International

Virtual

Observatory

Alliance

IVOA Single-Sign-On Profile: Authentication Mechanisms

Version 2.0

IVOA Working Draft 2015 June 11

This version:

2.00-20150527

Latest version:

http://www.ivoa.net/Documents/latest/SSOAuthMech.html

Previous version(s):

http://www.ivoa.net/documents/cover/SSOAuthMech-20080124.html

http://www.ivoa.net/documents/cover/SSOAuthMech-20070904.html

http://www.ivoa.net/documents/cover/SSOAuthMech-20070621.html

http://www.ivoa.net/documents/cover/SSOAuthMech-20060519.html

Working Group:

http://www.ivoa.net/twiki/bin/view/IVOA/IvoaGridAndWebServices

Editor(s):

Guy Rixon, André Schaaff, Giuliano Taffoni

Author(s):

Grid and Web Services Working Group

Abstract

Approved client-server authentication mechanisms are described for the IVOA single-sign-on profile: No Authentication; HTTP Basic Authentication; TLS with passwords; TLS with client certificates; Cookies; Open Authentication; Security Assertion Markup Language; WebID. Normative rules are given for the implementation of these mechanisms, mainly by reference to pre-existing standards. The Authorization mechanisms are out of the scope of this document.

Status of This Document

This document has been produced by the IVOA Grid and Web Services Working Group.

SSO Profile V2.0 deprecates the SOAP-oriented authentication methods and introduces new ones (RESTful-oriented), also naming conventions has been changed.

A list of current IVOA Recommendations and other technical documents *can be found at http://www.ivoa.net/Documents/.*

Acknowledgements

This document derives from discussions among the Grid and Web Services working-group of IVOA. It is particularly informed by prototypes built by Matthew Graham (Caltech/US-NVO), Paul Harrison (ESO/EuroVO), David Morris (Cambridge/AstroGrid) and Raymond Plante (NCSA/US-NVO). The prior art for the use of proxy certificates comes from the Globus Alliance.

This document has been developed with support from the National Science Foundation's Information Technology Research Program under Cooperative Agreement AST0122449 with The Johns Hopkins University, from the UK Particle Physics and Astronomy Research Council (PPARC) and from the Europena Commission’s Sixth Framework Program via the Optical Infrared Coordination Network (OPTICON).

Definitions

The **Virtual Observatory (VO)** is general term for a collection of federated resources that can be used to conduct astronomical research, education, and outreach. The **International Virtual Observatory Alliance (IVOA)** is a global

collaboration of separately funded projects to develop standards and infrastructure that enable VO applications.

