A Client-side CardSpace-Liberty Integration Architecture

Waleed A. Alrodhan
Royal Holloway, University of London
Egham, Surrey, TW20 0EX
United Kingdom
W.A.Alrodhan@rhul.ac.uk

Chris J. Mitchell
Royal Holloway, University of London
Egham, Surrey, TW20 0EX
United Kingdom
C.Mitchell@rhul.ac.uk

ABSTRACT

Over the last few years, many identity management schemes, frameworks and system specifications have been proposed; however these various schemes and frameworks are typically not interoperable. In this paper we propose an approach to enable interoperation between two of the most prominent identity management schemes, namely the Liberty Alliance Project scheme (specifically the ID-FF LEC Profile) and the Microsoft CardSpace (formerly known as InfoCard) scheme. This integration should enhance interoperability by enabling users to make use of identity management systems even if the system participants are using different schemes. The main advantages and disadvantages of the proposed integration model are also investigated.

1. INTRODUCTION

It has become common for Internet users to access multiple independent systems in a single working session, and hence users have a need for multiple digital identities. However, managing these digital identities can be a complex and difficult job, and to solve this dilemma a number of Identity Federation systems have been proposed and deployed. These systems are typically not interoperable, which makes it difficult to use them in open environments such as the Internet.

This paper proposes an approach to address this problem. Specifically, it proposes a method to enable interoperation between the Liberty Alliance Project scheme and the Microsoft CardSpace scheme.

The remainder of this paper is organised as follows. Section 2 provides an overview of the Liberty Alliance Project and the Microsoft CardSpace identity management system. Section 3 presents the proposed integration model. In section 4 we provide an operational analysis of the proposed integration model, section 5 contains a brief review of related work, and section 6 concludes the paper.

2. THE LIBERTY ALLIANCE PROJECT AND MICROSOFT CARDSPACE

This section provides a brief introduction to the two identity management architectures for which interoperation is enabled, namely the Liberty Alliance Project scheme and Microsoft CardSpace.

2.1 The Liberty Alliance Project and the ID-FF Single Sign-On Profiles

The Liberty Alliance Project (www.projectliberty.org) is an industry collaboration that was started in December 2001 by 16 major companies, including Sun, GM, United Airlines, and France Télécom. This collaboration now involves more than 150 members, including government agencies, companies, banks and universities. According to the project website, there are more than 400 million Liberty-enabled identities and clients across the world.

The Liberty Alliance Project (henceforth abbreviated to Liberty) aims to build open standard-based specifications for federated identity, provide interoperability testing, and to help provide solutions to identity theft. Liberty also aims to establish best practices and business guidelines for identity federation.

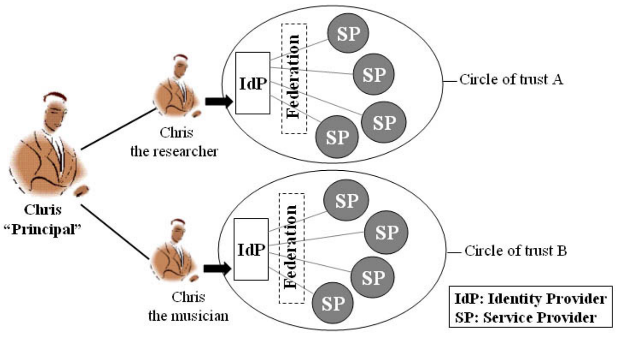

Figure 1: The Liberty model.

Figure 1 shows the general Liberty model, which is essentially a single sign-on model [2]. In this model, a principal (or user) can federate its various identities to a single identity issued by an identity provider, so that the user can access services provided by service providers belonging to the same circle of trust by authenticating just once to the identity provider. This, of course, relies on a pre-established trust relationship between the identity provider and every service provider in the circle of trust. The model provides a level of pseudonymity and unlinkability via the use of pseudonyms instead of real identifiers in the communications between the identity provider and the service providers, and this enhances user privacy. In the example shown in figure 1, the principal has federated its identities within two distinct circles of trust, which results in the user having two identities, one for circle of trust A and the other for circle of trust B. It merits mentioning that these two identities could also be federated, but this would require a pre-established trust relationship between the identity providers.

