Open Geospatial Consortium

Publication Date: 2015-08-19

Approval Date: 2015-06-05

Posted Date: 2015-05-14

Reference number of this document: OGC 15-022

Reference URL for this document: http://www.opengis.net/doc/PER/t11-common-security

Category: Public Engineering Report

Editor: Andreas Matheus

Testbed-11 Engineering Report:

Implementing Common Security Across the OGC Suite of Service Standards

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type: OGC® Engineering Report
Document subtype:
Document stage: Approved for public release
Document language: English

License Agreement

Contents

Page

Figures

Page

Tables

Page

Abstract

This OGC Engineering Report (ER) focuses on describing *Common Security* for all OGC Web Service Standards. This work was performed as part of the OGC Testbed 11 activity[1].

Business Value

The ability to describe *Common Security* across OGC Web Services standards enables the interoperable use of service instances to provide and share sensitive, high quality and high value geospatial information. The implementation of interoperable security definitions ensures use of protected OGC Web Services for many domains, including intelligence to commercial.

Keywords

ogcdocs, ogc documents, testbed-11, OGC, OWS, Security

[1] http://www.opengeospatial.org/projects/initiatives/testbed11

Testbed-11 Implementing Common Security Across the OGC Suite of Service Standards

1 Introduction

1.1 Scope

This OGC Engineering Report (ER) focuses on describing *Common Security* for all OGC Web Service Standards.

Before describing existing options, *Common Security* is in the context of OGC Web Services is first defined.

Next, the ER analyzes the methods used by Web Services in general and OGC Web Services regarding the Publish/Find/Bind paradigm. The reader needs to understand that OGC Web Services use a slightly different Publish/Find/Bind approach than the general UDDI approach as outlined in section 5.

In order to describe security constraints on OGC Web Services, the "common ground" for all service specifications needs to be analyzed in order to understand what options are available. In particular, to find at least one option to include security constraints in the service instance description is important. Additionally, in case there were multiple options, explaining when to favor one approach over another as well as shortcomings is important.

Finally, this ER provides options on how to describe security constraints across different OGC Web Services standards, including a discussion of feasibility of the approaches.

This ER concludes with recommendations to OGC standardization to describe *Common Security*.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Andreas Matheus	University of the Bundeswehr

1.3 Future work

Define a charter to form a new OGC Standards Working Group focused on security issues across the OGC standards baseline.

1.4 Forward

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

[1] **OGC 08-176r1**: OGC® OWS-6 Secure Sensor Web Engineering Report

NOTE This OWS-6 ER contains a comprehensive overview to security standards applicable to this ER.

[2] **OGC 06-121r3**: *OGC® Web Services Common Standard*

NOTE This OWS Common Standard contains a list of normative references that are also applicable to this Implementation Standard.

[3] **OGC 04-095**: OpenGIS® Filter Encoding Implementation Specification

[4] **ISO, 35.100**: Open systems interconnection (OSI)

[5] **ISO/IEC 10181-1:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Overview, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24404

[6] **ISO/IEC 10181-2:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Authentication framework, ISO 1996:

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=18198

[7] **ISO/IEC 10181-3:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Access control framework, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=18199

[8] **ISO/IEC 10181-4:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Non-repudiation framework, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=23615

[9] **ISO/IEC 10181-5:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Confidentiality framework, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24329

[10] **ISO/IEC 10181-6:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Integrity framework, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24330

[11] **ISO/IEC 10181-7:** Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Security audit and alarms framework, ISO 1996: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=18200

[12] **HTTP**: RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1 – IETF RFC 2616 (1999): http://tools.ietf.org/html/rfc2616

[13] **HTTP Authentication**: HTTP Authentication: Basic and Digest Access Authentication – IETF RFC 2617 (1999): https://tools.ietf.org/html/rfc2617

[14] **TLS:** Transport Layer Security – IETF RFC 2246 (1999): http://tools.ietf.org/html/rfc2246

[15] **HTTPS:** HTTP Over TLS – IETF RFC 2818 (2000): http://tools.ietf.org/html/rfc2818

[16] **X.509 / PKI**: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks, ITU-T Standard, 08/2005: http://www.ietf.org/html.charters/pkix-charter.html

[17] **URI:** Uniform Resource Identifiers (URI): Generic Syntax – IETF RFC 2396 (1998): https://tools.ietf.org/html/rfc2396

[18] **CRL:** Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile – IETF RFC 3280: http://tools.ietf.org/html/rfc3280

[19] **Cookies**: HTTP State Management Mechanism – IETF RFC 2109: https://tools.ietf.org/html/rfc2109

[20] **Web Services Security**: SOAP Message Security 1.1 (WS-Security 2004) – OASIS Standard Specification, 1 February 2006: http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

[21] **XML Digital Signature:** XML-Signature Syntax and Processing – W3C Recommendation 12 February 2002: http://www.w3.org/TR/xmldsig-core/

[22] **XML Encryption**: XML Encryption Syntax and Processing – W3C Recommendation 10 December 2002: http://www.w3.org/TR/xmlenc-core/

