REST/SOAP Harmonization proposal for Identity-based Web-Services

Version: 0.4

Date: 2012-10-16

Editor: Gaël Gourmelen, Orange

Contributors:

Fulup Ar Foll, Oracle
Ingo Friese, Deutsche Telekom AG
Jonas Högberg, EricssonKeith Uber, Ubisecure Solutions, Inc.

Status: This document is a **Kantara Initiative Draft Report** that has been approved by the Telecommunications Identity WG/DG (see section 3.9 and 4 of the Kantara Initiative Operating Procedures)

Abstract:

The aim of this document is to propose a solution in order to provide an easy and consistent way for both Web Service Clients and Web Service Providers to respectively invoke or expose Identity-based APIs through both REST and SOAP flavors, taking into account both legacy aspects and growing adoption of the OAuth standard.

Filename: kantara-draft-report-tiwg-REST-SOAP-harmonization-proposal-0.4

Notice:

Contents

1 INTRODUCTION

REST (Representational State Transfer) and SOAP (Simple Object Access Protocol) are two different approaches for the implementation of Web Services.

REST is Resource-oriented whereas SOAP is Activity-oriented. The type of application and the service it offers determines if REST or SOAP is more suitable ; *though one can still argue that we can use SOAP or REST indifferently for these two kinds of services (with a bit of tweaking)*.

To acknowledge the fact that both approaches can still make sense, here are some criteria that clearly distinguish in which case REST or SOAP is still more appropriate:

REST may be appropriate when	**SOAP** may be appropriate when
▪ The Web Services are completely stateless. ▪ A caching infrastructure can be leveraged for performance, and the service is to a large extent static. ▪ The interface can be exposed through standard CRUD operations (Create, Read, Update, and Delete). ▪ The Web Service Client and the Web Service Provider have a mutual understanding of the context and content being passed along. ▪ Client applications are browser-based implementations (e.g. based on AJAX).	▪ The Web Services are stateful and dynamic. ▪ A formal contract must be established to describe the interface that the Web Service offers (WSDL). ▪ Advanced security patterns (including but not limited to end-to-end message-level security) are required. ▪ The architecture must address complex nonfunctional requirements such as Transaction, Security, Addressing, Trust, Coordination and so on. With REST, developers must build this plumbing into the application layer themselves. ▪ Operations (actions) are specific to the service and go beyond basic CRUD operations. ▪ The architecture needs to handle asynchronous processing and invocation.

Even if REST is more and more used mainly as it is simpler to implement, the characteristics of SOAP (extreme definition and data type declaration with XML Schemas – *type, value ranges, etc*) correspond to what we are used to in telecom standards (e.g.: OMA Parlay X APIs, OMA SUPM, 3GPP GUP, …). That explains why some Telco APIs are still specified in either or both flavors.

Legacy aspects will also lead to situations where telecommunication operators will expose both REST and SOAP APIs (e.g.: some Orange APIs opened to partners such as Billing are still SOAP APIs whereas others such as User Profile are REST APIs).

In the case of Identity-based Web Services, the support of both REST and SOAP APIs brings however more complexity for both Web Service Providers and Web Service Clients if they need to support different Identity-based Web Services frameworks to handle common functions related to identity management, security, authorization... These functions are required to ensure that the access to the exposed resources is well-authorized for the requesting Web Service Client, acting on behalf of an end-user.

In the SOAP area, frameworks such as Liberty ID-WSF provide protocols and core components (ID-WSF Discovery Service and Interaction Service notably) to handle all these aspects in conjunction with a Federation Framework.

In the REST area, the OAuth specifications handle these aspects through the delivery of an Access Token delivered to an authenticated Web Service Client upon approval by the end-user.

As OAuth is today more and more adopted in the REST area[1] (more than ID-WSF in the SOAP area), the aim of this document is to describe how it can also be used to secure the access to SOAP APIs and thus providing an easy and consistent way for both Web Service Clients and Web Service Providers to respectively invoke or expose Identity-based APIs through both REST and SOAP flavors.

