Electronic Preprint for Journal of Information Processing Vol.22 No.3

Regular Paper

Privacy Enhancing Proxies in a Federation: Agent Approach and Cascade Approach

Hiroyuki Sato[1,a)] Yasuo Okabe[2] Takeshi Nishimura[3] Kazutsuna Yamaji[3] Motonori Nakamura[3]

Received: September 13, 2013, Accepted: February 14, 2014

Abstract: In the current network environment, access federations are proving very effective for building trustworthy and efficient service environment. However, operating federations causes some delicate problems regarding trust and security. Among the problems, privacy occupies an essential role in trust building for individual users. Conventionally, privacy aware technologies are concerned with providing anonymity. However, as privacy is understood as the right to control one's own information, and as better services are provided if some privacy information is provided, appropriate hiding and disclosing one's own information is considered more significant. Today, there are considered a wide variety of privacy usages for business, and because such scenarios have their own problems which must be separately solved, uniform "privacy aware technologies" are hard to conceive. They tend to be a collection of ad hoc technologies. In this paper, we consider a scenario of newspaper subscription with student discount. The proof that a subscriber is a student is sent to a newspaper provider from a university identity provider. We consider this scenario in order to extend the menu of services available in a university. Specifically, we explore technologies of proxies that include provision of anonymity and building of trust in a federation. We propose two solutions: SII-like agents, and cascading proxies to envision the privacy protection in this scenario. Their Web profiles are defined and implemented. Moreover, it is proved that both approaches effectively work to protect privacy.

Keywords: privacy, proxy technologies, access federation

1. Introduction

In modern Internet environment, it is being proved that access federations are very effective to build trustworthy and efficient service environment. In particular, they are used to reduce the cost of access, that is, authentication. It is commonly argued that thanks to SSO, the number of passwords that users must remember is drastically reduced. Access federations have been initiated as a small group of service providers with one institutional identity providers, that is, SSO within a given institution. As SSO is widely deployed, the federations have evolved to the form of multiple identity providers with a large number of service providers. Today, in academia, large access federations emerge in Europe, U.S.A. (InCommon), and Japan (GakuNin).

As a federation evolves, it extends the menu of services. Even in academic federations, many commercial service providers can participate in them (e.g., InCommon). Accordingly, there emerge some problems that are essential in order to securely operate a federation. They are essentially related to trust building in federations.

Among the problems, privacy occupies an essential role in trust building for individual users. As we enjoy services essential to our lives, it is natural that the human rights matter there. Particularly because of today's advanced data mining technologies, privacy is easily exposed. Technologies to protect privacy are required in addition to related legislation and regulations.

Conventionally, privacy aware technologies are concerned with providing anonymity. Cryptography is their typical technology component. However, as privacy is understood as the right to control one's own information, and as better services are provided if some privacy information is provided, appropriate hiding and disclosing one's own information is considered more significant. In this way, privacy aware technologies must interact with human intentions of control, and delicate setting is required there.

Actually, a trust framework is considered as a pragmatic solution to the privacy problem. Obeying the privacy policies of a given trust framework, its participants give assurance to endpoint users. However, we need to make additional efforts for this assurance because this solution is not supported by technologies. Assessment and audit are two social solutions serving as these efforts.

Concretely, we have a variety of scenarios for controlling privacy information. In our social life, some providers are trustworthy and some are not in terms of privacy. Releasing privacy information to untrustworthy service providers often causes privacy incidents, which are very hard to recover today. Because such scenarios have their own problems which must be separately solved, uniform "privacy aware technologies" are hard to conceive. They tend to be a collection of ad hoc technologies.

In this paper, we consider a scenario of newspaper subscrip-

[1] The University of Tokyo, Bunkyo, Tokyo 113–8658, Japan
[2] Kyoto University, Kyoto 606–8501, Japan
[3] National Insititute of Informatics, Chiyoda, Tokyo 101–8430, Japan
[a)] schuko@satolab.itc.u-tokyo.ac.jp

tion with student discount. The proof that a subscriber is a student is sent to a newspaper provider from a university identity provider. We consider this scenario in order to extend the menu of services available in a university. In an academic access federation, university identity providers are the authorities in which the attributes released from them are trustworthy. However, extending the service menus causes privacy problems because it easily interacts with students' private lives. Students do not want their university to collect information on their lives irrelevant to academic activities. Through solving these types of problems, we like to enrich the collection of privacy aware technologies that can be applied to a wide range of scenarios.

Specifically, we explore technologies of proxies that include provision of anonymity and building of trust in a federation. It is widely known that proxies are very effective to provide anonymity. We explore proxies to correctly handle privacy related information. They are used to control disclosure and hiding of informations on the side of students.

The rest of this paper is organized: Section 2 shows and analyzes our target scenario. Section 3 proposes our solutions by using proxies. We have two approaches: one is to use SII-like agents, and the other to use cascaded proxies. Section 4 surveys related work. Section 5 concludes this paper.