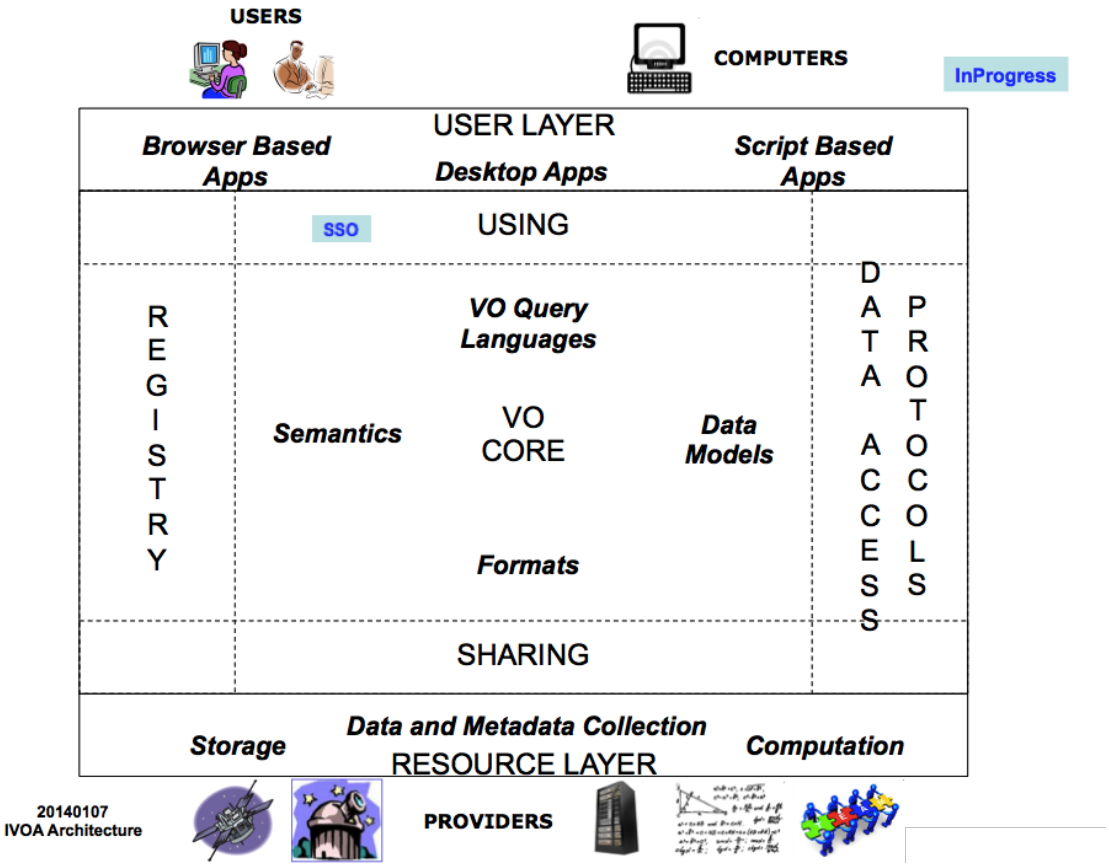
Contents

1 Introduction

IVOA's single-sign-on architecture is a system in which users assign cryptographic credentials to user agents so that the agents may act with the user's identity and access rights. This standard describes how agents use those credentials to authenticate the user's identity in requests to services. This standard describes also the authentication mechanism of an application or a service making a call (on behalf of someone or something else) to an API or to another service

This document is essentially a *profile* against existing security standards; that is, it describes how an existing standard should be applied in an IVOA application to support single sign-on capabilities in the IVOA. In the following sections, we make specific references to details spelled out in these standards. For the purposes of validating against this standard, those referenced documents should be consulted for a full explanation of those details. Unfortunately, a reader that is unfamiliar with these external standards might find this specification confusing. To alleviate this problem, each major section is concluded by a Commentary subsection that provides some explanations of the detailed terms and concepts being referred to. The Commentary subsection may also provide recommended scenarios for how this specification might actually be realized. Note that the statements in the Commentary subsections are non-normative and should not be considered part of precise specification; nevertheless, they are indicative of the intended spirit of this document.

1.1 The Role in the IVOA Architecture



2 Scope of this standard

2.1 Requirements

When a service is registered in an IVOA registry, that service's resource document may include metadata expressing conformance to one or more of the authentication mechanisms approved in the IVOA SSO profile. Such a service must implement those mechanisms as described in this document, and clients of the service must participate in the mechanism when calling the service. Is a service does not provide any SSO specification it is assumed that no authentication is required.

The registration of the service interface shall contain an XML element of type *SecurityMethod* as specified in the XML schema for VOResource [VOResource]. The value of this element distinguished the authentication mechanism using the values stated in the sections below.

Services registered without the metadata alluded to above need not support any authentication mechanism. If they do require authentication, they may use either the IVOA-standard mechanisms or others that are not IVOA standards.

2.2 Commentary

The IVOA SSO profile allows the development of a “realm” of interoperable services and clients. Service providers opt in to this realm by implementing this current standard and by registering accordingly in the IVOA registry. This allows clients to discover a secured service through the registry and to be able to use it without being customized for the details of the specific service.

Parts of the IVOA that are not intended to be widely interoperable need not opt in to the SSO realm. In particular, “private” services, accessed by web browsers and protected by passwords, are allowed. However, these private services should be reworked to follow the IVOA standard if they are later promoted to a wider audience.

An example of a registration for a secured interface follows.

```
<interface xmlns:vs:=”ivo://www.ivoa.net/xml/VODataService/v1.0”
      xsi:type=”vs:ParamHTTP”>
 <accessURL>http://some.where/some/thing</accessURL>
 <securityMethod>ivo://ivoa.net/sso/saml2.0</securityMethod>
</interface>
```

3 Approved authentication mechanisms

3.1 Requirements

The following authentication mechanisms are approved for use in the SSO profile.