[1] Important actors like Facebook, Google, Microsoft, Twitter, and Yahoo already deployed OAuth-compliant APIs.

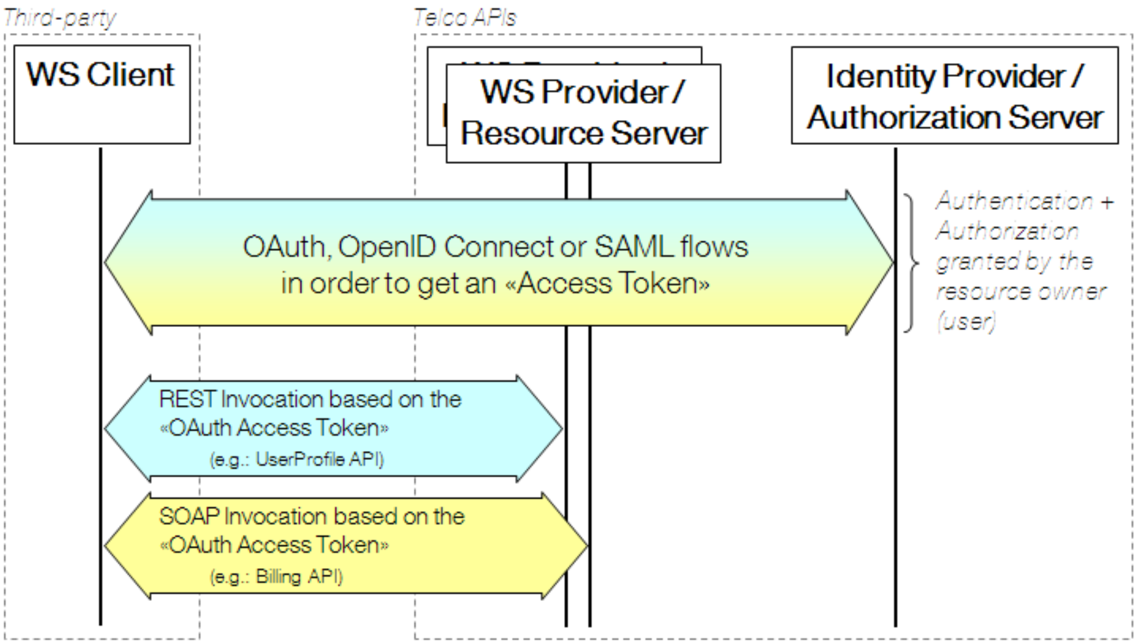
2 PROPOSAL

2.1 Principles

The proposal is here to rely on OAuth mechanisms to allow the user to control the access to his/her exposed resources and grant authorizations to requesting services (Web Service Clients) in both REST and SOAP contexts[2].

Concretely, a WS Consumer/Client has to implement the protocol flows defined in [OAuth2] (or [OpenIDConnect] or [OAuth2Saml2]) to obtain an «OAuth Access Token».

These tokens represent an authorization issued to the WS Consumer/Client with specific scopes (potentially multiple APIs exposed by the telecommunication operator in our context) and durations of access, granted by the resource owner (user), and enforced by the resource server and authorization server.


Third-party
Telco APIs
WS Client
WS Provider / Resource Server
Identity Provider / Authorization Server
OAuth, OpenID Connect or SAML flows in order to get an «Access Token»
Authentication + Authorization granted by the resource owner (user)
REST Invocation based on the «OAuth Access Token» (e.g.: UserProfile API)
SOAP Invocation based on the «OAuth Access Token» (e.g.: Billing API)

[2] Note that a proposal also exists to extend the usage of the OAuth framework for non-HTTP-based protocols: http://tools.ietf.org/id/draft-mills-kitten-sasl-oauth-04.txt. This can be seen as complementary to the approach proposed in our document for REST and SOAP APIs in order to provide even further harmonization between HTTP-based and non-HTTP-based protocols.