This work was granted by the Ministry of Internal Affairs and Communications (FY2012 Strategic International Cooperative R&D Promotion Programme: Privacy Enhancement for Open Federated Identity/Access Management Platforms).

2. Motivating Scenarios

As already mentioned, because there are so many scenarios related to privacy, it is reasonable to fix a scenario, to make intensive analysis on it, and to refine the used technologies to a general scenario.

The scenario that we choose in this paper is newspaper subscription with student discount. A newspaper company participates in an academic access federation in which universities provides identities of students. Assume that a student subscribes a newspaper, and that the newspaper company offers student discount, which is a typical scenario in student life. The company wants a strong guarantee that a student actually belongs to a university for fairness and for future promotions. Conventionally, such guarantee is given by a student herself by showing her own student ID card issued by her university. The company checks its validity, and provides a student discount.

This conventional scenario assumes that every transaction is made offline, which is hard to implement on line. However, in terms of privacy, it has an essential advantage. The university does not take care of the private lives of a student. It issues a student ID card, knowing that it may be used in her private life to get some offers including student discount. However, it respects students' private lives, and it does not check the usage. From the view of students, students do not want their private activities to be checked, which are closely related to the freedom of thought.

Therefore, the goal is to implement this "getting student discount by showing her student ID card" model on line in an academic federation. The proof that a subscriber is a student is sent

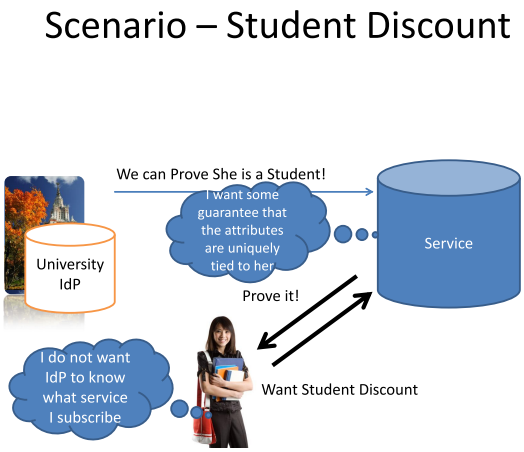


Fig. 1 The Scenario in Privacy Protection.

to a newspaper provider from a university identity provider. However, we need to guarantee that the university identity provider does not/cannot conduct any privacy related misbehavior.

We analyze this scenario as follows:

stakeholders

We have three kinds of stakeholders: university identity providers (IdP), a newspaper company (SP), and a student (user).

privacy requirements

(1) A user does not want IdP to know to which SP she requests a service.

(2) A user does not want any party in the federation to know the value that IdP releases. The only exception must be SP.

security requirements

(1) IdP does not want to release anything to an untrusted SP.

(2) SP wants the sent values to be trusted.

Figure 1 summarizes this scenario.

Note that the similar scenarios appear in other areas. For example, the PCI (Payment Card Industry) standard guarantees that only the amount of the payment matters. The content of customers' purchase must be protected.

Our motivation is to fulfill such requirements with minimal operational effort. Usually, a federation controls IdPs and SPs under some policies on operations. It is a natural idea that the requirements above can be enforced by the operation policies of the federation. However, vulnerabilities appear when too heavy a burden and responsibility are put on human operations. We aim at supporting the operations by developing technologies by which we can control the related operations.

3. Privacy Enhancing Proxies

To fulfill the privacy and security requirements, we place proxies between SP and IdP. Security requirements in operating proxies in an access federation are discussed in trust analysis.

Note that proxy is one of standard technologies to provide anonymities. By combining proxies with cryptography, we show how the requirements are fulfilled.

We have two approaches: one is to use SII-like agents, and the other to use cascaded proxies.