- No authentication required.
- HTTP Basic Authentication
- Transport Layer Security (TLS) with passwords.
- Transport Layer Security (TLS) with client certificates.
- Cookies
- Open Authentication (OAuth)
- Security Assertion Markup Language (SAML)
- WebID – (to be discussed)

The mechanism is associated with the interface provided by the service and registered in the IVOA registry.

SSO mechanism	<securityMethod>
No authentication required.	none
HTTP Basic Authentication	http//www.w3.org/Protocols/HTTP/1.0/spec/html#BasicAA
TLS with password	ivo://ivoa.net/sso/tls-with-password
TLS with client certificate	ivo://ivoa.net/sso/tls-with-certificate
Cookies	ivo://ivoa.net/sso/cookie
Open Authentication	ivo://ivoa.net/sso/OAuth
SAML	ivo://ivoa.net/sso/saml2.0
WebID	ivo://ivoa.net/sso/webid

Services that are registered with a IVOA registry as having a *WebService* type interface [VOResource] shall support OAuth, or shall support cookies or shall support TLS with client certificates or shall require no authentication.

Interfaces by which a user logs in to the SSO system shall support either TLS with client certificates, or TLS with passwords, or SAML or a combination of the them.

3.2 Commentary

Services with interface type *WebService* are RESTfull services. RESTFull web service authentication is commonly done using stateless mechanisms (e.g. OAuth, x509 etc.). TLS with client certificate mechanisms allow the service to verify that the client holds the private key matching a certificate transmitted with the message. Authentication succeeds if the service trusts the issuer of the certificate. That trust is determined by reference to a set of certificates for trusted certificate authorities (CAs) configured into the service by the service provider.

4 HTTP Basic Authentication

4.1 Requirements

Services using HTTP basic authentication shall use the athentication mechanism described in the RFC 7235 that updates RFC2617. Interfaces using this mechanism shall be be registered with the security method http://www.w3.org/Protocols/HTTP/1.0/spec/html#BasicAA

4.2 Commentary

HTTP provides a simple challenge-response authentication framework that can be used by a server to challenge a client request and by a client to provide authentication information.

The HTTP authentication framework does not define a single mechanism for maintaining the confidentiality of credentials. HTTP depends on the security properties of the underlying transport- or session-level connection to provide confidential transmission of header fields. Connection secured with TLS are recommended prior to exchanging any credentials.

5 Details of TLS

5.1 Requirements

Services using Transport Layer Security (TLS) shall do so according to the TLS v1.2 standard [RFC5246].

5.2 Commentary

TLS supersedes the Secure Sockets Layer that is an outdated cryptographic protocol. TLS v1.0 was based on SSL v3.0; the actual version of TLS is V1.2 described in [RFC5246]. TLS v1.2 is back-compatible with TLS v1.0, TLS v1.1 and SSL v3.0. “TLS versions 1.0, 1.1, and 1.2, and SSL 3.0 are very similar, and use compatible ClientHello messages; thus, supporting all of them is relatively easy.[...] TLS 1.2 clients that wish to support SSL 2.0 servers MUST send version 2.0 CLIENT-HELLO messages defined in [SSL2].”[RFC5246]

6 Details of TLS-with-client-certificate

6.1 Requirements

Certificates shall be transmitted and checked according to the TLS v1.2 standard [RFC5246].

Services implementing TLS must support certificate chains including proxy certificates according to RFC6818 [RFC6818].

Interfaces using this mechanism shall be be registered with the security method *ivo://ivoa.net/sso#tls-with-client-certificate*.

6.2 Commentary

When Mutual Certificate Authentication is configured for REST services, both, the client and the service perform identity verification or authentication through X509 certificates.

The client authenticates the service during the initial SSL handshake, when the server sends the client a certificate to authenticate itself.

7 Details of TLS-with-password

7.1 Requirements

The user-name and password shall be passed in the message protected by the TLS mechanism, not as part of the mechanism itself. The “HTTP basic authentication” should be used with particular attention.

Interfaces using this mechanism shall be be registered with the security method *ivo://ivoa.net/sso#tls-with-password*.

7.2 Commentary

“HTTP basic authentication” passes the user-name and password in the HTTP headers, assuming that the credentials are not a natural part of the message body. This standard applies the TLS-with-Password mechanism only to the special case of logging in to the SSO realm. Hence, the user name and password are logically part of the message body, not the message header.

