Privacy Policy Negotiation Framework for Attribute Exchange

Makoto Hatakeyama and Hidehito Gomi

Internet Systems Research Laboratories, NEC Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa 211-8666, Japan

Abstract

Service providers often incorporate other services from different service providers seamlessly. The service provider composing the service out of other services needs to exchange user attributes with the service providers it interacts with. Currently the exchange of user attributes among service providers based on the user's consent includes mostly all user attributes available. This approach is not sufficiently concerned with confirmation of privacy policies made by providers and users and henceforth does not consider the data minimization principle included in domestic privacy legislation. Therefore providers need a mechanism in order to satisfy the requirement that providers confirm privacy policies, determine a mutual agreement on attributes exchange and prevent leakage of privacy as well as respect legal aspect. In this paper, we propose a privacy policy negotiation framework, in which attributes are only exchanged after the interacting providers have confirmed what kinds of attributes to exchange and the way how to use and store them.

1. Introduction

Recently many service providers offer all kinds of services to users on the Internet. To provide richer values to the users, services often incorporate other services from different service providers seamlessly. A travel booking site for example provides besides the ordinary travel booking, car rental services or event and tour booking services. Thus, the service provider composing the service out of other services needs to exchange personal user attributes with the service providers he interacts with. Currently the exchange of user attributes among service providers based on the user's consent. When providers exchange attributes, they need to manage the exchanged attributes appropriately. For example, a service provider acting as an attribute sender needs to limit the attributes to be sent to other providers, because it is responsible for management of its users attributes and must not disclose more attributes than its users permit to send for prevention of privacy leakage as well as for conformance with domestic legislation in order to fulfill the concepts of user control and data minimization. Furthermore, a service provider acting as an attribute receiver needs to limit attributes to be received, since the management of many attributes causes a more expensive management and a higher risk of information leakage. Therefore, providers have to exchange only the necessary attributes they need to perform the provided tasks. Hence, a mechanism is required, which enables the service providers to agree on what attributes to exchange based on requirements and policies of the involved user, attribute sender and attribute receiver.

Previous work is not sufficiently concerned with confirmation method of privacy policies which describe attributes to be transferred between providers and management of attributes. Though providers have to confirm the requirements of a sender, a receiver and a user for attribute exchange, these providers currently have no means to confirm these requirements. In previous work, there are some variations of privacy policies descriptions [Myers 00] [P3P] [APPEL]. Myers et al. proposed an expressive model of decentralized information flow labels, which has the ability to express privacy policies. This method aims on the confirmation between a user and a provider statically. But, providers which exchange user attributes cannot confirm whether privacy policies satisfy the requirements of a sender, a receiver and a user or not. [P3P] and [APPEL] describe that policies consist of purpose, recipient, retention, data, access, disputes and remedies. A user and a website describe these policies and rules, and the website can determine its method of attribute management using policy matching rules defined by APPEL. Using these specifications, an agreement between a web site and a user can be made, but a protocol for the mutual agreement between providers is not concerned. Backes et al. also proposed a privacy policy comparison

[Backes 04]. This enables providers to check refinement of enterprise privacy policy efficiently. But when exchanging attributes between providers, providers cannot determine privacy policies to be followed. In order to establish the relation between untrusted providers, "Automated Trust Negotiation" protocol is proposed [Seamons 02] [Winsborough 04]. In these protocols, a provider determines exchanged attributes according to attributes which the provider receives. During the interaction, the provider collects attributes which are the base of trust and the provider establishes trust relation between the other providers.

The purpose of this paper is to introduce a means for providers to be able to confirm privacy policies before an attribute exchange takes actually place and to determine what kinds of attributes to exchange and how to manage these attributes. In order to realize the above purposes, we propose a privacy policy negotiation framework which satisfies the following three requirements: One is that the provider forwarding attributes selects only necessary attributes to be exchanged based on the user's policies and the receiving provider's policies. A second is that providers confirm a mutual agreement about the kinds of exchanged attributes and the management of attributes before exchange. The other is providers limit the disclosure of privacy policies, since the policies may include privacy information.

Using this privacy policy negotiation framework, providers and users specify privacy policies by themselves. Interacting providers then confirm what attributes to exchange and the policy of the attributes and lifecycle. The providers exchange attributes only after they judge that they can exchange attributes as a result of confirmation.

2. Privacy Policy Management

In this section, we described the proposed privacy policy negotiation framework in more detail. In this part of privacy policy negotiation framework, an attribute sender (***AS***) and attribute receiver (***AR***) describes its own privacy policies. For each user, *AS* and *AR* describe their own privacy policies.

In order to determine kinds of exchanged attributes and method of attribute usage and storage based on privacy policies, policy matching rules are needed. Therefore we specify privacy policies and policy comparison method.

2.1 Privacy Policy

In this framework, a privacy policy is defined as a set of statements about what kinds of attributes providers exchange, how they use and store those attributes. Privacy policy negotiation is a process of making a mutual agreement on shared privacy policies between providers. Providers have their own liabilities to exchange, use and store attributes based on the agreement.

Privacy policies are categorized into following three types:

One is the user policy $\boldsymbol{Pol_u}$ which is described by a user who is associated with these attributes and owns them. A user describes rules about what providers can use attributes and how they must use and store the attributes. A user makes a set of Pol_u for each attribute. This policy includes the user's consent for attribute exchange.

Another is the sender policy $\boldsymbol{Pol_s}$ managed by *AS*. *AS* determines rules about what kinds of attributes *AS* can send to other providers, to whom *AS* can send attributes and how *AR* must use and store the attributes. This policy is made by a responsible member of *AS* based on a relationship between *AS* and *AR*. For example, *AS* may contract with *AR* not to exchange a specific attribute. In this situation, even if a user permits to exchange this attribute, *AS* will not forward it to *AR*. In order for *AS* to determine the attributes to be sent, *AS* makes Pol_s which reflects the contracts with *AR*.

The other policy is the receiver policy $\boldsymbol{Pol_r}$ managed by *AR*. *AR* settles the rules about what kinds of attributes *AR* is required to receive, and how *AR* uses and stores the attributes. *AR* can describe some Pol_r because kinds of attributes *AR* requires and a way of usage and storage of attributes will change if the situation to use attributes differs. This policy is made by a responsible member of *AR* in order to limit attributes to be received for cost cut of attributes management and reducing the risk of privacy lekage by conforming to domestic legislation.

2.2 Policy Comparison

AS and *AR* should not exchange more attributes than they require. If *AR* obtained more attributes than it requires to perform the provided task, it has to manage too many attributes accurately resulting in higher costs than necessary. In order to the limit exchanged attributes, providers have to examine their policies mutually and determine usage and storage of attributes.

To exchange attributes between *AS* and *AR*, they must confirm that they follow the three types of policies introduced in the previous section. *AS* confirms whether these Pol_r satisfies Pol_u and Pol_s or not before sending any attributes. *AR* confirms policies in order for *AR* to reject the policies which *AR* does not require.

Before providers exchange attributes, they must confirm that policies satisfy the following condition:

$$Pol_r \subseteq (Pol_s \cap Pol_u). \cdots (1)$$

AS sends attributes to *AR*, only when *AS* and *AR* confirms that the three policies satisfy the condition (1). In cases, where *AS* does not manage any of the user attributes *AR* is requesting by Pol_r, thus the following condition holds

$$Pol_s \cap Pol_u = \phi. \cdots (2)$$

no attributes are send from A_s to A_r. A_r only receives attributes from A_s when the user permits the access and when condition (1) is satisfied.

3. Privacy Policy Negotiation

3.1 Trust Relation between Providers

In this policy negotiation framework, exchange of privacy policies and attributes premises on mutual trust between AS and AR. Providers disclose their policies and attributes based on the trust. Though AS trusts AR's management of exchanged attributes, each AS and AR has its own liability of attribute management. When AR receives attributes from AS, AR manages attributes by itself. Even if providers have trust relation, AS must not disclose attributes before AS and AR confirm policies and liabilities of management of attributes. A provider can prove its proper management by reference of the confirmation log.

3.2 Requirements for Privacy Policy Negotiation

AR has to send the only necessary Pol_r. If *AR* discloses all of Pol_r, *AS* would judge that *AR* requires all attributes written in the Pol_r and Pol_s would get more attributes than *AR* really requires. In order to limit received attributes, *AR* limits disclosure of Pol_r.

To reduce interactions of policy negotiation, AS sends privacy policies to *AR*. *AR* can get a cue to determine right Pol_r to send to *AS*. *AS* discloses the policies which describe the condition that *AS* can send attributes to *AR*. This policy (henceforth denoted as proposal policy $\boldsymbol{Pol_p}$) satisfies the following condition:

$$Pol_p = Pol_s \cap Pol_{u\_AR} \quad \cdots (3)$$

This Pol_p stands for the condition of *AR*'s usage and storage which *AS* and a user determine.

Pol_p may include user's privacy information, because Pol_p is related with user's attribute. *AS* does not disclose all of Pol_p at one time, but *AS* discloses a part of Pol_p. A privacy policy can be divided to some kinds of statements such as "Purpose", "Recipient" , "Data" and so on. In order to disclose partial policies, Pol_p is expressed as a set (4). *AS* does not send all of Pol_p to *AR*, but sends a part of the subset of Pol_p. *AS* can protect leakage of Pol_u because Pol_{pn} which *AS* discloses include part of Pol_u and *AR* cannot get all of Pol_u.

$$Pol_p = \{Pol_{p1}, Pol_{p2}, \cdots, Pol_{pn}\} \cdots (4)$$

AR can reduce unnecessary interactions for policy negotiation, because *AR* can receive a cue of accurate policies. *AR* can also divide Pol_r to a set of statements (5). If Pol_{rm} does not satisfy the condition (6), *AR* can understand that *AR* cannot receive user's attributes.

$$Pol_r = \{Pol_{r1}, Pol_{r2}, \cdots, Pol_{rn}\} \cdots (5)$$

$$Pol_{pm} \supseteq Pol_{rm} \cdots (6)$$

3.2 Privacy Policy Negotiation protocol

We propose the following privacy policy negotiation protocol to confirm that policies satisfy condition (1). This protocol is only concerned with policy exchanges. Providers determine attributes to be exchanged and the method of usage as well as attribute lifecycle through the policy negotiation protocol (see Figure 1). In this framework, *AS* has Pol_u, Pol_s and user's attributes. *AR* has Pol_r.

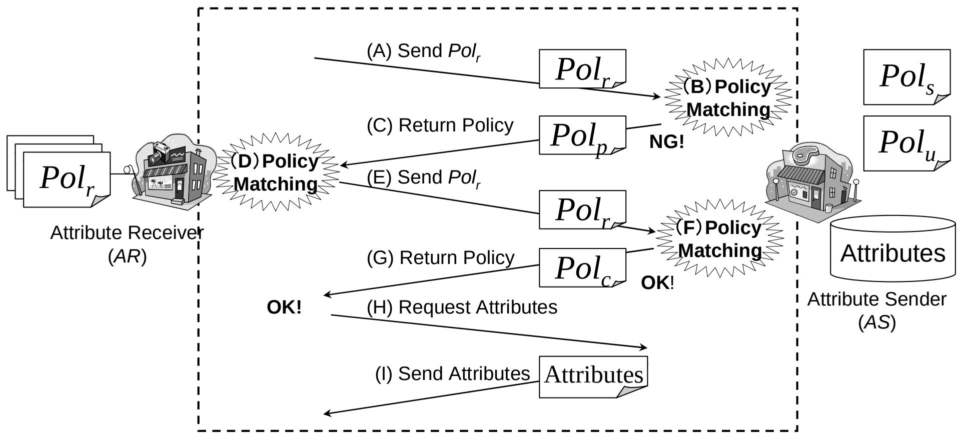

Figure 1: Privacy Policy Negotiation

In privacy policy negotiation, both *AS* and *AR* confirm whether the three policies satisfies the condition (1) and (6). For the two involved providers to agree on the attribute to be exchanged, *AR* discloses Pol_r to *AS* and *AS* discloses subset of Pol_p shown in condition (4).

Step A: *AR* sends Pol_r

The privacy policy negotiation protocol is initiated by *AR* sends Pol_r to *AS*.

Step B: *AS* compares Pol_r with Pol_u and Pol_s

After *AS* receives Pol_r, *AS* confirms that Pol_r satisfies condition (1) and decides the reaction of the Pol_r. If Pol_r satisfies condition (1), *AR* takes Step G. But if Pol_r does not satisfy condition (1), *AR* takes Step C.

Step C: Policy matching is failed

If Pol_r does not satisfy condition (1), *AS* sends *AR* an error message with a cue for successful policy matching. The cue is the subset of P_Policy (i.e., Pol_{pm}). Pol_{pm} is the subset of Pol_p which is shown by condition (3). Pol_p includes user information because Pol_p includes Pol_u which is information related with user information. Therefore *AS* does not disclose all of Pol_p, but discloses a part of Pol_p for protection of user information.

Step D: *AR* compares Pol_r with Pol_p

When *AR* receives Pol_{pm}, *AR* finds new Pol_r which satisfies condition (6). *AR* may receive many Pol_{pm} because *AR* sends Pol_r many times and receives Pol_{pm}. *AR* uses all of Pol_{pm} for search of Pol_r. If *AR* does not have Pol_r which satisfies the condition (6), *AR* terminates the negotiation and attribute exchange.

Step E: *AR* sends Pol_r

AR resends newly adapted Pol_r which is the result of policy comparison at Step D.

Step F: *AS* compares Pol_r with Pol_u and Pol_s

This is the same as Step B. If the received Pol_r does not satisfy condition (1), *AS* sends different Pol_{pm} which *AS* sent at Step C.

Step G: Policy matching is succeeded

If Pol_r satisfies condition (1), *AS* sends a confirmation policy $\boldsymbol{Pol_c}$ and a message which tells a termination of negotiation. *AS* and *AR* can deduct the responsibility they have for attribute management from Pol_c. Pol_c satisfies the following condition:

$$Pol_c = Pol_r \subseteq (Pol_s \cap Pol_u). \cdots (7)$$

Pol_c is a mutual agreement about attribute management between *AS* and *AR*, because Pol_c includes all of the three types of privacy policies.

Step H and I: Attribute exchange

AR sends an attribute request after *AR* confirms Pol_c. To make a mutual agreement to use and store of attributes between *AS* and *AR*, *AR* confirms Pol_c. The exchange of attributes can be based on other already existing protocols such as Liberty ID-WSF [ID-WSF] and SAML [SAML].

4. Discussion

Providers can exchange the only necessary attributes, because privacy policies satisfy the condition (1). Using this privacy policy negotiation protocol, both a user and providers can limit exchanged attributes based on their requirements written in privacy policies.

The presented privacy policy negotiation framework enables providers to confirm privacy policies and liabilities that providers should follow. If providers preserve a record of privacy policies that they confirmed during policy negotiation before attribute exchange, they do not have to negotiate again. If all of Pol_u, Pol_s and Pol_r do not change, the result of negotiation does not change, so that providers can confirm a record of negotiation result in stead of negotiation again.

AS can protect privacy information in Pol_u, because *AS* sends only a subset of Pol_p. At Step B in Figure 1, *AS* discloses the subset of Pol_p which includes Pol_u, hence *AR* hardly guesses all of Pol_p. On the other hand *AS* discloses a new distinct subset of Pol_p at Step F, only if *AS* can confirm that *AR* follows the subset of Pol_p which *AS* sent in Step C.

5. Conclusions

This paper has proposed a framework for privacy policy negotiation and management for attribute exchange. We have categorized privacy policies into three types and defined the condition which policies must satisfy for attribute exchange. We have also specified a policy negotiation protocol, which enables providers to determine kinds of exchanged attributes and to confirm liabilities of how to use and store the attributes before exchange. We intend to investigate the audit trail of policy negotiation to manage attributes properly in the future.

6. Acknowledgement

The authors would like to thank Luigi Lo Iacono for his comments on an earlier version of this paper.

References

[Myers 00] A. Myers and B. Liskov. *Protecting Privacy using the Decentralized Label Model.* ACM Transactions on Software Engineering and Methodology, Volume 9 No.4, pages 410-442, 2000.
[P3P] W3C. *The Platform for Privacy Preferences 1.0 (P3P1.0).* W3C Recommendation, April 2002. Available online at: http://www.w3.org/TR/P3P/
[APPEL] W3C. *A P3P Preference Exchange Language 1.0 (APPEL1.0).* W3C Working Draft, April 2002. Available online at: http://www.w3.org/TR/P3P-Preferences/
[Backes 04] M.Backes, W.Bagga, G.Karjoth, and M.Schunter. *Efficient Comparison of Enterprise Privacy Policy.* In Proceedings of the 2004 ACM Symposium on Applied Computing, pages 375-382, 2004.
[ID-WSF] Liberty Alliance Project. *Liberty ID-WSF Web Services Framework Overview. Version 1.1*, 2005. Available on line at: http://www.projectliberty.org/resources/specifications.php
[SAML] OASIS. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005.
Available on line at: http://www.oasis-open.org/apps/org/workgroup/security/
[Seamons 02] K. Seamons, M. Winslett, T. Yu, L. Yu and R. Jarvis. *Protecting Privacy during Online Trust Negotiation.* In 2nd Workshop on Privacy Enhancing Technologies, 2002.
[Winsborough 04] W. Winsborough and N. Li. *Safety in Automated Trust Negotiation.* In 2004 IEEE Symposium on Security and Privacy (S&P '04), page 147-160, 2004