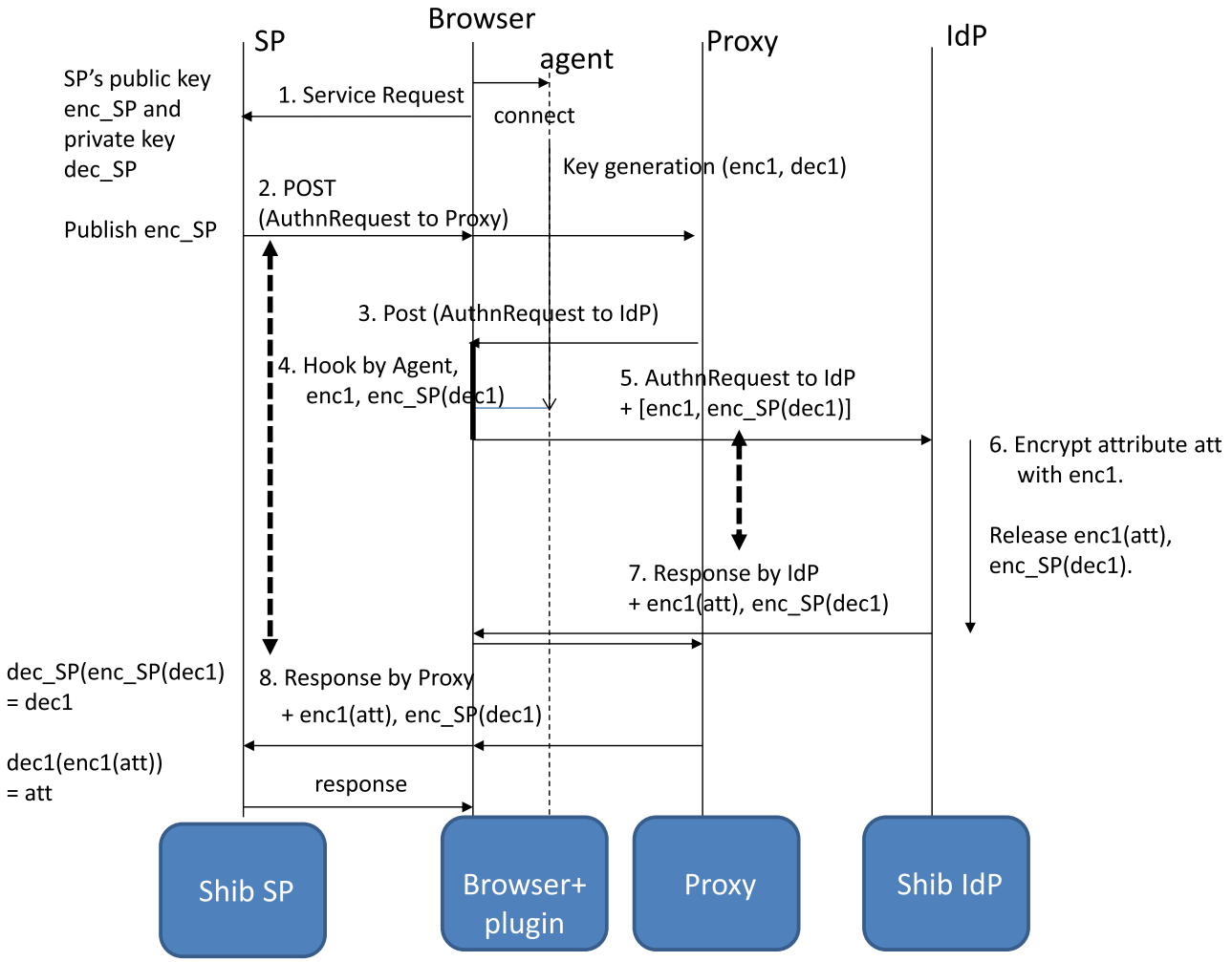


Fig. 2 WebSSO profile of using Ephemeral Key Pairs.

3.1 Problem Specification

Let us represent the privacy requirements in the last section in terms of proxies. We have four stakeholders: IdP, SP, user and proxy(ies).

(1) IdP releases values to a proxy, but does not know the final destination, SP.
(2) The released value is encrypted, and can be decrypted only by SP. In particular, a proxy cannot eavesdrop on the value.

3.2 Solution by an SII-like Agent

Here, we propose a solution based on ephemeral key pairs. The key pairs are generated by a user, and we assume a close tie with the key pairs and the user. Moreover, we consider SAML Web SSO environment, in which we can use browsers in handling SAML. We solve the problem in this environment.

In our solution, an agent is in charge of key generation and delivery. The agent is attached to a browser, and generates an ephemeral key pairs per transaction. Then it delivers the pair to IdP, and finally to SP.

In OpenID Connect [14], in addition to ordinary RP (Relying Party)s and OP (Openid Provider)s, SII (Self-Issued IdP)s are defined. An SII generates ephemeral keys that are used in communications between RP and a user agent. In other words, an SII plays an IdP for the sake of its owner user. To effectively operate SIIs, An SII is involved in the trust where RPs and OPs participate in. It is assumed that the owner user of SII is responsibile for its trust.

In SAML, we introduce an SII-like agent as follows.

3.2.1 Profile

First, we show the WebSSO profile in **Fig. 2**. Note that we assume an appropriate PKI for the trust of a certificate issued to a user.

Here, we show how the requirements are fulfilled. First we assume that SP has its own key pairs, and publishes its public key. Appropriate PKI must also be assumed.

In this solution, a user has her agent as a plugin of her Web browser. We assume a strong tie of the user and the agent.

The transaction is processed as follows (the numbers in the figure correspond to the numbers below):

(1) First, a user issues a request to SP as usual.
(2) SP posts AuthnRequest to Proxy.
(3) Proxy then builds an AuthnRequest to IdP, and posts it as a part of HTTP redirect.
(4) The agent (a plugin of browser) detects the AuthnRequest to IdP in the redirect. It generates an ephemeral key pair, encrypts its private key by SP's public key, and inserts the encrypted private key and the public key into the original request.