8 The use of Cookies

8.1 Requirements

Cookie-Based Authentication uses server side cookies to authenticate the user on every request. The way to manage cookies for authentication is described in RFC6265.

8.2 Commentary

RESTful web services should use session-based authentication, either by establishing a session token via a POST or by using an API key as a POST body argument or as a cookie. User names, passwords, session tokens, and API keys should not appear in the URL, as this can be captured in web server logs, which makes them intrinsically valuable.

9 Details on SAML authentication

9.1 Requirements

Services using SAML authentication mechanisms shall shall do so according to the saml-core-2.0-os OASIS standard [SAML2.0]. SAML includes protocols and protocol bindings and security [OASIS-SAMLv2-BIND].

9.2 Commentary

SAML presumes two primary roles in any transaction: the organisation where the identity is established, known as the Identity Provider (“IdP”), or Asserting Party (“AP”); and the organisation which (for this transaction) wants to use this identity, known as the Service Provider (“SP”), or Relying Party (“RP”).

A user attempts to access an application with the Service Provider. The SP needs to establish the identity of this user, and so sends an authentication request to the Identity Provider.

The user authenticate with the IDP (IDP is taking care of the authentication mechanisms and protocols e.g. Kerberos, ldap etc.) so the IdP can send back an ‘Assertion’ to the SP. Now the SP knows who the user is, and can process that user accordingly.

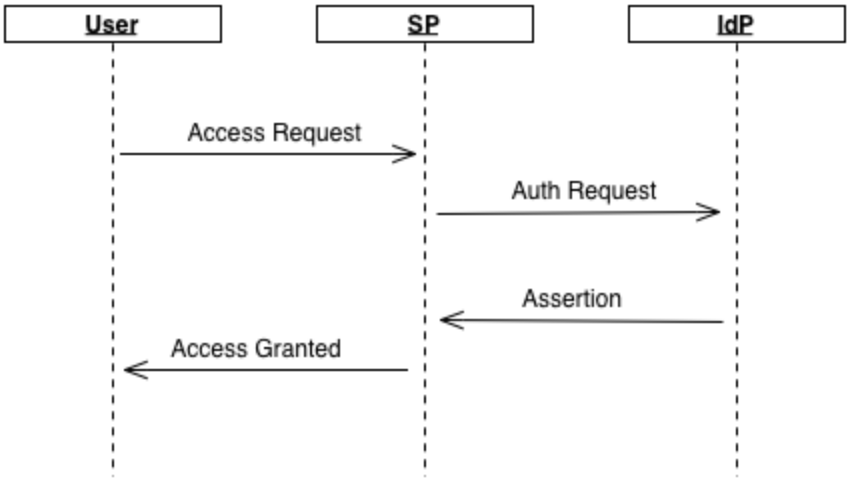


Fig. Simplified picture of SAML 2.0 authentication.

SAML2.0 allow also to service discovery mechanisms.

10 Details on OAuth

10.1 Requirements

Services using OAuth authentication mechanisms shall do so according to the RFC6749.

10.2 Commentary

Open Authentication 2.0 (also in conjunction with OpenID Connect) is actually the adopted standard to handle identity in the framework of RESTful web

services. OAuth is used when an application is making a request on behalf of a user.
OAuth introduces the notion of an 'authorization token', a 'refresh token' and Authorization Service (AS). The 'authorization' token states that the client application has the right to access services on the server. However, it does not supersede any access control decisions that the server-side application might make.