As shown in figure 2, the Liberty implementation specifications are divided into three frameworks: the Identity Federation Framework (ID-FF) [12], the Identity Web Services Framework (ID-WSF) [11] and the Service Interface Specifications (ID-SIS) [5] and [6]. In this paper, we focus on the ID-FF.

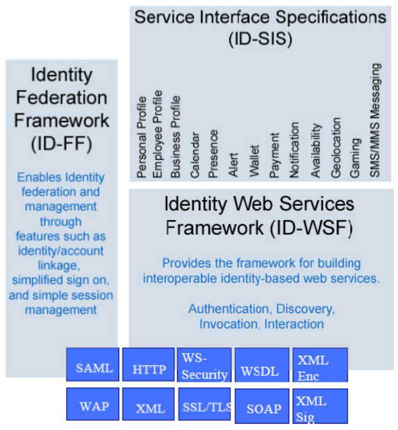
Service Interface Specifications (ID-SIS)
Identity Federation Framework (ID-FF)
Enables Identity federation and management through features such as identity/account linkage, simplified sign on, and simple session management
Personal Profile
Employee Profile
Business Profile
Calendar
Presence
Alert
Wallet
Payment
Notification
Availability
Geolocation
Gaming
SMS/MMS Messaging
Identity Web Services Framework (ID-WSF)
Provides the framework for building interoperable identity-based web services.
Authentication, Discovery, Invocation, Interaction
SAML
HTTP
WS-Security
WSDL
XML Enc
WAP
XML
SSL/TLS
SOAP
XML Sig

Figure 2: The Liberty Frameworks.

The ID-FF provides approaches for implementing the federation and single sign-on model, and the techniques needed for that model, including session management and identity/account linkage. In order to realise this federation model, a set of profiles is required (so called ID-FF Liberty profiles). An ID-FF Liberty profile may best be defined as the combination of message content specifications and message transport mechanisms for a single client type (that is, user agent) [3].

There are many types of ID-FF Liberty profile, including Single Sign-On and Federation Profiles, Register Name Identifier Profiles, Identity Federation Termination Notification Profiles, Single Logout Profiles, Identity Provider Introduction, NameIdentifier Mapping Profile and NameIdentifier Encryption Profile. In this paper we are primarily concerned with the Single Sign-On and Federation Profiles. The currently defined Single Sign-On profiles are the Artifact profile, the Browser POST profile and the Liberty-enabled client and proxy (LEC) profile [3].

2.2 Microsoft CardSpace

CardSpace is the name for a Microsoft WinFX software component that is built on the concept of the “identity metasystem”. This identity metasystem is designed to comply with the Laws of Identity, as promulgated by Microsoft [1]. The metasystem provides a way to represent identities using claims, and a means to bridge technology and organisational boundaries using claims transformations [7]. The CardSpace identity management architecture is designed to provide the user with control over his digital identities in a user friendly manner, and to tackle identity management security problems such as breaches of privacy and identity theft, with no single identity authority control. CardSpace works with Internet Explorer browsers (CardSpace plug-ins for browsers other than Microsoft Internet Explorer can also be developed, such as the Firefox Plug-in[1]).

The concept behind CardSpace is relatively simple; it is based on the identification process we experience in the real world when using physical identification cards. In the CardSpace system, an identity provider issues virtual identification cards (named InfoCards) to users, who can later use them to identify themselves to any service provider who trusts this identity provider. It merits mentioning here that the InfoCard is stored in the user’s machine and does not contain any security-sensitive information.

CardSpace is an identity federation model that essentially facilitates a single sign-on service, although it is not clear from the published Microsoft papers and documents how single sign-on sessions are handled within the metasystem. The CardSpace metasystem makes use of Web Services (WS-*) protocols to achieve its objectives. Note that most of these protocols make use of a *Security Token Service* [4].