(5) IdP receives the AuthnRequest, and authenticates a user.
(6) IdP is configured so that the key pair and the requested attributes are released. The attributes are encrypted by the public key sent from the agent.
(7) Proxy receives the Response from IdP. Then Proxy builds the Response to SP based on the received Response. This Response is sent to SP.
(8) Then, SP receives via Proxy the encrypted attributes with the key pair. SP now decrypts the attributes with its private key.
(9) SP provides services depending on the obtained attribute values.

In this profile, there are two pairs of AuthnRequest/Reponse's. In the figure, we draw dotted lines to clarify the correspondences.

As for privacy requirements,

(1) IdP releases its values to Proxy. It does not know SP, the entity to which the value is sent.
(2) The attribute values released by IdP are encrypted by SP's public key. Therefore, only SP can decrypt the values. Proxy cannot decrypt the values.

As for security, we assume channel bindings of Agent and IdP. Agent directly delivers its key pair to IdP, the sent key pair cannot be compromised by MITM attack if we have safe channel binding.

Furthermore, because the delivered key (for decryption) is signed by the agent, SP can validate that the key is not compromised by the proxy.

In OpenID Connect, there is defined an SII (Self Issued IdP). It plays the role of an agent of a user. In this meaning, the first solution is very similar to SIIs.

Originally, SII is designed so that it works as an IdP personal to a user. Personal devices such as mobile phones are supposed to work as SIIs. Therefore, its functions are the same as ordinary IdPs except that it is out of scope of discovery services, and that a user needs to trust its key pairs.

In our approach, we limit the functions of agent to the management of key pairs. Agent can be light weighted. In our implementation, we assume that the agent is implemented as a plugin of a browser. User interaction looks the same as the one without agents.

3.2.2 Implementation

We assume Shibboleth IdPs and SPs as our platform. As for proxies, any technologies can be adopted. The minimal requirements in implementation are that for SP and IdP, the modification must be done by modules of Shibboleth.

The IdP system is configured in `attribute-resolver.xml` so that the encrypted key and the public key are given predefined attribute names, and IdP releases them to a collection of proxies that are designated in advance. Proxy and plugin can be coded from scratch. We have implemented Proxy in SimpleSAML.php. The agent system is implemented as a plugin to Firefox and Chrome.

The profile shown in Section 3.2.1 is implemented as follows. We use the same number as the profile as for the corresponding description.

- First, two new attribute definitions are given: `ag_pub_key` and `ag_priv_key`. They are used in the encryption and decryption, respectively. `ag_priv_key` must be defined in `attribute-resolver` of the IdP shown below:

```
<resolver:DataConnector id="agentParams"
          xsi:type="extra-dc:AgentParams" />
 <resolver:AttributeDefinition xsi:type="ad:Simple"
           id="agentPrivKeyDef"
           sourceAttributeID="ag_priv_key">
        <resolver:Dependency ref="agentParams" />
        <resolver:AttributeEncoder
           xsi:type="enc:SAML2String"
           name="urn:mokha:attribute:ag-priv-key"
           friendlyName="ag-priv-key" />
  </resolver:AttributeDefinition>
```

- In order to enable encryption of designated attributes, they must be defined as an `SAML2SecureString` in `AttributeDefinition` as defined in this implementation.

```
<resolver:AttributeDefinition
        xsi:type="gakunin-ad:SecureSimple"
        id="title" sourceAttributeID="title">
  <resolver:Dependency ref="myLDAP" />
  <resolver:Dependency ref="agentParams" />
  <resolver:AttributeEncoder
        xsi:type="enc:SAML1String"
        name="urn:mace:dir:attribute-def:title" />
  <resolver:AttributeEncoder
        xsi:type="gakunin-enc:SAML2SecureString"
        name="urn:oid:2.5.4.12"
        friendlyName="title" />
</resolver:AttributeDefinition>
```

2,3 `AuthnRequest` is sent from SP to IdP in the original form of SAML. It is redirected to Proxy.
4 The agent hooks the `AuthnRequest`, and inserts the key pairs generated by the agent itself. It is expressed in `<samlp:Extensions>` as below:

```
<samlp:AuthnRequest
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
      ...
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
 <saml:Issuer>
   https://px.example.com/simplesamlphp/sp
 </saml:Issuer>
 <samlp:NameIDPolicy
   AllowCreate="true"
   Format=
 "urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
 />
 <samlp:Extensions>
  <protocol_type
   xmlns="http://idprivacy.mokha.co.jp/proxy_extension">
        agent
  </protocol_type>
  <ag_pub_key
   xmlns="http://idprivacy.mokha.co.jp/proxy_extension">
       (base64-pub)
  </ag_pub_key>
  <ag_priv_key
   xmlns="http://idprivacy.mokha.co.jp/proxy_extension">
        (base64-priv)
  </ag_priv_key>
 </samlp:Extensions>
</samlp:AuthnRequest>
```

Protocol type (agent), ag_pub_key, and ag_priv_key are inserted.

6 IdP builds its response. It contains assertion statements together with encrypted attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2:Assertion
  xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="_80c3d8170f6b6a51d032a52821b425b2"
  IssueInstant="2013-02-05T06:45:54.015Z"
  Version="2.0"
...