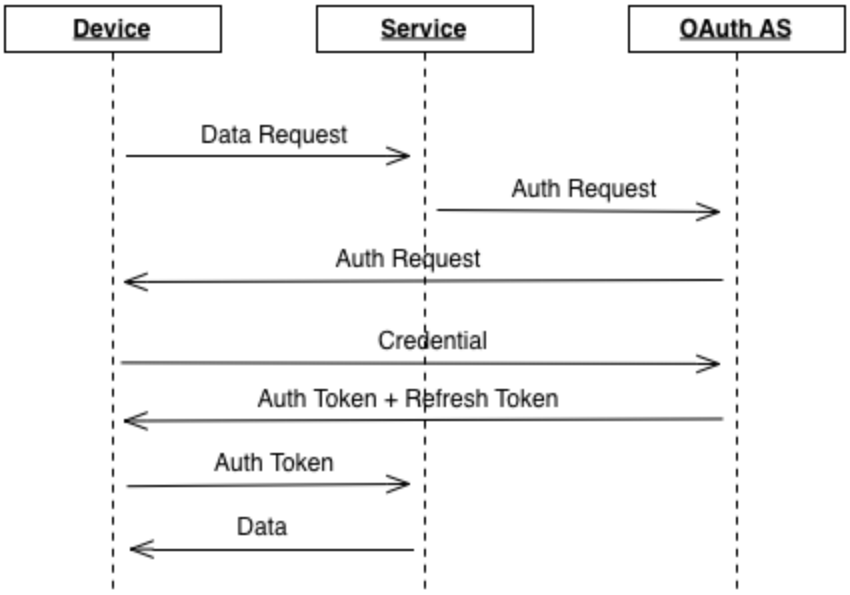


Fig. Simplified picture of OAuth flow.

OAuth is related to delegate credential from an application to another.

8 The WebID

8.1 Requirements

WebID is an open standard for identity and login developed by W3C WebID community group and defined by the W3C 1.0 specification document [WebID].

8.2 Commentary

A WebID is a way to uniquely identify a person, company, organisation, or other agent using a URI. The value of having a URI as an identifier are:

- that it can be linked to by other profiles, to create a linked web of trust

- that it can be tied to information enabling a method of authentication (such as OpenID or even more directly with foaf+ssl)

FOAF is one fairly-commonly used WebID-centered vocabulary, used for profiles of agents (people, organizations, or software).

11 Federated Authentication

Federated authentication addresses the interoperability between identity provides, and it enables organizations to share trusted identities across the boundaries that separate them. In practice the user identity stays safely in the ownership of the relevant organisation; and identity information is exchanged through the use of secure tokens. This provides improved security, reduced risk, and supports modern business practice much more completely, both for business-to-business and business-to-consumer models. Secure tokes can be delegated to services and application to act on behalf of the user.
Common protocols for federated authentication are: SAML, OpenID, SCIM (System for Cross-domain Identity Management), OAuth, OpenID connect, etc

11.1 Commentary

The attribute query profile in SAML allows to request for more attributes
in a corporate context and SAML sounds more appropriate than OAuth for SSO. OAuth is more appropriate for Internet interaction between applications or perhaps applications comprising a Service Oriented Architecture in a large corporate environment.

12 Certificate chains

12.1 Requirements

The certificate chain sent by a client to a service during authentication shall contain only end-entity certificates [RFC2459] and proxy certificates [RFC3820]; both are forms of the X.509v3 certificate format. The transmitted chain shall not include any self-signed certificates. The identity which the client wishes authenticated must be the subject of the first end-entity certificate in the chain.

For the purpose of establishing trust, the service shall be configured with one or more chains of certificates each of which ends with a trust anchor (see section 8.2). These configured chains must contain only end-entity certificates.

To establish trust in the identity with which a message is signed, a service shall concatenate the chain transmitted by the client with one of the configured chains, forming a single chain from the certificate used in the message signature back to

the trust anchor. The service shall then validate the relationship between each two links in the chain according to the IETF rules [RFC3820].

A certificate may contain a limit on the length of the chain of proxy certificates that may be derived from it. The validation of the certificate chain shall respect this limit if it is present.

If the chain is validated successfully, then the service shall deem the chain to authenticate the identity that is the subject of the first end-entity certificate in the chain.

12.2 Commentary

In a chain of certificates, each certificate is signed by the certificate that comes after it, or is self-signed. Self-signed certificates trusted by the service are called “trust anchors”; they are generally certificates issued by certificate authorities (CAs). End-entity certificates are issued on behalf of users (humans or automate) by CAs and are often “permanent” – that is, they have a long lifetime. Proxy certificates are usually short-term credentials (valid typically for a few hours or days).

Proxy certificates are often used to solve two problems. The first is that a user may authenticate herself using a long lived EEC; however, it would not be safe to allow an application to have long-term use of that certificate, particularly if that application is remote. To solve this, a short-lived proxy certificate can be provided to the application to limit its privileges. Note that in some authentication models, the user can authenticate with a delegation service which can provide the application with a certificate. This can be either a short-lived proxy or a short-lived EEC; either will solve the problem.