[23] **XML Signature Best Practices**: XML Signature Best Practices – W3C Working Group Note 11 April 2013: http://www.w3.org/TR/2013/NOTE-xmldsig-bestpractices-20130411/

[24] **XLink**: XML Linking Language (XLink) Version 1.0, W3C Recommendation 27 June 2001: http://www.w3.org/TR/xlink/

[25] **WSDL 1.1**: Web Services Description Language (WSDL) 1.1, W3C Note 15 March 2001: http://www.w3.org/TR/wsdl

[26] **WSDL 2.0**: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Recommendation 26 June 2007: http://www.w3.org/TR/wsdl20/

[27] **UDDI**: UDDI Spec Technical Committee Draft, OASIS, Dated 20041019: http://www.uddi.org/pubs/uddi_v3.htm

[28] **WS-Policy**: Web Services Policy 1.5 – Framework, W3C Recommendation 04 September 2007: http://www.w3.org/TR/2007/REC-ws-policy-20070904/

[29] **WS-Policy Attachment**: Web Services Policy 1.5 – Attachment, W3C Recommendation, 04 September 2007: http://www.w3.org/TR/ws-policy-attach/

[30] **WS-SecurityPolicy:** WS-SecurityPolicy 1.2, OASIS Standard, 1 July 2007: http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-os.pdf

[31] **WS-Trust:** WS-Trust 1.3, OASIS Standard, 19 March 2007: http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf

[32] **WS-SecureConversation:** WS-SecureConversation 1.3, OASIS Standard, 1 March 2007: http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-os.pdf

[33] **WS-Reliable Messaging:** Web Services Reliable Messaging (WS-ReliableMessaging) Version 1.2, Committee Draft, 28 February 2008: http://docs.oasis-open.org/ws-rx/wsrm/200702/wsrm-1.2-spec-cd-01.pdf

[34] **WS-RM Policy**: Web Services Reliable Messaging Policy Assertion (WS-RM Policy) Version 1.2, Committee Draft, 28 February 2008: http://docs.oasis-open.org/ws-rx/wsrmp/200702/wsrmp-1.2-spec-cd-01.pdf

[35] **WS-MakeConnection:** Web Services Make Connection (WS-MakeConnection) Version 1.1, Committee Draft, 28 February 2008: http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-cd-01.pdf

[36] **WS-Federation / WS-Authorization / WS-Privacy:** Web Services Federation Language (WS-Federation) Version 1.2, Editors Draft – 06, May 21, 2008: http://www.oasis-open.org/committees/download.php/28360/ws-federation-1.2-spec-ed-06.doc

[37] **WS-MetadataExchange:** Web Services Metadata Exchange (WS-MetadataExchange), Version 1.1, August 2006, Microsoft, IBM, Sun and SAP: http://specs.xmlsoap.org/ws/2004/09/mex/WS-MetadataExchange.pdf

[38] **WS-Transfer:** Web Services Transfer (WS-Transfer), W3C Member Submission, 27 September 2006: http://www.w3.org/Submission/WS-Transfer/

[39] **WS-RT:** Web Services Resource Transfer (WS-RT), Version 1.0, August 2006: http://schemas.xmlsoap.org/ws/2006/08/resourceTransfer/WS-ResourceTransfer.pdf

3 Conventions

3.1 Abbreviated terms

DTD	Document Type Definition
ECP	(SAML) Enhanced Client Proxy Profile

GeoXACML	Geospatial eXtensible Access Control Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol over SSL/TLS
INSPIRE	INfrastructure for SPatial InfoRmation in Europe
PKI	Public Key Infrastructure
OAuth	OAuth
OWS	OGC Web Services
RFC	Request For Comments
SAML	Security Assertion Markup Language
SOS	OGC Sensor Observation Service
SPS	OGC Sensor Planning Service
SOAP	SOAP
TLS	Transport Layer Security
UDDI	Universal Description Discovery and Integration
WFS	OGC Web Feature Service
WMS	OGC Web Map Service
WMTS	OGC Web Map Tiling Service
WSDL	Web Services Description Language
WPS	OGC Web Processing Service
WS-*	Web Services Security (all sub topics included)
XACML	eXtensible Access Control Markup Language
XLink	XML Linking Language

3.2 Used parts of other documents

This document uses significant parts of other OGC documents. To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, the largely copied parts are shown with a light grey background (5%).

4 Implementing Common Security overview

4.1 Attempt to define Common Security

The main objective of this ER is to describe *Common Security* for OGC Web Service standards. This can only be achieved if the generic term *Common Security* is defined.

During past OGC Testbeds, a common understanding of the requirements regarding Common Security was established. We explicitly base this report on that definition from the OWS-6 ER ([1], section 6.2). Please note that the definition from [1], section 6.2 is using the security framework concept outlined in ISO 10-181 (all parts). ISO 10-181[2] introduces a flexible security framework for open interconnected systems that can be implemented in part or in full.

Based on that definition, *Common Security* would require describing the constraints towards the following security requirements:

***Authentication Framework**: ISO 10181-2 defines all basic concepts of authentication in Open Systems: It identifies different classes of authentication mechanisms, the services for their implementation and the requirements for supporting protocols. It further identifies requirements for the management of identity information.*

***Access Control Framework**: ISO 10181-3 defines all basic concepts for access control in Open Systems and the relation to other frameworks such as the Authentication and Audit Frameworks.*

***Non-repudiation Framework:** ISO 10181-4 refines and extends the concepts of non-repudiation, given in ISO 7598-2. It further defines general non-repudiation services and the mechanisms to provide these services.*

***Confidentiality Framework**: ISO 10181-5 defines the basic concepts of confidentiality, identifies classes of confidentiality mechanisms and their maintenance. It further addresses the interactions of the confidentiality mechanisms with other services.*

***Integrity Framework**: ISO 10181-6 defines the basic concepts of integrity, identical to the Confidentiality Framework.*

***Security Audits and Alarms Framework**: ISO 10181-7 defines the basic concepts for security audit and alarms and the relationship to other security services.*

[2] ISO 10181-**1** describes an overview regarding the security framework defined in parts 2 – 7.

4.2 Applicability of the Frameworks for *Common Security* and OGC Web Services

Because the goal of this ER is to describe *Common Security* for OGC Web Services, knowing which of the frameworks above need to be implemented and in particular which of the frameworks have implications to the client application accessing an OGC based Web Service or the "calling client" is important.

Guidance for describing *Common Security* is only necessary for those frameworks that are (i) to be implemented and (ii) relevant for the calling client to know about and understand is reasonable.

Authentication Framework: This framework can be implemented standalone but is typically required by other frameworks such as Access Control and Security Audits and Alarms Framework. Common use cases require its implementation for enabling (i) access control based on user characteristics and (ii) accountability such as auditing / logging of user information with a request / response.

In the context of OGC Web Services, this framework is relevant and has direct implication to the calling client, as the execution of the service will only take place after successful authentication.

Access Control Framework: This framework could be implemented standalone. If implemented without the Authentication framework, this does mean in particular that deriving access decisions must be undertaken without user information. For Web Services, the typical information available is coming from the computing environment: HTTP protocol, server hostname, request URL, service operation and parameters, date & time, requested resources, and IP address of the client. Typically, the Authentication framework is implemented and provides user information. Then, authorization decisions can be derived based on user information and the computing environment.

In the context of OGC Web Services, the Authentication framework is relevant and has indirect implication to the calling client. Indirect means that the calling client may be able to execute the service endpoint but then the attempt of executing the actual service with provided parameters may result in access denied.

As an example, a simple implementation of the Authentication Framework might imply access control to a Web Server path: the URI to access a service. Assuming the access control framework is implemented for /protected/ogc/service then the client is either able to bind to the service or not. In other words, the client is either able to execute any operation of the service (e.g. GetCapabilities, GetMap, GetFeatureInfo, etc.) or no operation at all. This is because all service operations are part of the query parameter that comes after the URI separator "?".

A more sophisticated example would be that the implementation of the access control framework is based on a complex GeoXACML policy that introduces fine-grained access control on subject attributes, service operation, resource characteristics and environment

information such as IP address of the client and the date. In this example, releasing the actual policy is important to the client as any subsequent execution attempt may result in "not authorized".

Non-repudiation Framework: This framework can be implemented standalone and independent from the other frameworks. This framework ensures for transactional services that an adversary cannot repeat the execution of an approved operation. The existence of this framework is usually not communicated to the calling client.

In the context of OGC Web Services, the non-repudiation framework is not relevant, as the strict concept of transactions is not supported by OGC Web Services. Even though some OGC Web Services provide a write interface, and for the Web Feature Services this interface is named Transactional WFS, this interface provides create/delete/modify and not transactional operations. However, if a particular WFS-T instance shall support transactions, then the implementation of this framework does make sense.

Confidentiality Framework: This framework can be implemented standalone and independent from the other frameworks. The existence of a Confidentiality framework has direct implication to the calling client, as the communication with the service must meet the established confidentiality requirements.

In the context of OGC Web Services, this framework is relevant but can only be implemented with limitations, as outlined in the next sub-section.

Integrity Framework: This framework can be implemented standalone and independent from the other frameworks. The existence of an Integrity framework has direct impact to the calling client, as the communication with the service must meet the established integrity requirements.

In the context of OGC Web Services, this framework is relevant but can only be implemented with limitations, as outlined in the next sub-section.

Security Audits and Alarms Framework: This framework can be implemented standalone and should always be implemented, as it guarantees the proper functioning of the security system and trigger administrative actions in case alarms are issued.

Linking the implementation of this framework with OGC Web Services standards is difficult as such an implementation depends on the overall security policy. If the policy mandates a security watch-dog, then this framework must be implemented. The decision to implement is independent from the choice of service type.

Table 1 — Framework implementation implications

	Relevant to implement for OGC Web Services	Implication to the calling client

Authentication Framework	Y	Y
Access Control Framework	Y	(Y)
Non-repudiation Framework	(Y)	N
Confidentiality Framework	Y*	Y
Integrity Framework	Y*	Y
Security Audits and Alarms Framework	N	N

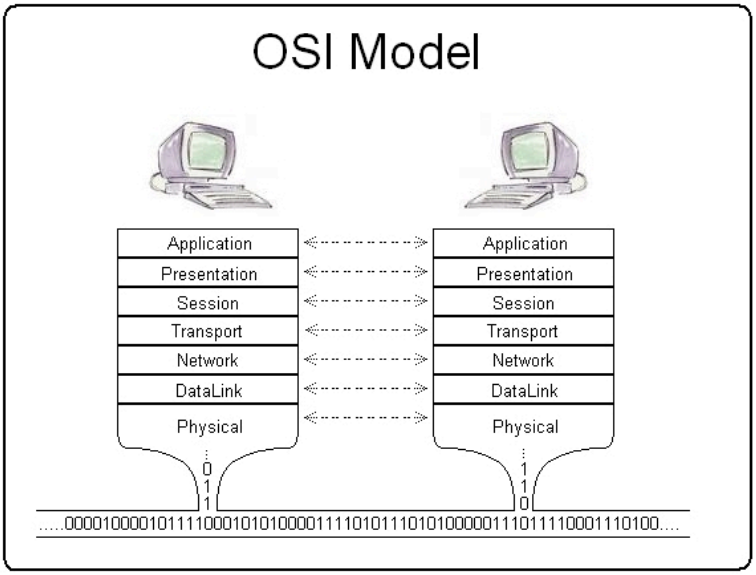
The (Y) indicates that the implementation of this security framework introduces implications to the calling client but that the effect is not necessarily relevant to the programming logic of the client. The implication has effect on the user, as they may be challenged with different responses from the protected service: one request is permitted, the next perhaps denied. In the case where the client shall only issue those requests that will get permitted by the service, the client programming logic must read (and understand) the details of this framework's implementation: The access control rules.

The Y* indicates, that the implementation of this security framework is possible but with limitations as outlined in the next section 4.3.

4.3 Variations of Implementing the Frameworks in the context of OGC Web Services

In general, the communication between distributed systems and applications is based on the conceptual OSI (Open Systems Interconnection) model (see [3] for details). This model is illustrated below.

Figure 1 — OSI Model



[Wikipedia: http://en.wikipedia.org/wiki/File:Osi_model_trad.jpg]

In this model, please note that the "Application" layer (also called layer 7) is still part of OSI model and does not represent the application (client or server program) itself. However, for OGC Web Services implementations, this layer includes the required communication protocol to be leveraged to make a service a **Web** service: HTTP. In that sense the usual case for implementing an OGC Web Service is using Operating System or other framework libraries that abstract communication over HTTP.

Based on the general options of communication supported by HTTP (RFC 2616), two different approaches for encoding execution requests to an OGC Web Services evolved. The different approaches can best be characterized by analyzing the way in which the service operation parameters are submitted with the execution request:

1) Structuring of operation parameters using URI encoding according to RFC 2396 also known as Key-Value-Pair or query string is the simplest way supported. The approach leverages the HTTP GET method to connect to the service endpoint and to submit the service parameters. Even this simple approach is very powerful and can quickly be implemented. This approach has limitations in expressiveness of the parameters. When using this approach with secure service endpoints and according[3] to RFC 2396, it is explicitly recommended to not put any secure / confidential information such as username or password in the query string. One dominant reason is that the URL + query string is typically captured in Web Server log files.

[3] "It is clearly unwise to use a URL that contains a password which is intended to be secret … except in those rare cases where the 'password' parameter is intended to be public."

2) An improvement over KVP is to use XML encoded requests. This relaxes the limitations on expressiveness introduced by KVP. In order to submit XML encoded service parameters, the HTTP method POST is used.

From these general options with HTTP to submit service parameters for OGC Web Services, three main implementation categories evolved: XML, SOAP and KVP.

4.3.1 Category I: HTTP+KVP

This category of implementations of OGC Web Services mainly uses the HTTP protocol plus the GET verb provided by RFC 1616 to communicate with the service endpoint. In addition, the full use of RFC 2396 URIs including the query string is possible. Please note that this category also supports REST and RESTful approaches.

The biggest advantage is at the same time a disadvantage: simplicity. An additional disadvantage for using KVP is the different implementations regarding the maximum length of the URL+query string. This implies a fundamental restriction on the use of HTTP GET + KVP.

Table 2 — HTTP/1.1 citation on URI limitation (RC 2616, 3.2.1)

```
The HTTP protocol does not place any a priori limit on the length of
a URI. Servers MUST be able to handle the URI of any resource they
serve, and SHOULD be able to handle URIs of unbounded length if they
provide GET-based forms that could generate such URIs. A server
SHOULD return 414 (Request-URI Too Long) status if a URI is longer
than the server can handle (see section 10.4.15).

   Note: Servers ought to be cautious about depending on URI lengths
   above 255 bytes, because some older client or proxy
   implementations might not properly support these lengths.
```

Is there a maximum length conclusion for a client? No. From different sources on the Internet, different answers are available about URI length limitations: Internet Explorer seems to limit URIs to 2083 characters. This limitation in particular implies a problem with data URLs: `….`

A typical OGC Web Service request leveraging HTTP GET + KVP looks like this:

```
http://localhost/noa/basic?SERVICE=WFS&VERSION=1.1.0&
REQUEST=GetFeature&TYPENAME=mda%3Anoticeofarrival&
NAMESPACE=xmlns%28mda%3Dhttp%3A%2F%2Frelease.niem.gov%2Fniem%2Fd
omains%2Fmaritime%2F3.0%2Fmda%2F%29&
OUTPUTFORMAT=text%2Fxml%3B subtype%3Dgml%2F3.1.1
```

Looking at the different types of OGC Web Services, it seems that the URI limitation has only impact to specifications like CSW, WCS and WFS that support the use of OGC Filter Encoding. And the actual filter statement could become quite lengthy and exceed the quasi de facto length limit of 2083 characters. As an example, let's consider an example from the OGC Filter Encoding v1.1 specification [3]:

Table 3 — Example 14 from OGC 04-095 (Filter Encoding)

```
<Filter>
    <And>
        <Within>
            <PropertyName>WKB_GEOM</PropertyName>
            <gml:Polygon name="1" srsName="EPSG:63266405">
                <gml:outerBoundaryIs>
                    <gml:LinearRing>
                        <gml:posList>-98.5485,24.2633 ...</gml:posList>
                    </gml:LinearRing>
                </gml:outerBoundaryIs>
            </gml:Polygon>
        </Within>
        <PropertyIsBetween>
            <PropertyName>DEPTH</PropertyName>
            <LowerBoundary><Literal>400</Literal></LowerBoundary>
            <UpperBoundary><Literal>800</Literal></UpperBoundary>
        </PropertyIsBetween>
    </And>
</Filter>
```

And, applying this example Filter to the example WFS GetFeature from above using **only one** typeName has the length of 1016 characters already:

```
http://localhost/noa/basic?SERVICE=WFS&VERSION=1.1.0&
REQUEST=GetFeature&TYPENAME=mda%3Anoticeofarrival&
NAMESPACE=xmlns%28mda%3Dhttp%3A%2F%2Frelease.niem.gov%2Fniem%2Fdomains%2Fmaritime%2F3.0%2
Fmda%2F%29&  OUTPUTFORMAT=text%2Fxml%3B subtype%3Dgml%2F3.1.1&

FILTER=%3CFilter%3E%20%3CAnd%3E%20%3CWithin%3E%20%3CPropertyName%3EWKB_GEOM%3C%2FProperty
Name%3E%20%3Cgml%3APolygon%20name%3D%221%22%20srsName%3D%22EPSG%3A63266405%22%3E%20%3Cgml
%3AouterBoundaryIs%3E%20%3Cgml%3ALinearRing%3E%20%3Cgml%3AposList%3E-
98.5485%2C24.2633%20...%3C%2Fgml%3AposList%3E%20%3C%2Fgml%3ALinearRing%3E%20%3C%2Fgml%3Ao
uterBoundaryIs%3E%20%3C%2Fgml%3APolygon%3E%20%3C%2FWithin%3E%20%3CPropertyIsBetween%3E%20
26%20Copyright%20%C2%A9%20Open%20Geospatial%20Consortium%2C%20Inc%20(2005)%20%3CPropertyN
ame%3EDEPTH%3C%2FPropertyName%3E%20%3CLowerBoundary%3E%3CLiteral%3E400%3C%2FLiteral%3E%3C
%2FLowerBoundary%3E%20%3CUpperBoundary%3E%3CLiteral%3E800%3C%2FLiteral%3E%3C%2FUpperBound
ary%3E%20%3C%2FPropertyIsBetween%3E%20%3C%2FAnd%3E%20%3C%2FFilter%3E
```