This token is then conveyed in both REST (as specified in [OAuth2]) and SOAP calls. For SOAP calls, the proposal is to convey the OAuth Access token in a <wsse:Security> SOAP header as profiled in the following chapter (only OAuth2 Bearer Access tokens are considered at this stage). This would be the minimal step in order to be able to reuse standard XML Signature mechanisms to securely bind the OAuth Access Token to the SOAP message. A further step would be to support the ID-WSF Basic SOAP Binding [LIB-Basic-SOAP] to benefit from additional messaging-specific features.

2.2 WS-Security OAuth Access Token profile

The <wsse:BinarySecurityToken> element is introduced in the “WSS: SOAP Message Security” [WSS] document as a way of conveying any encoded binary security token in a <wsse:Security> SOAP header.

The use of this element to convey OAuth Bearer Access tokens mainly requires the definition of a new value ("**#OAuth2-Bearer**" – *standard value and associated namespace to be defined in relevant standard organization, for example OASIS*) for its ValueType attribute in order to clearly distinguish OAuth Bearer Access tokens from other types of binary tokens.

```
<wsse:Security mustUnderstand="1">
   <wsu:Timestamp wsu:Id="ts">
      <wsu:Created>2011-05-17T04:49:17Z</wsu:Created >
   </wsu:Timestamp>
   <wsse:BinarySecurityToken ValueType="#OAuth2-Bearer"
      EncodingType="wsse:Base64Binary">7Fjfp0ZBr1KtDRbnfVdmIw</wss
e:BinarySecurityToken>
</wsse:Security>
```

Depending on agreements between Web Service Client and Web Service Provider, the exchanged SOAP messages can be integrity protected by implementing the signature mechanisms defined in [WSS].

2.3 Use of the ID-WSF Basic SOAP Binding

The ID-WSF Basic SOAP Binding [LIB-Basic-SOAP] provides a profile that is intended to be a basic, scaled-down version of the Liberty ID-WSF 2.0 SOAP Binding Specification and Security Mechanisms 2.0.

As specified in [LIB-Basic-SOAP], the following header blocks MUST be included in the SOAP header:

- <wsa:MessageID>
- <wsa:RelatesTo> (mandatory on response)
- <wsa:Action>
- <sbf:Framework>
- <wsse:Security>

The following headers MAY be included in the SOAP header:

- <wsa:To>

[LIB-Basic-SOAP] can be used as a basis to define Identity-based SOAP Web Services except that, in our context, **it MUST also support the WS-Security OAuth Access Token profile** defined above.

3 CONCLUSION

This document proposes a simple solution in order to provide an easy and consistent way for both Web Service Clients and Web Service Providers to respectively invoke or expose Identity-based APIs through both REST and SOAP flavors. It enables APIs providers to rely on OAuth to secure the access to their APIs in a uniform way with minimal impacts on existing SOAP APIs (legacy aspects).

4 REFERENCES

4.1 Informative

[OAuth2] Hammer-Lahav, E., Recordon, D., and D. Hardt, "The OAuth 2.0 Authorization Protocol", draft-ietf-oauth-v2-23 (work in progress), January 2012.

[OpenIDConnect] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Standard 1.0,” December 2011.

[OAuth2Saml2] Mortimore, C., "SAML 2.0 Bearer Assertion Profiles for OAuth 2.0", draft-ietf-oauth-saml2-bearer-08 (work in progress), August 2011.

[WSS] “Web Services Security: SOAP Message Security 1.1”, OASIS Standard, 1 February 2006.

[LIB-Basic-SOAP] “Liberty ID-WSF Basic SOAP Binding Specification”, version 1.0, Liberty Alliance Project

Revision History

0.1	Initial draft
0.2	Integration of comments received from the Kantara Initiative Telecommunication Identity Work Group
0.3	Additional comments from the Kantara Initiative Telecommunication Identity Work Group
0.4	Approval by the Kantara Initiative Telecommunication Identity Work Group