2.3 Message Flows within the Liberty ID-FF LEC Profile and the CardSpace Framework

Before describing the proposed integration model, we briefly describe the message flows within both the Liberty ID-FF LEC profile and the CardSpace framework. There are three main roles within both identity management architectures:

1. The Identity Provider or Identity Issuer (IdP), which issues the identity to the user;
2. The Service Provider (SP), as referred to in the Liberty specifications, or the Relying Party (RP), in Microsoft terminology; the SP or RP needs to identify the user before providing services to him/her;
3. The Principal or User.

It is important to mention that, in order for a principal to employ Liberty federation using the LEC profile, the principal must possess a Liberty-Enabled browser in order to handle and understand the messages sent and received. To make the browser Liberty-Enabled, the principal needs to install certain java components on the machine; such components can be downloaded freely from the Internet (e.g. the SecureID ID-FF 1.1 and ID-FF 1.2 Java Toolkits[2], and the Sun FederationSPAdapter[3]).

Figure 3 shows the message flow within the ID-FF LEC profile, if we assume that the client has already been authenticated by the IdP. Note that the Liberty-Enabling component must be installed on the Principal’s PC prior to performing the protocol. The main steps in the protocol are as follows:

1. **User Agent → SP** : Service Request (HTTP Request with Liberty Enabled Header)

[1]http://xmldap.blogspot.com/2006/05/firefox-identity-selector.html
[2]http://www.sourceid.org
[3]http://docs.sun.com/source/819-4682

2. **SP → User Agent** : Authentication Request + "optionally" an IdP List
3. **User Agent or User** : Selects the IdP to be used
4. **User Agent → IdP** : SAML-Assertion Request
5. **IdP → User Agent** : SAML-Assertion Response
6. **User Agent → SP** : Authentication Response + SAML-Assertion (within an HTML Form)
7. **SP → User Agent** : Service Granted!

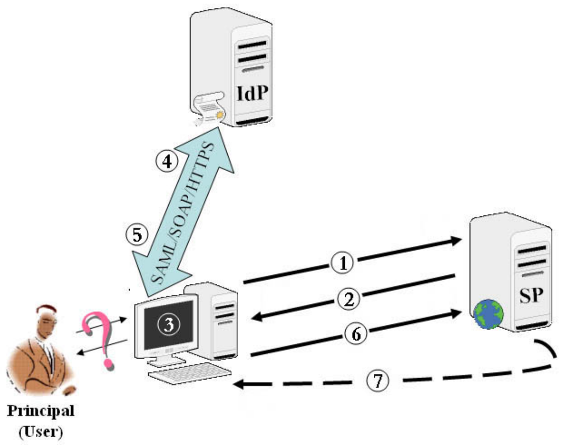

Figure 3: The ID-FF LEC profile message flow.

The SAML messages in steps 4 and 5 are bound to SOAP, which is carried over an SSL connection to provide confidentiality (integrity is preserved using an XML-Signature). As shown in the figure, how the IdP is selected in step 3 is not specified in the Liberty specifications.

Figure 4 shows the message flow within the CardSpace framework. As shown in the figure, there is an additional component here, namely the *Security Token Service* (henceforth abbreviated to STS), which is responsible for the security policy and token management within the IdP (and optionally within the RP). The User Agent here is essentially an CardSpace enabled browser (also called the *Service Requestor*). Note that the CardSpace component must be installed on the Principal's PC prior to step 3 of the protocol. The main steps in the protocol are as follows:

1. **User Agent → RP** : HTTP GET Login HTML Page Request
2. **RP → User Agent** : HTML Login Page + CardSpace Tags (XHTML or HTML object tags)
3. **User Agent ↔ RP-STS** : User Agent retrieves Policy via *WS-SecurityPolicy*
4. **User Agent ↔ User** : User picks an InfoCard
5. **User Agent ↔ IdP** : User Authentication
6. **User Agent ↔ IdP-STS** : User Agent retrieves security token via *WS-MetadataExchange* and *WS-Trust*
7. **User Agent → RP-STS** : User Agent presents the security token via *WS-Trust*
8. **RP → User Agent** : Welcome, you are now logged in!

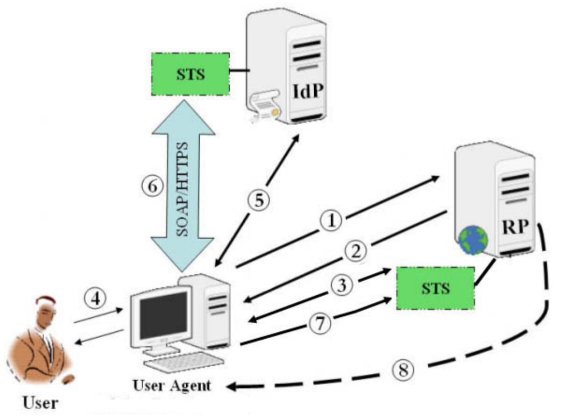

Figure 4: The MS-CardSpace profile message flow.

Unlike the ID-FF LEC profile, in Step 4 of the CardSpace message flow the user is presented with the InfoCards that can be used to identify himself to that particular RP, and picks one of them. The WS-MetadataExchange and WS-Trust messages in step 6 are transported using SOAP, which is carried via an SSL connection to provide confidentiality (integrity is preserved using an XML signature). If the RP does not have an STS server, the messages in steps 3 and 7 will be carried using HTTP over an SSL/TLS channel.

In the remainder of this paper we assume that all SPs and IdPs support at least one of the two identity architectures (either Liberty or CardSpace, or both). In such a situation, Windows users will face one of the following two scenarios:

- *Scenario A*: The Identity Provider/Identity Issuer supports both identity management architectures (Liberty and CardSpace). In this case the IdP perform the difficult task of maintaining two different identity architectures, thereby providing flexibility to users when they federate their identities with the SPs or RPs. Users are able to access services provided by Service Providers regardless of which identity management architecture they adopt.
- *Scenario B*: The Identity Provider/Identity Issuer supports only one identity management architecture (either Liberty or CardSpace). In this case, users are only able to access Service Providers using the identity management architecture supported by their Identity Issuer or IdP.

In Scenario B, Windows users have a compatibility problem if the Relying Party is CardSpace-Enabled while their IdP is Liberty-Enabled, or vice versa. Table 1 shows the applicability of the identity management systems in all of the possible scenarios that might occur; L.E. stands for Liberty Enabled, and CS.E. stands for CardSpace Enabled. The (✓) sign indicates that there is no compatibility problem, whereas the (×) sign indicates the opposite.

3. THE INTEGRATION MODEL

There is a noticeable similarity between the message flow within the ID-FF LEC profile and the message flow within the CardSpace framework. This similarity can be exploited in order to integrate the two identity management archi-

Table 1: The applicability of the identity management architectures.

User Agent	L.E. IdP L.E. SP	CS.E. IdP CS.E. RP	L.E. IdP CS.E. RP	CS.E. IdP L.E. SP
Unenhanced	×	×	×	×
L.E.	✓	×	×	×
CS.E.	×	✓	×	×
L.E. and CS.E.	✓	✓	×	×

tectures. This section presents the motivation for this integration proposal, and describes the proposed integration model.

3.1 Motivation

Liberty is currently the leading identity management architecture for identity federation, and it has gained the acceptance of a number of technology-leading companies and organisations. By contrast, the CardSpace metasystem currently only works on the Windows operating system (this seems likely to continue, at least in the near future), and is deployed freely with Windows Vista, with backwards compatibility with Windows XP. Given the wide use of Windows, CardSpace is likely to have a significant impact despite this restriction. Thus, enabling integration between these two systems is likely to be of significant benefit to a wide range of service providers and users. It seems that interoperation between Liberty and CardSpace can be achieved by integrating the frameworks.

The CardSpace identity management architecture consists of two parts:

1. *The user agent supporting components*: i.e. the service requestor and the identity selector. These components are responsible for managing the cards and communicating with other parties in the model. It appears that these components could be integrated with the Liberty ID-WSF services; however, how this might be achieved is beyond the scope of this paper.
2. *The Identity Framework*: i.e. the message flow and the rules for communication between the parties. This framework is similar to the Liberty ID-FF LEC profile, in which the user agent is "Liberty-enabled" in the same way as the user agent is "CardSpace-enabled" in the CardSpace framework.

3.2 Integrating the two schemes

In the integration model we propose, we introduce an identity management architecture adaptor that is both Liberty-enabled and CardSpace-enabled. We propose placing this adaptor in the user's machine. This gives the user the ability to use any IdP, and access a service provided by any RP or SP. This means that, in the presence of such an adaptor, we can change the last two entries in the fifth row of table 1 to (✓) instead of (×), which implies that such a user would have the ability to make use of an identity system regardless of the identity management architecture adopted by the RP or the SP. That is, this integration model is designed to resolve the incompatibility situation that may occur if the IdP is Liberty-enabled and the RP is CardSpace-enabled, or vice versa.

Before describing the proposed integration model, we outline a major difference between the scope of the Liberty ID-FF and the CardSpace framework. The CardSpace framework's main goal is to support the authentication of a user by a trusted third party (i.e. an IdP) and the provision of assertions by the trusted third party to the SP that claims regarding attributes of the user are correct. By contrast, in Liberty, the ID-FF is designed to achieve Identity Federation by asserting to the SP that the user has been successfully authenticated by a trusted third party (i.e. the IdP) using an authentication method, and no claims (i.e. user attributes) are involved in the authentication process. Nevertheless, asserting user attributes by a trusted third party can be achieved using the Liberty protocols; however, this falls under the Liberty ID-WSF, which is a different framework, as discussed earlier in this paper.

The *identity management architecture adaptor* proposed here is a piece of software installed on the user's machine which understands both the Liberty and CardSpace frameworks, and their message flows and formats. The adaptor's main job is to interpose itself between IdPs and SPs/RPs adhering to different identity management architectures, in order to translate particular messages generated by one party to the other. We consider operational details of the identity management architecture adaptor later in this section.

Before presenting the integration model and its message flow, we note the following restrictions on its operation:

1. Given that the Liberty specifications are based only on SAML-Assertion tokens, only SAML-Assertion tokens are permitted within the integration model, i.e. other security tokens supported by CardSpace (e.g. Kerberos v5 tickets) are not permitted. The SAML-Assertion tokens will simply be forwarded from the IdP to the SP/RP.
2. For the proof of rightful possession of the security token, only asymmetric techniques are permitted. In the asymmetric proof technique, the IdP inserts the public key of the User Agent in the security token, so that the User Agent can use its private key to prove rightful possession of the token to the RP (or SP). There is a clear resemblance between the CardSpace *asymmetric* proof-key technique [8], and SAML *holder-of-key* confirmation method [9]; hence, mapping between these techniques is viable using a relatively simple process.
3. For a CardSpace-enabled RP and a Liberty-enabled IdP, the identity management architecture adaptor will discard any token freshness restrictions requests imposed by the RP, since the Liberty-enabled IdP may not be capable of understanding them.

In the CardSpace framework, the claims to be asserted in the RP Security Policy are represented as attributes of an Attribute Statement, that is contained within SAML-Assertion requests and responses exchanged before the security token can be retrieved from the IdP. However, in the Liberty ID-FF, the IdP does not expect any SAML Attribute Statements in the *AuthenticationRequestEnvelope*, i.e. the SOAP envelope that carries the authentication requests. This presents a problem that must be solved before the Identity Management Architecture Adaptor can convert the relevant messages. We propose two possible solutions to this problem.

1. We could convert the claims in the CardSpace Security Policy into attributes within a SAML Attribute Statement in the *AuthenticationRequestEnvelope*. However,

for this to work, we must ensure that the IdP is able to process such statements in order to assert them in the *AuthenticationResponseEnvelope*. However, this solution goes outside the Liberty standards, and would potentially require the Liberty-Enabling software component to be modified.

2. Alternatively, we could make the integration model only accept CardSpace Security Polices with no claims listed, and hence the RP that issues the polices will only require an assertion that the user has been authenticated by a trusted third party. However, this solution will severely impact on the usability of the integration model.

Figure 5 shows the message flow for the integration model in the case where the IdP is CardSpace-enabled and the SP is Liberty-enabled. Note that the Liberty-Enabling component must be active on the Principal's PC prior to performing the protocol, and the CardSpace component must be active on the Principal's PC prior to step 3 of the protocol. Here, the message flow would be:

1. **User Agent → SP** : Service Request (HTTP Request with Liberty Enabled Header)
2. **SP → User Agent** : Authentication Request + "optionally" an IdPs List
   - [The identity management architecture adaptor converts the Liberty `Authentication Request`, received from the SP, into a CardSpace `RP Retrieved Security Policy`, and forwards it to the CardSpace-enabling component]
3. **User Agent ↔ User** : User Picks an InfoCard
4. **User Agent ↔ IdP** : User Authentication
5. **User Agent ↔ IdP-STS** : User Agent retrieves security token via *WS-MetadataExchange* and *WS-Trust*
   - [The identity management architecture adaptor converts the CardSpace `Retrieved Security Token`, received from the IdP-STS, into a Liberty `Authentication Response`, and forwards it to the Liberty-enabling component]
6. **User Agent → SP** : Authentication Response + SAML-Assertion (within the HTML Form)
7. **SP → User Agent** : Service Granted!

Figure 6 shows the message flow for the integration model in the case where the IdP is Liberty-enabled and the RP is CardSpace-enabled. Note that the Liberty-Enabling component must be active on the Principal's PC prior to step 4 of the protocol, and the CardSpace component must be active on the Principal's PC prior to step 3 of the protocol. Here, the message flow would be:

1. **User Agent → RP** : HTTP GET Login HTML Page Request
2. **RP → User Agent** : HTML Login Page + CardSpace Tags (XHTML or HTML object tags)
3. **User Agent ↔ RP-STS** : User Agent retrieves Policy via *WS-SecurityPolicy*
   - [The identity management architecture adaptor converts the CardSpace `RP Security Policy`, received from the RP, into a Liberty `Authentication Request`, and forwards it to the Liberty-Enabling component]
4. **User Agent or User** : Selects the IdP to be used
5. **User Agent → IdP** : SAML-Assertion Request
6. **IdP → User Agent** : SAML-Assertion Response
   - [The identity management architecture adaptor converts the Liberty `Authentication Response`, received from the IdP, into a CardSpace `IdP-STS Retrieved Security Token`, and forwards it to the CardSpace component]
7. **User Agent → RP-STS** : User Agent presents the security token via *WS-Trust*
8. **RP → User Agent** : Welcome, you are now logged in!

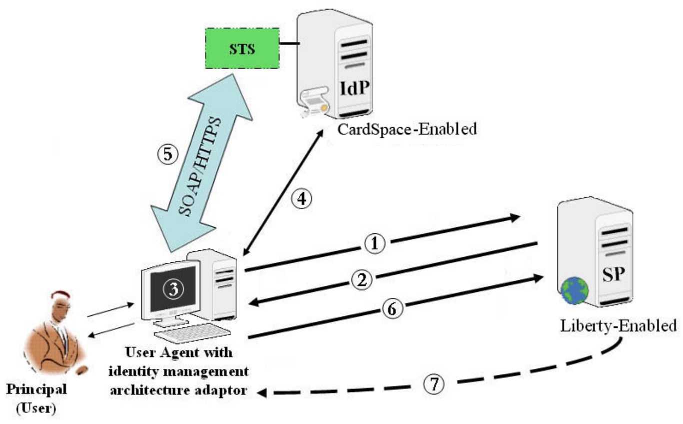

Figure 5: Message flow within the integration model (a).

As described above, the identity management architecture adaptor interposes itself between the IdP and the SP/RP in order to translate messages at certain stages of the protocol run. The identity management architecture adaptor must be able to make four types of message conversion, including two token forwarding operations. Figure 7 shows the message types that need to be converted by the identity management architecture adaptor, along with the respective message formats.

Thus, the identity management architecture adaptor software must perform two types of task:

- Convert the formats of four types of message.
- Forward the converted messages to either liberty enabling or CardSpace software components.

For the first task, implementing the required message conversions should not be difficult, since all the messages formats are open and published (XML definitions). However, the second task is not so straightforward, since it requires a well-defined set of APIs in order for the identity management architecture adaptor software to communicate with the Liberty-Enabling and CardSpace components. The precise

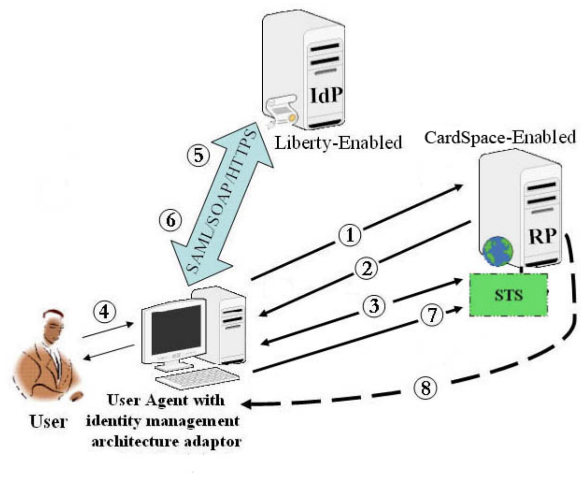

Figure 6: Message flow within the integration model (b).

details of the operation of the adaptor will therefore depend on how these components are implemented. Indeed, it is possible that the adaptor could be integrated into the respective components.

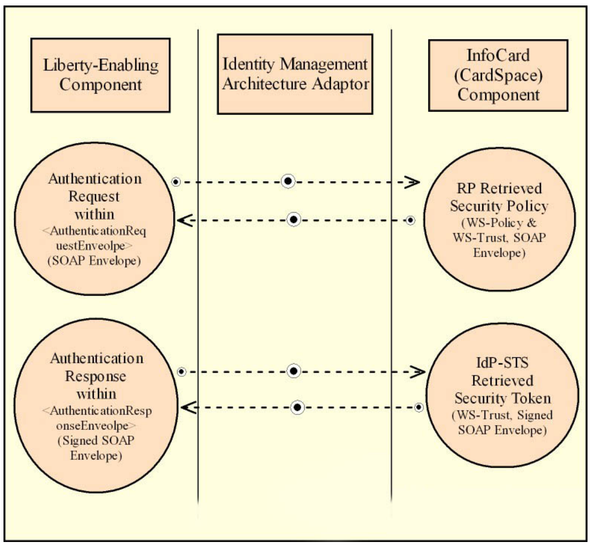
Liberty-Enabling Component
Identity Management Architecture Adaptor
InfoCard (CardSpace) Component
Authentication Request within <AuthenticationRequestEnveolpe> (SOAP Envelope)
RP Retrieved Security Policy (WS-Policy & WS-Trust, SOAP Envelope)
Authentication Response within <AuthenticationResponseEnveolpe> (Signed SOAP Envelope)
IdP-STS Retrieved Security Token (WS-Trust, Signed SOAP Envelope)

Figure 7: Message format conversions.

4. AN ANALYSIS OF THE INTEGRATION MODEL

The integration model takes advantage of the similarity between the ID-FF LEC profile and the CardSpace framework, and this should help to reduce the effort required for full system integration. Moreover, the proposed integration model is designed to be implemented without the need for technical cooperation between Microsoft and Liberty.

Implementing such a model might be non-trivial task, but the benefits could be significant. If interoperability between leading identity management systems is not supported, then this could be a major obstacle to the global adoption of such schemes.

The proposed model is designed to integrate two frameworks with somewhat different scopes. This difference in scope has given rise to the most serious obstacle facing this integration model, namely the dilemma of translating the CardSpace Security Policy claims. Neither of the two solutions proposed in this paper are ideal, since they either involve reducing the applicability of the scheme, or making modifications to the core of the Liberty-Enabling software component.

One potential limitation of the proposed model is that the user agent must be both CardSpace-enabled and Liberty-enabled. However, Windows Vista user agents (i.e. browsers) will, by default, be CardSpace-enabled, and to make the browsers Liberty-enabled will simply require the installation of certain java scripts on user machines; as a result this does not seem to be a major issue. Another possible limitation of the proposed model is the added restrictions on the token type, encryption and freshness requests; these restrictions will prevent the users from utilising certain features offered by CardSpace. However, these restrictions only affect the CardSpace framework because, from the Liberty perspective, token handling will remain the same [10].

A further possible limitation is the necessity for interactions between the adaptor and the CardSpace metasystem; because of the closed nature of CardSpace, this might not be straightforward, unless a well-defined set of APIs is publicly available. Finally, use of the proposed integration model will result in a delay at the user system while the identity management architecture adaptor performs the necessary conversions; however, any such delay is likely to be very small.

5. RELATED WORK

Recently, the Bandit[4] project, which is part of the Higgins[5] project, has developed an open source integration system between Liberty and CardSpace (a demo is available at the project's web site). Although the source code is provided, it seems that there is no published specification of the integration system, which makes it difficult to discover exactly how the Bandit scheme works.

One obvious difference between the integration scheme we propose in this paper and the Bandit integration scheme relates to the location of the integration adapter. Unlike the scheme discussed above, in Bandit the integration adapter is placed on the RP (or SP), not on the user machine. We believe that placing the integration adapter on the user machine increases the usability of the scheme. It is much simpler for a user to deploy an integration scheme, and it seems likely that many well-known service providers will only support a single identity management system for operational and commercial reasons.

6. CONCLUSIONS

In this paper we have proposed a model enabling integration of Liberty and CardSpace. This integration model takes advantage of the similarity in the message flows between the Liberty ID-FF LEC framework and the CardSpace framework. The proposed integration model is based on a client-side identity management architecture adaptor that converts

[4]http://www.bandit-project.org
[5]http://www.eclipse.org/higgins

the format of messages within the message flows of the two schemes. We have also presented an analysis of the main limitations and benefits of our proposed integration model.

The model is designed to integrate two frameworks with somewhat different scopes, and this has caused certain technical problems, in particular in translating the CardSpace claims. In this paper we have suggested possible solutions for such problems.

We believe that enabling interoperation between the two most prominent identity federation architectures could be of major benefit to both the users and producers of these systems.

7. REFERENCES

[1] K. Cameron. The laws of identity, May 2005. Microsoft Corporation.

[2] K. Cameron and M. B. Jones. Design rationale behind the identity metasystem architecture, February 2006. Microsoft Corporation.

[3] S. Cantor, J. Kemp, and D. Champagne (editors). Liberty ID-FF bindings and profiles specification — 1.2-errata-v2.0, 2004. Liberty Alliance Project.

[4] M. B. Jones. A guide to supporting InfoCard v1.0 within web applications and browsers, March 2006. Microsoft Corporation.

[5] S. Kellomai (editor). Liberty ID-SIS employee profile service specification — version: 1.0, 2003. Liberty Alliance Project.

[6] S. Kellomai (editor). Liberty ID-SIS personal profile service specification — version: 1.0, 2003. Liberty Alliance Project.

[7] Microsoft Corporation. A technical reference for InfoCard v1.0 in windows, August 2005.

[8] Microsoft Corporation and Ping Identity Corporation. A guide to integrating with InfoCard v1.0, August 2005.

[9] R. Monzillo, C. Kaler, A. Nadalin, and P. Hallem-Baker (editors). Web Services Security: SAML Token Profile 1.1, February 2006. OASIS Standard Specification, OASIS Open.

[10] P. Thompson and D. Champagne (editors). Liberty ID-FF implementation guidelines — version 1.2, 2004. Liberty Alliance Project.

[11] J. Tourzan and Y. Koga (editors). Liberty ID-WSF web services framework overview — version: 1.1. Liberty Alliance Project.

[12] T. Wason (editor). Liberty ID-FF architecture overview — version: 1.2. Liberty Alliance Project.