  <saml2:AttributeStatement>
    <saml2:Attribute
       FriendlyName="eduPersonAffiliation"
       Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
       NameFormat=
       "urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
      <saml2:AttributeValue
        xmlns:xsi=
        "http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">
       (Encrypted Value)
      </saml2:AttributeValue>
    <saml2:Attribute FriendlyName="ag-priv-key"
       Name="urn:mokha:attribute:ag-priv-key"
       NameFormat=
       "urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
      <saml2:AttributeValue
        xmlns:xsi=
        "http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">
       (Encrypted Private Key)
      </saml2:AttributeValue>
    </saml2:Attribute>
  </saml2:AttributeStatement>
</saml2:Assertion>
```

Here, `eduPersonAffiliation` is encrypted by the `ag_priv_key` which is further encrypted by the SP's public key.

7 SP sees if the sent attribute `ag_priv_key` is given. When it is given, SP decrypts the sent encrypted attributes by using the key.

3.3 Solution by Cascading Proxies

Next, we consider a solution by cascading proxies.

In the first solution, we consider a single proxy. This is effective when a user directly subscribes a service of an SP. She discloses her identity in the subscription. The SP identifies the IdP that can release her attribute.

In a more delicate situation, a user does not want any connection between SPs and IdPs. If the level of assurance on attribute values is given by their participating federation, their sources do not matter.

Cascading proxies can solve this problem with the same requirements: the released value is encrypted, and can be decrypted only by SP. In particular, proxies cannot obtain the value. Software architectures that adopt proxy cascades are common in enhancing proxies. Tor is its typical example. However, in Tor, the data that the last proxy receives is decrypted by itself. Because this flaw is essential in onion protocol, we need another solution.

Instead, we assume that we have proxies that represent IdPs and those that represent SPs. A proxy representing IdPs is in charge of protecting information issued by an IdP, and transferring the information to an SP designated by the IdP. In particular, it hides the name of issuing IdP. The same applies to a proxy representing SPs. Connecting the two proxies, we have the property that the names of IdPs and SPs are anonymized. Moreover, the information issued by an IdP are protected against being eavesdropped by other parties, particularly by an proxy representing an SP. In the same way, the information issued by an SP cannot be compromised by other parties other than the SP proxy. Combining the two, we obtain the property that the information is protected against fraudulent proxies.

In order to envision these properties, we use commutative encryption scheme. Commutative encryption scheme uses two keys $K1$ and $K2$ such that encryption enc_{Ki} $(i = 1, 2)$ satisfies $enc_{K1} \circ enc_{K2} = enc_{K2} \circ enc_{K1}$. Accordingly, to the data $enc_{K2}(enc_{K1}(d))$, a party having $K1$ can apply $dec_{K1}(enc_{K2}(enc_{K1}(d))) = dec_{K1}(enc_{K1}(enc_{K2}(d)) = enc_{K2}(d)$. This scheme is used to secret sharing among multi-parties. Here, we apply it to cascading encryption and decryption. Let PxI be a proxy representing IdPs, and PxS a proxy representing SPs. Specifically, we assume that in a federation, we have a connection below:

$$\text{IdP} \longrightarrow \text{PxI} \longrightarrow \text{PxS} \longrightarrow \text{SP}$$

SP and the proxy representing IdPs share the key $K1$, and IdP and the proxy representing SPs share the key $K2$. We assume that the keys are sessionwise generated, and that the parties do not have to share keys beforehand.

Data A is encrypted as $enc_{K2}(A)$ by IdP and sent to the neighbor. PxI further encrypts the data by $K1$ as $enc_{K1}(enc_{K2}(A))$ and sends it to the neighbor. PxS decrypts the data by its key $K2$ as $dec_{K2}(enc_{K1}(enc_{K2}(A))) = dec_{K2}(enc_{K2}(enc_{K1}(A))) = end_{K1}(A)$ and sends it to SP. Finally, SP decrypts the data by its key $K1$. It is clear that the proxies residing in the middle of the communication cannot obtain A.

3.3.1 Profile

We show the profile for cascading proxies in **Fig. 3**.

Here, $\oplus$ represents encryption/decryption, and we assume that $K1$ and $K2$ are commutative symmetric keys.

The transaction is processed as follows. A major problem here is key generations and distributions of $K1$ and $K2$. Here, we explain a sessionwise key generation and distribution.

(1) First, SP generates a symmetric key $K1$ that will be shared by SP and PxI in the session.

(2) Next, SP sends PxS a request of attribute A, with $K1$ encrypted by the public key of PxI (K1) and the public key of SP.

(3) PxS receives the data. Then PxS generates a symmetric key $K2$ that will be shared by PxS and IdP in the session.
Then, PxS sends PxI the received request together with $K2$ encrypted by the public key of IdP and the public key of PxS.

(4) PxI receives the data, and transfers it to IdP.

(5) IdP decrypts $K2$. IdP can make decryption because $K2$ is encrypted by the public key of IdP. At this point, $K2$ is shared by IdP and PxS. Then, it sends the encrypted attribute ($A \oplus K2$ in Fig. 3) with the encrypted $K2$ to PxI. Furthermore, it sends all of the encrypted keys.

(6) PxI receives the data, and decrypts $K1$. At this point, $K1$ is

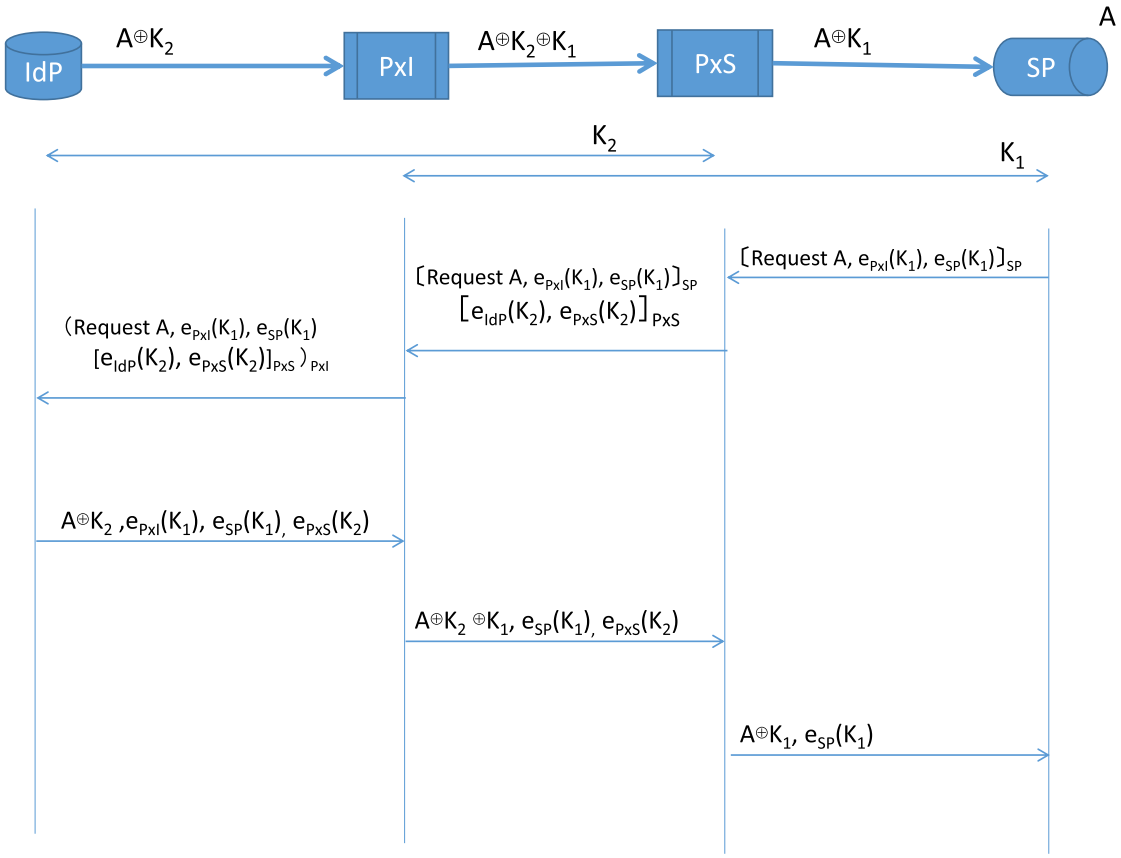


Fig. 3 Profile of Cascading Proxies.

shared by PxI and SP. It is possible because the sent data contains $K1$ encrypted by the public key of PxI. PxI further encrypts the attribute with $K1$ ($A \oplus K2 \oplus K1$ in Fig. 3). Then it sends the data with still encrypted keys to PxS.

(7) PxS receives the data, and decrypts $K2$. It is possible because the data contains the encrypted $K2$ with the public key of PxS. PxS applies $(A \oplus K2 \oplus K1) \oplus K2 = (A \oplus K1 \oplus K2) \oplus K2 = A \oplus K1$. Then it sends $A \oplus K1$ and $K1$ encrypted with the public key of SP to SP.

(8) SP receives the data, and decrypts the data.

In this profile, we see that each entity communicates only with its neighbors, and that the attribute A cannot be decrypted by proxies PxI and PxS, which fulfills the privacy and security requirements.

We have implemented the proxies by using simpleSAML.php together with IdPs and SPs by adding related modules of Shibboleth.

3.4 Security/Privacy/Trust Analysis

In order to effectively operate proxies, trust analysis of proxies is indispensable. In this section, we make a security analysis of our two approaches, and reveal trust requirements in order to securely operate our methods.

The security and privacy analysis on the side of a user includes those on the threats below:

(1) The attribute values may be eavesdropped by an entity other than IdP.

(2) SP may be identified by IdP (tracking by IdP).

(3) Proxies may track the communications (tracking by Proxies).

(4) Proxies may perform MITM attack.

(5) Agent may behave against the user (Approach 1 only).

All the threats listed above are tamed in our two approaches as stated in the previous sections. However, we must note that this security analysis is based on a certain trust assumptions of a federation.

First, in the operation, we assume that all entities work within a fixed federation. A federation provides trust to each participating entity.

In particular, we can assume that their public keys are available in the metadata of a federation. Second, we assume PKI on key pairs of IdP and SP. Because of these assumptions, we have, as for (1) and (2), eavesdropping is impossible thanks to PKI assumption.

In addition to the trust above, we need extra trust on proxies as for threats (3) and (4). As a major requirement, a proxy must not collude with IdPs and SPs. Because a user's privacy is protected against malicious behavior of IdPs and SPs, a certain trusted party must operate proxies. This must be assured by the trust of the operating federation. We can assume that a trust framework provider (operation manager of federation, or federation itself) is a trusted party to a user. This means an extra burden on a federation. The content that goes through proxies is appropriately encrypted, and in this sense, we do not need further security on proxies.

As for the threat 5 of the first approach, we need the assump-

tion of close tie of a user's browser and the agent. This must be assured externally to the trust.

The second problem is the assurance level of attribute values. SP requires some assurance level of the released attribute values. LoAA (Level of Assurance on Attributes) is an essential problem in utilizing attribute values.

3.5 Problems in Deployment

Besides the tecnical analysis on security and trust, we also have problems in deployment of this solution in a federation.

First, the target federation must be securely operated minimally to ensure secure data communications. Metadata management, policies in releasing specific attributes, and control of IdPs and SPs are indispensable for secure operations. Furthermore, the federation is responsible for operating proxies. The trust of proxies is the key to these technologies. In this meaning, building a certain trust framework is required.

Second, the federation is responsible for indirect attacks against privacy. In particular, As for the identification of SP, if the number of SPs is very small, IdPs can easily identify an SP with high probability. This problem can only be solved by hiring enough SPs to lower the probability. We notice this problem as one for an operating federation.

As the third problem, the federation is responsible for incident responses. As for some specific possible incidents such as duplicate subscriptions, there are proposed an ad-hoc solution such as Ref. [9]. However, generally, in a federation, because there are multiple stakeholders such as IdP, SP and proxies, incident analysis itself is not an easy task. Proxies are responsible for connecting an IdP and an SP whose logs can be used in the analysis. In this sense, proxies must be operated by the federation itself as the anchor of the analysis.

Finally, we discuss the interoperability of privacy related technologies. Today, in a SAML federation, uApprove is provided for an opt-in tool for releasing a user's attribute to an SP. Using it as a plug-in to Shibboleth, it displays pairs of attribute names and values computed by instructions in `DataConnector`. In our implementation, because the values are encrypted by using the instructions, encrypted values are displayed to a user. A user must judge release of attributes by seeing only attribute names. This problem is essential in the architecture of Shibboleth and uApprove. We must admit that what we can provide so far is an "out-of-band" solution such as offline approval.

4. Related Work

Shibboleth [15] began as one of Internet2 middleware activities, and is now one of major SAML processors. uApprove [16] began as a project of SWITCH, and is now widely adopted in Japan and Europe.

As federations extend their coverage to social identities and activities [6], [13], privacy has become one of central issues in their operations. Privacy protection has been discussed in terms of anonymities and encryption. Reference [10] discusses ID-based encryption in privacy protection, and Ref. [11] discusses authentication in terms of privacy.

It is often a case that a trust framework declares its own privacy policies. Actually, Ref. [7] defines the criteria of the privacy policies, which reflects the requirements of US ICAM.

Technologies of proxies are now commonly used to provide anonymity. Tor and onion routing [2], [3], [17], a sophisticated proxy architecture, are widely used for providing anonymity. References [5], [8] also discuss proxies for privacy protection.

In OpenID communities, OpenID Connect [14] is defined, in which an SII is introduced. Reference [12] also discusses external devices to strengthen the level of authentication.

The commutative encryption is introduced in Ref. [1], and mainly used in secret sharing [4].

5. Conclusion

In this paper, we have discussed proxy technologies in terms of privacy protection. Two approaches, an SII-like agent and cascading, have been designed and implemented, and proved to be effective. Furthermore, the trust of the proxy systems has been analyzed in order for the privacy enhancing proxies to work effectively. The operations of a federation including LoAA has also been discussed.

References

[1] Agrawal, R., Evfimievski, A. and Srikant, R.: Information sharing across private databases, *International Conference on Management of Data* (*ACM SIGMOD*), pp.86–97 (2003).

[2] Dingledine, R., Mathewson, N. and Syverson, P.: Tor: The Second-Generation Onion Router, Naval Research Lab (2004), available from ⟨https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf⟩.

[3] Feigenbaum, J., Johnson, A. and Syverson, P.: A model of onion routing with provable anonymity. *Proc. 11th Financial Cryptography and Data Security*, pp.57–71 (2007).

[4] Jun, H., Ximei, L., Lijuan, L. and Chunming, T.: A New Secret Sharing Scheme Based on Commutative Encrypted Function, *Information Technology and Applications* (*IFITA*), Vol.2, pp.27–29 (2010).

[5] Ma, Z., Manglery, J., Wagner, C. and Bleier, T.: Enhance Data Privacy in Service Compositions through a Privacy Proxy, *Proc. 2011 6th International Conference on Availability, Reliability and Security*, pp.615–620 (2011).

[6] Nakamura, M., Nishimura, T., Yamaji, K., Sato, H. and Okabe, Y.: Privacy Preserved Attribute Aggregation to Avoid Correlation of User Activities Across Shibboleth SPs, *Proc. 36th Computer Software and Applications W* (*MidArch*), pp.367–372 (2013).

[7] Open Identity Exchange: Trust Framework Assessment Package (2010).

[8] Pertlic, R.: Proxy re-encryption in a privacy-preserving cloud computing DRM scheme, *Proc. 4th International Conference on Cyberspace Safety and Security 2012*, pp.194–211 (2012).

[9] Okabe, Y., Sato, H., Nishimura, K., Yamaji, K. and Nakamura, M.: An Anonymizing Proxy that ides a Service Provider against an Attribute Provider, *Multimedia, Distributed, Cooperative, and Mobile Symposium* (*DICOMO*) 2013, 8F-2 (2013). (In Japanese)

[10] Shao, J.: Anonymous ID-based proxy re-encryption, *Proc. 17th Australasian Conference on Information Security and Privacy*, pp.364–375 (2012).

[11] Tan, Z.: A lightweight conditional privacy-preserving authentication and access control scheme for pervasive computing environments, *Journal of Network and Computer Applications*, Vol.35, No.6, pp.1839–1846 (2011).

[12] Trammel, J., Yalcinalp, U., Kalfas, A., Boag, J. and Brotsky, D.: Device token protocol for persistent authentication shared across applications, *Proc. 1st European Conference on Service-Oriented and Cloud Computing*, pp.230–243 (2012).

[13] Zabrisky, C., Getts, M., Basney, J., Caskey, P. and Carmody, S.: Social2SAML: Accepting Social identities for InCommon Federated Services (online), available from ⟨http://www.incommon.org/docs/iamonline/20121212_IAM_Online.pdf⟩ (2012).

[14] OpenID Foundation, available from ⟨http://openid.net/connect/⟩.

[15] Shibboleth.net, available from ⟨http://shibboleth.net/⟩.

[16] SWITCH, available from ⟨http://www.switch.ch/aai/support/tools/uApprove.html⟩.

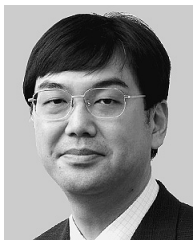
[17] Tor Project, available from ⟨http://www.torproject.org/⟩.

Hiroyuki Sato is an Associate Professor in the University of Tokyo. Received his B.Sc., M.Sc. and Ph.D. from the University of Tokyo in 1985, 1987, 1990, respectively. Majoring: computer science and information security. He is a member of ACM, IEEE, IPSJ and JSSST.

Yasuo Okabe received his M.E. from Department of Information Science, Kyoto University in 1988. From 1988 he was an Instructor of Faculty of Engineering, from 1994 he was an Associate Professor of Data Processing Center, and from 1998 he was an Associate Professor of Graduate School of Informatics, Kyoto University. He is now a Professor of Academic Center for Computing and Media Studies, Kyoto University. Ph.D. in engineering. His current research interest includes Internet architecture, network security and distributed algorithms. He is a fellow of IEICE and a member of IPSJ, ISCIE, JSSST, IEEE, and ACM.

Takeshi Nishimura graduated from the University of Tokyo, Japan, where he received his B.Sc. and M.Sc. degrees in information science in 1996 and 1998, respectively. From 2001, he was a Research Associate at the University of Tokyo. Since 2009, he is a Project Researcher at National Institute of Informatics, Japan. His research interests are authentication and authorization for federated identity, identity federation management and public key infrastructure (PKI).

Kazutsuna Yamaji received a Ph.D. in Systems and Information Engineering from Toyohashi University of Technology in 2000. Currently he is an Associate Professor at the National Institute of Informatics, Japan. His research interests include open science, data shari ng and identity federation. He is a member of IPSJ, IEICE, and the Japan Society of Information and Knowledge (JSIK).



Motonori Nakamura graduated from Kyoto University, Japan, where he received his B.E., M.E. and Ph.D.—degrees in engineering in 1989, 1991 and 1996, respectively. From 1994, he was an Assistant Professor at Ritsumeikan University. From 1995, he was an Associate Professor at Kyoto University. Currently he is a professor at National Institute of Informatics, Japan (NII). His research interests are message transport network systems, network communications, next generation internet and Identity & Access Management. He is a member of IEEE, ISOC, IEICE, IPSJ and JSSST.