The other problem is that one service requiring authentication will need to call another service requiring authentication. Solving this problem in the IVOA SSO framework requires the use of proxy certificates. Below is a scenario for how this can be accomplished.

A user agent signs on to the SSO system (the details of the sign-on process are not specified in this document) in a user’s name and receives a chain of two certificates: a proxy and an end-entity certificate. The user-agent holds the private key matching the proxy certificate but not that matching the end-entity certificate. The agent signs messages with the proxy and includes in the message the end-entity certificate in order to complete the chain back to the CA certificate that is the trust anchor.

When service B receives an authenticated request from agent A (signed with a proxy certificate), and when makes an authenticated call to service C while satisfying A’s request, then B derives from A’s proxy a new proxy to which B holds the private key (the protocol for deriving this proxy is specified outside this document). This new proxy is signed using A’s original proxy. Therefore, C

receives a chain of two proxy certificates and one end-entity certificate. If C then calls service D, C sends a chain of three proxies and one end-entity certificate.

Some of the chain-validation rules from RFC3820 can be summarized as follows (this is an incomplete description of the rules and is non-normative for the IVOA SSO profile).

- A proxy certificate may be followed by an end-entity certificate or another proxy certificate.
- An end-entity certificate may not be followed by a proxy certificate.
- There must be exactly one self-signed certificate and it must be the last in the chain: this is the trust anchor.
- Any proxy certificate must carry the *ProxyCertInfo* extension (and must therefore be in X.509v3 format).
- Any end-entity certificate that follows another end-entity certificate must carry the basic-constraints extension with the CA field set to true (and must therefore be in X.509v3 format).
- There must be at most one end-entity certificate that does not have the CA field in the basic-constraints extension set to true. This must be the first end-entity certificate in the chain.

13 Changes since previous version of this document

- All references to “identity certificate” (a Globus term) are changed to “end-entity certificate” (an IETF term defined in RFC3820).
- The form of registration for secured interfaces is specified.

References

[RFC2246] T. Dierks, C. Allen, *The TLS Protocol Version 1.0*, http://www.ietf.org/rfc/rfc2246.txt

[RFC2459] R. Housley, W. Ford, W. Polk, D. Solo, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile,* http://www.ietf.org/rfc/rfc2459.txt

[RFC2617] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, HTTP Authentication: Basic and Digest Access Authentication, http://www.ietf.org/rfc/rfc2617.txt

[RFC3275] D. Eastlake, J. Reagle, D. Solo, *XML-Signature Syntax and Processing*, http://www.ietf.org/rfc/rfc3275.txt

[RFC3546] S. Blake-Wilson, M. Nystrom, D. Hopwood, J. Mikkelsen, T. Wright, *Transport Layer Security (TLS) Extensions*, http://www.ietf.org/rfc/rfc3546.txt

[RFC3820] S. Tuecke, V. Welch, D. Engert, L. Pearlman, M. Thompson, *Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile,* http://www.ietf.org/rfc/rfc3820.txt

[RFC5246] T. Dierks, E. Rescorla, The Transport Layer Security (TLS) Protocol Version 1.2, http://www.ietf.org/rfc/rfc5246.txt

[RFC6265] A. Barth, HTTP State Management Mechanism, http://www.ietf.org/rfc/rfc6265.txt

[RFC6749] D. Hardt, The OAuth 2.0 Authorization Framework, http://www.ietf.org/rfc/rfc6749.txt

[RFC6818] P. Yee, Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revication List (CRL) Profile, http://www.ietf.org/rfc/rfc6818.txt

[RFC7235] R. Fielding, J. Reschke, Hypertext Transfer Protocol (HTTP/1.1): Authentication, http://www.ietf.org/rfc/rfc7235.txt

[WebID] Henry Story and the WebID Incubator Group, WebID specifications, http://webid.info/spec/

[VOResource] Raymond Plante, Kevin Benson, Matthew Graham, Gretchen Greene, Paul Harrison, Gerard Lemson, Tony Linde, Guy Rixon, Aurelien Stebe, *VOResource: an XML Encoding Schema for Resource Metadata, Version 1.02,* http://www.ivoa.net/Documents/cover/VOResource-20061107.html

[WS-Security] Organization for the Advancement of Structured Information Standards, *Web Services Security SOAP Message Security 1.0,* http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf