing in \mathcal{P} . Similarly, Bob will disclose the mirror equivalent of \mathcal{P}' in his next message. This process continues until one party finds that either the current policy can be satisfied by a set of unprotected credentials or there is no solution to the current policy. Such a negotiation process is called a parsimonious negotiation strategy, first introduced by Winsborough et al. in [27].

Policy migration has several advantages over this approach. By disclosing the policy of each individual credential separately, a party can control the direction of the progress of a trust negotiation according to its own preference. For example, if there are multiple ways in which Alice might be able to satisfy Bob’s policy for R , she might clearly prefer one set of disclosures over another. Most families of negotiation strategies include strategies that will allow Alice to steer the direction of the negotiation toward what she views as the most desirable disclosure sequence. This flexibility and autonomy is desirable in decentralized systems. Further, in the worst case, the size of the mirror equivalent of R ’s policy may increase exponentially with the number of rounds of messages, which will result in high communication cost. This problem is also avoided by allowing the disclosure of the policy of each individual credential.

Compared to ack policies and non-response, policy migration has the following advantages:

1. Policy migration is very easy to deploy in current trust negotiation systems. It does not require any new extensions to the components of the existing trust negotiation model, e.g., the organization of policies, the format and semantics of messages, or the information allowed to be exchanged. Also, its management overhead is low. Unlike ack policies and non-response proposed in [26, 25], a user only needs to specify what credentials are considered possession-sensitive or non-possession sensitive. The major protection is supplied by the security agent using a policy migration algorithm, which is simple and easy to implement in the situation discussed in this paper.
2. Policy migration is self-contained. It does not require any pre-arranged cooperation from a large group of users. The protection power of policy migration comes from the fact that Alice will behave in the same way as

certain ordinary users who do not have Alice’s possession-sensitive credential, or do have Alice’s non-possession sensitive credential. Bob cannot rely on observing an unusual response from Alice to infer whether she has a specific possession-sensitive credential, or does not possess a non-possession sensitive credential.

3. Policy migration provides a nice balance between the potential success of a trust negotiation and the protection of possession-sensitive and non-possession sensitive credentials. In most cases, the potential success of a trust negotiation is preserved when we migrate the constraints in the policy of a possession-sensitive or non-possession sensitive credential. Therefore, success of a trust negotiation is still guaranteed. Specifically, we have identified a large group of policies under which policy migration can always preserve the potential success of a trust negotiation.

The version of policy migration presented in this paper has limitations that need to be addressed in future work. In particular, policy migration needs to be extended to handle the case where possession-sensitive and non-possession sensitive credentials appear in multiple policies belonging to multiple parties.

In general, if Alice’s possession and non-possession sensitive credentials are mentioned in many policies that are relevant to the negotiation, then policy migration can still be used to protect her privacy. However, in this situation policy migration can become more complex. Alice must do book-keeping to remember what covering policies she has used where, and reason more carefully about when to disclose a possession-sensitive credential or to tell Bob the fact that she does not possess a non-possession sensitive credential. To make this happen, Alice’s negotiation software may need to use a highly accurate representation of the progress of a trust negotiation, such as a version of the disclosure trees [30] originally developed for use in proofs of correctness of the interactions between the strategies used by two negotiating parties.

Another interesting question is how to generalize policy migration to handle the situation where both Alice and Bob have possession or non-possession sensitive credentials involved in a trust negotiation. Intuitively, as long as the covering policies of Bob and Alice do not strengthen the semantics of their corresponding original policies, the potential success of the trust negotiation will not be jeopardized and their privacy will be protected. However, the higher the number of sensitive credentials involved in the negotiation, the more management effort will be required. Mechanisms are needed to manage policy migration so that complex situations such as this can be handled efficiently.

Even with the extensions described above, policy migration will not be a complete solution to the problem of protecting possession-sensitive and non-possession sensitive credentials.

1. Policy migration is designed to prevent inferences through simple observation of one’s *behavior* during trust negotiation, such as one might expect from an automated attack program. Therefore, the human-intelligible meaning of policies is not considered. In practice, typically, when a credential is possession-sensitive, its policy tends to be very specific and strict. For example,

Alice may specify that only people working for the NSA can see her CIA agent card. If we migrate such a constraint to Alice’s driver’s license, then its covering policy may require one to be either a police officer or be working for the NSA, which is clearly an unusual policy for one’s driver’s license. In fact, it is well known that the content of an access control policy may reveal the sensitive nature of a credential. Several approaches have been proposed to protect sensitive policies [7, 22, 29]. By combining policy migration with sensitive policy protection schemes, we may get a more comprehensive solution.

2. Policy migration is subject to *comparison attacks*. Bob may negotiate trust with Alice on several occasions. Suppose that during a previous trust negotiation where her CIA agent card is not involved at all, Alice discloses her driver’s license’s policy to Bob. Later, during a more sensitive trust negotiation which involves her CIA agent card, Alice may migrate the policy for her CIA agent card to that of her driver’s license. If Bob maintains a copy of Alice’s original driver’s license’s policy, he will be able to guess that the one received later is a covering policy, which can be used to infer that Alice is actually connected to the CIA. Such an attack is hard to prevent because it is unlikely for Alice to remember each of her interactions with all other parties. Further, even if Alice does remember, or has never interacted with Bob before, Bob may still get a copy of Alice’s original policy from a third party Carol, who negotiated trust with Alice before. As mentioned in section 2, it is difficult for Alice to control the propagation of her private information once it has been disclosed to others.

Overall, the effect of policy migration is to make inferences regarding possession-sensitive and non-possession sensitive credentials harder instead of making them impossible. In the original model of trust negotiation, without proper protection, inference is quite simple and can easily be automated. By using policy migration, one has to analyze the semantics of a policy or keep track of a user’s policies before the inference can succeed. This process requires more human intervention, which makes it more difficult to be conducted by computers in an efficient way.

Inference prevention is widely known as a hard problem. Even for a database with a simple query-answering model, it is still very difficult to find all the possible ways to infer sensitive information from seemingly reasonable queries, not to mention preventing those inferences. This is especially true when we have to consider the semantics of queries along with attackers’ background knowledge [5]. The problem can only get harder in trust negotiation because of the existence of multiple channels for information flow. Next we list some other ways that one’s privacy may be violated in ATN, some of which have been partially addressed in the literature.

Sensitive policies The original model of trust negotiation [20, 21, 30] assumes that the appropriate access control policies can be shown to Bob when he requests access to Alice’s resources. However, realistic access control policies tend to contain sensitive information, because the details of Alice’s policy \mathcal{P} for disclosure of a credential C tends to give hints about C ’s contents. For example, if there is any information in C

that is very sensitive, then it is possible to guess at the nature of that information by looking at \mathcal{P} : is her parole officer allowed to see C ? Her welfare case worker? The local HIV, cancer, or mental health clinic? More generally, a company’s internal and external policies are part of its corporate assets, and it will not wish to indiscriminately broadcast its policies in their entirety. Several schemes have been proposed to protect the disclosure of sensitive policies. In [7], Bonatti and Samarati suggested to divide a policy into two parts — prerequisite rules and requisite rules. The constraints in a requisite rule will not be disclosed until those in prerequisite rules are satisfied. In [22], Seamons et al. proposed to organize a policy into a directed graph so that constraints in a policy can be disclosed gradually. In [29], access control policies are treated as first-class resources, thus can be protected in the same manner as services and credentials.

Private information propagation Once a trust negotiation is finished, no matter whether it is successful or not, Alice has gathered some information about Bob. After this point, it is hard for Bob to control how the gathered information will be used by Alice. Alice may actively initiate trust negotiation with others with the purpose of collecting information instead of establishing trust. She can later either sell the information to an online marketing company or simply make it public. Though the P3P standard [24] enables an individual to be aware of the privacy practice policy of a server before disclosing her sensitive information, it is difficult to check how well that policy is enforced. Also, trust negotiation makes it possible for any individual to be an information collector, which causes P3P to be not quite applicable.

Many researchers have observed that the use of unlinkable pseudonymous credentials and selective attribute disclosure can alleviate this problem [8, 9, 15]. However, as long as Alice collects information about Bob, be it through simple credential disclosures or zero-knowledge techniques, there is no effective way to prevent Alice from further spreading that information.

Besides specifying P3P policies, it is becoming an increasing concern for companies and organizations to effectively *enforce* privacy practice policies. In [2], Agrawal et al. presented a server-centric approach to enforce P3P policies through query translation and rewriting in databases. Agrawal et al. [1] also proposed principles for designing databases that enforce a company’s privacy policies. Karjoth et al. [18] proposed a privacy-centric access control language and designed an architecture for privacy policy enforcement in the entire life cycle of customers’ information, based on the principle of separation of duty.

Information gathering and analysis Once Alice has gathered enough information from Bob, she may be able to infer more sensitive information about Bob through the seemingly insensitive information. For example, if Alice knows Bob’s address, which is located near the center of the campus of a university, then she may reasonably infer that Bob is a student in the university and his age is probably in the range of 18 to 30.

Such inferences will be much more powerful when information from multiple sources or credentials is collected. Inference prevention techniques developed by the database community, such as those proposed in [4, 5, 10, 11], can be applied. However, many of those techniques require tracing a user’s knowledge of an entity in order to effectively prevent inference. This is especially hard in the context of trust negotiation, where Alice and Bob are usually from different organizations and do not have identities in the same security domain. Further, Alice may assume identities from different domains during different interactions with Bob, which makes it hard for Bob to tell whether he is interacting with the same person during different trust negotiation. It is also impractical for an individual to keep track of all his interactions with others.

Recently much work has been done on mutual authentication and authorization through the use of cryptographic techniques that offer improved privacy guarantees. For example, Balfanz et al. [3] designed a secret-handshake scheme where two parties reveal their memberships in a group to each other if and only if they belong to the same group. Li et al. [19] proposed a mutual signature verification scheme to solve the problem of cyclic policy interdependency in trust negotiation. Under their scheme, Alice can see the content of Bob’s credential signed by a certification authority CA only if she herself has a valid certificate also signed by CA and containing the content she sent to Bob earlier.

Policy migration is complementary to this prior work on secret handshakes. For example, in the work by Balfanz et al., mutual authorization can only happen between two parties that belong to the same group, i.e., their credentials are issued by the same certification authority. The mutual signature verification scheme devised by Li et al. has a similar constraint. In trust negotiation, this constraint will often not be met. Trust negotiation relies on more general access control policies, where mutual authorization between two parties is achieved through the use of credentials from different security domains (signed by different certification authorities). The above works also do not address the question of how the two parties learn about each other’s authorization requirements (access control policies), an issue which is a major concern in this paper. Researchers have begun to work on removing these and other limitations of secret handshakes [14], so that they may become another general-purpose weapon against unwanted leaks of information during trust negotiation.

5. CONCLUSION

During trust negotiation, trust is established through iterative and bilateral disclosures of credentials and access control policies between two parties. During this process, information can flow to others in a variety of forms, some of which are not well protected by access control policies in the original model for trust negotiation. Two types of information leaks have been identified where sensitive information can be easily inferred, even though access control policies are strictly enforced. By observing Alice’s behavior during a trust negotiation, an attacker can infer whether some attributes of Alice satisfy certain constraints or whether Alice possesses or does not possess certain sensitive credentials. We call credentials vulnerable to the first kind of in-

ference attribute-sensitive credentials and those vulnerable to the second kind of inference possession-sensitive and non-possession sensitive credentials.

In this paper, we focus on the protection of possession-sensitive and non-possession sensitive credentials. We propose desiderata for good protection schemes. In particular, we argue that the protection scheme should not only be able to prevent inference but also be able to guarantee the success of trust establishment as much as possible. Also, the management overhead introduced by the protection scheme should be low. Otherwise, its deployment in practice will be limited.

We propose policy migration as an approach to protecting sensitive credentials. In this approach, the policy of a possession-sensitive or non-possession sensitive credential is integrated into those of other credentials, so that Alice's behavior will be the same as other users whether or not she has the sensitive credential. We identify under what circumstances policy migration can preserve the potential success of a trust negotiation, in the sense that the chance of a successful trust negotiation will not be jeopardized by policy migration.

Admittedly, inference prevention is a hard problem, especially in the context of trust negotiation, where multiple rounds of information exchange take place between the two parties. We have also analyzed the limitations of policy migration. In particular, an attacker may be able to figure out whether policy migration happens, either by considering the semantics of a disclosed access control policy or by comparing policies for the same credential that were received during different sessions of trust negotiation.

In order to systematically study inference prevention in trust negotiation, we believe that a strict theoretical foundation needs to be laid so that information flow between two negotiation participants can be formally modeled and analyzed. That is one direction we would like to further pursue in our future work. We are also interested in designing new interaction models for trust negotiation, where inferences cannot be easily conducted by an attacker.

6. ACKNOWLEDGMENTS

This work was done when Ting Yu was in the University of Illinois. This research was sponsored by DARPA through Space and Naval Warfare Systems Center San Diego grant number N66001-01-18908 (BYU) and AFRL contract numbers F33615-01-C-1805 (BYU) and F30602-97-C-0336 (NAI Labs). We also thank Paul Syverson and the anonymous reviewers for their helpful comments.

7. REFERENCES

- [1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic Databases. In *28th International Conference on Very Large Data Bases*, Hong Kong, Aug. 2002.
- [2] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Implementing P3P Using Database Technology. In *19th International Conference on Data Engineering*, Bangalore, Mar. 2003.
- [3] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H. Wong. Secret Handshakes from Pairing-Based Key Agreements. In *IEEE Symposium on Security and Privacy*, Berkeley, CA, May 2003.
- [4] J. Biskup and P. Bonatti. Lying Versus Refusal for Known Potential Secrets. *Data & Knowledge Engineering*, 38(2), 2001.
- [5] J. Biskup and P. Bonatti. Controlled Query Evaluation for Known Policies by Combining Lying and Refusal. In *International Symposium on Foundations of Information and Knowledge Systems*, Salzaau Castle, Germany, Feb. 2002.
- [6] M. Blaze, J. Feigenbaum, and A. D. Keromytis. KeyNote: Trust Management for Public-Key Infrastructures. In *Security Protocols Workshop*, Cambridge, UK, 1998.
- [7] P. Bonatti and P. Samarati. Regulating Service Access and Information Release on the Web. In *Conference on Computer and Communications Security*, Athens, Nov. 2000.
- [8] S. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. The MIT Press, 2000.
- [9] J. Camenisch and E. Herreweghen. Design and Implementation of the Idemix Anonymous Credential System. In *ACM Conference on Computer and Communication Security*, Washington D.C., Nov. 2002.
- [10] L. Chang and I. Moskowitz. An Integrated Framework for Database Privacy Protection. In *14th IFIP WG11.3 Working Conference on Data and Application Security*, Amsterdam, Aug. 2000.
- [11] I. Dinur and K. Nissim. Revealing Information while Preserving Privacy. In *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, San Diego, CA, June 2003.
- [12] A. Herzberg, J. Mihaeli, Y. Mass, D. Naor, and Y. Ravid. Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2000.
- [13] A. Hess, J. Jacobson, H. Mills, R. Wamsley, K. Seamons, and B. Smith. Advanced Client/Server Authentication in TLS. In *Network and Distributed System Security Symposium*, San Diego, CA, Feb. 2002.
- [14] J. Holt, R. Bradshaw, K. Seamons, and H. Orman. Hidden Credentials. In *ACM Workshop on Privacy in the Electronic Society*, Washington, DC, Oct. 2003.
- [15] R. Jarvis. Selective Disclosure of Credential Content during Trust Negotiation. Master's thesis, Depart. of Computer Science, Brigham Young University, Apr. 2003.
- [16] T. Jim. SD3: A Trust Management System with Certified Evaluation. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2001.
- [17] W. Johnson, S. Mudumbai, and M. Thompson. Authorization and Attribute Certificates for Widely Distributed Access Control. In *IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 1998.
- [18] G. Karjoth, M. Schunter, and M. Waidner. Platform for Enterprise Privacy Practices: Privacy-enabled Management of Customer Data. In *2nd Workshop on Privacy Enhancing Technologies*, San Francisco, CA, Apr. 2002.

- [19] N. Li, W. Du, and D. Boneh. Oblivious Signature-Based Envelope. In *ACM Symposium on Principles of Distributed Computing*, Boston, MA, July 2003.
- [20] N. Li, J. Mitchell, and W. Winsborough. Design of A Role-based Trust-management Framework. In *IEEE Symposium on Security and Privacy*, Berkeley, California, May 2002.
- [21] N. Li, W. Winsborough, and J. Mitchell. Distributed Credential Chain Discovery in Trust Management. *Journal of Computer Security*, 11(1), Feb. 2003.
- [22] K. Seamons, M. Winslett, and T. Yu. Limiting the Disclosure of Access Control Policies during Automated Trust Negotiation. In *Network and Distributed System Security Symposium*, San Diego, CA, Feb. 2001.
- [23] K. Seamons, M. Winslett, T. Yu, L. Yu, and R. Jarvis. Protecting Privacy during On-line Trust Negotiation. In *2nd Workshop on Privacy Enhancing Technologies*, San Francisco, CA, Apr. 2002.
- [24] W3C, <http://www.w3.org/TR/WD-P3P/Overview.html>. *Platform for Privacy Preferences (P3P) Specification*.
- [25] W. Winsborough and N. Li. Protecting Sensitive Attributes in Automated Trust Negotiation. In *ACM Workshop on Privacy in the Electronic Society*, Washington, DC, Nov. 2002.
- [26] W. Winsborough and N. Li. Towards Practical Automated Trust Negotiation. In *3rd International Workshop on Policies for Distributed Systems and Networks*, Monterey, California, June 2002.
- [27] W. Winsborough, K. Seamons, and V. Jones. Automated Trust Negotiation. In *DARPA Information Survivability Conference and Exposition*, Hilton Head Island, SC, Jan. 2000.
- [28] T. Yu. *Dynamic Trust Establishment in Open Systems*. PhD thesis, Department of Computer Science, University of Illinois, Sept. 2003.
- [29] T. Yu and M. Winslett. A Unified Scheme for Resource Protection in Automated Trust Negotiation. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2003.
- [30] T. Yu, M. Winslett, and K. Seamons. Supporting Structured Credentials and Sensitive Policies through Interoperable Strategies in Automated Trust Negotiation. *ACM Transactions on Information and System Security*, 6(1), Feb. 2003.