When completing the geometry with the simple state boundary for California (110 point locations), the request URI becomes 7718 characters in length (read part over the 2083 character limit):

```
http://localhost/noa/basic?SERVICE=WFS&VERSION=1.1.0&
REQUEST=GetFeature&TYPENAME=mda%3Anoticeofarrival&
NAMESPACE=xmlns%28mda%3Dhttp%3A%2F%2Frelease.niem.gov%2Fniem%2Fdomains%2Fmaritime%2F3.0%2
Fmda%2F%29&  OUTPUTFORMAT=text%2Fxml%3B subtype%3Dgml%2F3.1.1&

FILTER=%3CFilter%3E%20%3CAnd%3E%20%3CWithin%3E%20%3CPropertyName%3EWKB_GEOM%3C%2FProperty
Name%3E%20%3Cgml%3APolygon%20name%3D%221%22%20srsName%3D%22EPSG%3A4326%22%3E%20%3Cgml%3Ao
uterBoundaryIs%3E%20%3Cgml%3ALinearRing%3E%20%3Cgml%3Apos%3E%2041.9983%20-
124.4009%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2042.0024%20-
123.6237%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2042.0126%20-
123.1526%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2042.0075%20-
122.0073%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.9962%20-
121.2369%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.9983%20-
119.9982%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2039.0021%20-
120.0037%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2037.5555%20-
117.9575%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2036.3594%20-
116.3699%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2035.0075%20-
114.6368%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.9659%20-
114.6382%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.9107%20-
114.6286%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.8758%20-
114.6382%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.8454%20-
114.5970%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.7890%20-
114.5682%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.7269%20-
114.4968%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.6648%20-
114.4501%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.6581%20-
114.4597%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.5869%20-
114.4322%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.5235%20-
114.3787%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.4601%20-
114.3869%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.4500%20-
114.3361%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.4375%20-
114.3031%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.4024%20-
114.2674%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.3559%20-
114.1864%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.3049%20-
114.1383%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.2561%20-
114.1315%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.2595%20-
114.1651%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.2044%20-
114.2249%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1914%20-
```

```
114.2221%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1720%20-
114.2908%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1368%20-
114.3237%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1186%20-
114.3622%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1118%20-
114.4089%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.0856%20-
114.4363%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.0276%20-
114.4336%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.0117%20-
114.4652%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.9582%20-
114.5119%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.9308%20-
114.5366%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.9058%20-
114.5091%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.8613%20-
114.5256%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.8248%20-
114.5215%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.7597%20-
114.5050%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.7083%20-
114.4940%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.6832%20-
114.5284%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.6363%20-
114.5242%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.5895%20-
114.5393%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.5528%20-
114.5242%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.5311%20-
114.5586%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.5070%20-
114.5778%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.4418%20-
114.6245%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.4142%20-
114.6506%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.4039%20-
114.7055%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.3546%20-
114.6973%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.3041%20-
114.7302%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.2858%20-
114.7206%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.2754%20-
114.6808%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.2582%20-
114.6698%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.2467%20-
114.6904%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.1720%20-
114.6794%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0904%20-
114.7083%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0858%20-
114.6918%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0328%20-
114.6629%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0501%20-
114.6451%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0305%20-
114.6286%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0282%20-
114.5888%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0351%20-
114.5750%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0328%20-
114.5174%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.9718%20-
114.4913%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.9764%20-
114.4775%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.9372%20-
114.4844%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.8427%20-
114.4679%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.8161%20-
114.5091%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7850%20-
114.5311%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7573%20-
114.5284%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7503%20-
114.5641%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7353%20-
114.6162%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7480%20-
114.6986%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7191%20-
114.7220%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.6868%20-
115.1944%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.5121%20-
117.3395%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2032.7838%20-
117.4823%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.0501%20-
117.5977%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.2341%20-
117.6814%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.4578%20-
118.0591%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.5403%20-
118.6290%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.7928%20-
118.7073%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2033.9582%20-
119.3706%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.1925%20-
120.0050%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.2561%20-
120.7164%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.5360%20-
120.9128%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2034.9749%20-
120.8427%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2035.2131%20-
121.1325%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2035.5255%20-
121.3220%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2035.9691%20-
121.8013%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2036.2808%20-
122.1446%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2036.7268%20-
122.1721%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2037.2227%20-
122.6871%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2037.7783%20-
122.8903%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2037.8965%20-
```

```
123.2378%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2038.3449%20-
123.3202%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2038.7423%20-
123.8338%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2038.9946%20-
123.9793%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2039.3088%20-
124.0329%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2039.7642%20-
124.0823%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2040.1663%20-
124.5314%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2040.4658%20-
124.6509%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.0110%20-
124.3144%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.2386%20-
124.3419%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.7170%20-
124.4545%20%3C%2Fgml%3Apos%3E%20%3Cgml%3Apos%3E%2041.9983%20-
124.4009%20%3C%2Fgml%3Apos%3E%20%3C%2Fgml%3ALinearRing%3E%20%3C%2Fgml%3AouterBoundaryIs%3
E%20%3C%2Fgml%3APolygon%3E%20%3C%2FWithin%3E%20%3CPropertyIsBetween%3E%2026%20Copyright%2
0%C2%A9%20Open%20Geospatial%20Consortium%2C%20Inc%20(2005)%20%3CPropertyName%3EDEPTH%3C%2
FPropertyName%3E%20%3CLowerBoundary%3E%3CLiteral%3E400%3C%2FLiteral%3E%3C%2FLowerBoundary
%3E%20%3CUpperBoundary%3E%3CLiteral%3E800%3C%2FLiteral%3E%3C%2FUpperBoundary%3E%20%3C%2FP
ropertyIsBetween%3E%20%3C%2FAnd%3E%20%3C%2FFilter%3E%20
```

Category II: HTTP+XML

This category leverages the HTTP protocol for the communication basis and submits the service operation parameters, including any Filter, encoded in XML via HTTP POST and mime type text/xml.

The obvious example is that this approach overcomes the simplicity constraint from KVP and the size limitation of the URI. However, there usually is also a size limit for the content-length when POSTing information to a Web Server. But that limit typically is 2MB or more and can be configured in the Web Server setup.

Also, from the Web Server log file perspective the use of POSTed XML requests has the advantage compared to the GET KVP approach that the XML request is not recorded by default. But of course, the logging of XML encoded requests is also possible.

The big disadvantage of using XML requests is the issue of carrying attacks and malicious code as well as replay requests. Therefore, XML validation should be enabled and XML content inspection.

The GetFeature request from above using HTTP GET+KVP looks like the following, using HTTP POST+XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature
    service="WFS"
    version="1.1.0"
    outputFormat="text/xml; subtype=gml/3.1.1"
xmlns:mda="http://release.niem.gov/niem/domains/maritime/3.0/mda/"
xmlns:wfs="http://www.opengis.net/wfs" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <wfs:Query typeName="mda:noticeofarrival">
        <ogc:Filter>
            <ogc:And>
                <ogc:Within>
                    <ogc:PropertyName>WKB_GEOM</ogc:PropertyName>
                    <gml:Polygon xmlns:gml="http://www.opengis.net/gml" name="California"
srsName="EPSG:4326">
```

```
                    <gml:outerBoundaryIs>
                        <gml:LinearRing>
                            <gml:pos> 41.9983 -124.4009 </gml:pos>
                            <gml:pos> 42.0024 -123.6237 </gml:pos>
                            <gml:pos> 42.0126 -123.1526 </gml:pos>
                            <gml:pos> 42.0075 -122.0073 </gml:pos>
                            <gml:pos> 41.9962 -121.2369 </gml:pos>
                            <gml:pos> 41.9983 -119.9982 </gml:pos>
                            <gml:pos> 39.0021 -120.0037 </gml:pos>
                            <gml:pos> 37.5555 -117.9575 </gml:pos>
                            <gml:pos> 36.3594 -116.3699 </gml:pos>
                            <gml:pos> 35.0075 -114.6368 </gml:pos>
                            <gml:pos> 34.9659 -114.6382 </gml:pos>
                            <gml:pos> 34.9107 -114.6286 </gml:pos>
                            <gml:pos> 34.8758 -114.6382 </gml:pos>
                            <gml:pos> 34.8454 -114.5970 </gml:pos>
                            <gml:pos> 34.7890 -114.5682 </gml:pos>
                            <gml:pos> 34.7269 -114.4968 </gml:pos>
                            <gml:pos> 34.6648 -114.4501 </gml:pos>
                            <gml:pos> 34.6581 -114.4597 </gml:pos>
                            <gml:pos> 34.5869 -114.4322 </gml:pos>
                            <gml:pos> 34.5235 -114.3787 </gml:pos>
                            <gml:pos> 34.4601 -114.3869 </gml:pos>
                            <gml:pos> 34.4500 -114.3361 </gml:pos>
                            <gml:pos> 34.4375 -114.3031 </gml:pos>
                            <gml:pos> 34.4024 -114.2674 </gml:pos>
                            <gml:pos> 34.3559 -114.1864 </gml:pos>
                            <gml:pos> 34.3049 -114.1383 </gml:pos>
                            <gml:pos> 34.2561 -114.1315 </gml:pos>
                            <gml:pos> 34.2595 -114.1651 </gml:pos>
                            <gml:pos> 34.2044 -114.2249 </gml:pos>
                            <gml:pos> 34.1914 -114.2221 </gml:pos>
                            <gml:pos> 34.1720 -114.2908 </gml:pos>
                            <gml:pos> 34.1368 -114.3237 </gml:pos>
                            <gml:pos> 34.1186 -114.3622 </gml:pos>
                            <gml:pos> 34.1118 -114.4089 </gml:pos>
                            <gml:pos> 34.0856 -114.4363 </gml:pos>
                            <gml:pos> 34.0276 -114.4336 </gml:pos>
                            <gml:pos> 34.0117 -114.4652 </gml:pos>
                            <gml:pos> 33.9582 -114.5119 </gml:pos>
                            <gml:pos> 33.9308 -114.5366 </gml:pos>
                            <gml:pos> 33.9058 -114.5091 </gml:pos>
                            <gml:pos> 33.8613 -114.5256 </gml:pos>
                            <gml:pos> 33.8248 -114.5215 </gml:pos>
                            <gml:pos> 33.7597 -114.5050 </gml:pos>
                            <gml:pos> 33.7083 -114.4940 </gml:pos>
                            <gml:pos> 33.6832 -114.5284 </gml:pos>
                            <gml:pos> 33.6363 -114.5242 </gml:pos>
                            <gml:pos> 33.5895 -114.5393 </gml:pos>
                            <gml:pos> 33.5528 -114.5242 </gml:pos>
                            <gml:pos> 33.5311 -114.5586 </gml:pos>
                            <gml:pos> 33.5070 -114.5778 </gml:pos>
                            <gml:pos> 33.4418 -114.6245 </gml:pos>
                            <gml:pos> 33.4142 -114.6506 </gml:pos>
                            <gml:pos> 33.4039 -114.7055 </gml:pos>
                            <gml:pos> 33.3546 -114.6973 </gml:pos>
                            <gml:pos> 33.3041 -114.7302 </gml:pos>
                            <gml:pos> 33.2858 -114.7206 </gml:pos>
                            <gml:pos> 33.2754 -114.6808 </gml:pos>
                            <gml:pos> 33.2582 -114.6698 </gml:pos>
                            <gml:pos> 33.2467 -114.6904 </gml:pos>
                            <gml:pos> 33.1720 -114.6794 </gml:pos>
                            <gml:pos> 33.0904 -114.7083 </gml:pos>
                            <gml:pos> 33.0858 -114.6918 </gml:pos>
                            <gml:pos> 33.0328 -114.6629 </gml:pos>
                            <gml:pos> 33.0501 -114.6451 </gml:pos>
                            <gml:pos> 33.0305 -114.6286 </gml:pos>
                            <gml:pos> 33.0282 -114.5888 </gml:pos>
                            <gml:pos> 33.0351 -114.5750 </gml:pos>
                            <gml:pos> 33.0328 -114.5174 </gml:pos>
```

```
                                <gml:pos> 32.9718 -114.4913 </gml:pos>
                                <gml:pos> 32.9764 -114.4775 </gml:pos>
                                <gml:pos> 32.9372 -114.4844 </gml:pos>
                                <gml:pos> 32.8427 -114.4679 </gml:pos>
                                <gml:pos> 32.8161 -114.5091 </gml:pos>
                                <gml:pos> 32.7850 -114.5311 </gml:pos>
                                <gml:pos> 32.7573 -114.5284 </gml:pos>
                                <gml:pos> 32.7503 -114.5641 </gml:pos>
                                <gml:pos> 32.7353 -114.6162 </gml:pos>
                                <gml:pos> 32.7480 -114.6986 </gml:pos>
                                <gml:pos> 32.7191 -114.7220 </gml:pos>
                                <gml:pos> 32.6868 -115.1944 </gml:pos>
                                <gml:pos> 32.5121 -117.3395 </gml:pos>
                                <gml:pos> 32.7838 -117.4823 </gml:pos>
                                <gml:pos> 33.0501 -117.5977 </gml:pos>
                                <gml:pos> 33.2341 -117.6814 </gml:pos>
                                <gml:pos> 33.4578 -118.0591 </gml:pos>
                                <gml:pos> 33.5403 -118.6290 </gml:pos>
                                <gml:pos> 33.7928 -118.7073 </gml:pos>
                                <gml:pos> 33.9582 -119.3706 </gml:pos>
                                <gml:pos> 34.1925 -120.0050 </gml:pos>
                                <gml:pos> 34.2561 -120.7164 </gml:pos>
                                <gml:pos> 34.5360 -120.9128 </gml:pos>
                                <gml:pos> 34.9749 -120.8427 </gml:pos>
                                <gml:pos> 35.2131 -121.1325 </gml:pos>
                                <gml:pos> 35.5255 -121.3220 </gml:pos>
                                <gml:pos> 35.9691 -121.8013 </gml:pos>
                                <gml:pos> 36.2808 -122.1446 </gml:pos>
                                <gml:pos> 36.7268 -122.1721 </gml:pos>
                                <gml:pos> 37.2227 -122.6871 </gml:pos>
                                <gml:pos> 37.7783 -122.8903 </gml:pos>
                                <gml:pos> 37.8965 -123.2378 </gml:pos>
                                <gml:pos> 38.3449 -123.3202 </gml:pos>
                                <gml:pos> 38.7423 -123.8338 </gml:pos>
                                <gml:pos> 38.9946 -123.9793 </gml:pos>
                                <gml:pos> 39.3088 -124.0329 </gml:pos>
                                <gml:pos> 39.7642 -124.0823 </gml:pos>
                                <gml:pos> 40.1663 -124.5314 </gml:pos>
                                <gml:pos> 40.4658 -124.6509 </gml:pos>
                                <gml:pos> 41.0110 -124.3144 </gml:pos>
                                <gml:pos> 41.2386 -124.3419 </gml:pos>
                                <gml:pos> 41.7170 -124.4545 </gml:pos>
                                <gml:pos> 41.9983 -124.4009 </gml:pos>
                            </gml:LinearRing>
                        </gml:outerBoundaryIs>
                    </gml:Polygon>
                </ogc:Within>
                <ogc:PropertyIsBetween>
                    <ogc:PropertyName>DEPTH</ogc:PropertyName>
                    <ogc:LowerBoundary><Literal>400</Literal></ogc:LowerBoundary>
                    <ogc:UpperBoundary><Literal>800</Literal></ogc:UpperBoundary>
                </ogc:PropertyIsBetween>
            </ogc:And>
        </ogc:Filter>
    </wfs:Query>
</wfs:GetFeature>
```

4.3.2 Category III: HTTP+SOAP

This category leverages the HTTP protocol for the communication basis and submits the service operation parameters in the <Body> part of the XML encoded SOAP message.

The <Header> part becomes available to introduce options for mainly implementing the Integrity and Confidentiality frameworks by leveraging the WS-* stack.

4.4 Applying *Common Security* to the different Implementation Options

4.4.1 Category I: HTTP+KVP

4.4.1.1 Authentication

The HTTP 1.1 protocol (as per RFC 1616) supports the use of HTTP authentication via the HTTP status code 401. Typically, the use of HTTP authentication with the methods BASIC and DIGEST, as defined in RFC 2617, can be used. As the username and password are submitted with the request as clear text within the HTTP header attribute "Authorization" the implementation of HTTP Authentication on HTTP is not recommended. Even though DIGEST improves BASIC by the use of nonce values to prevent reply attacks, HTTP Authentication should only be used with secure HTTP communication, as introduced by RFC 2818 (HTTP over TLS). In this sense, the independent implementation of the Authentication Framework is not recommended.

It is important to note that the approach of using HTTP Authentication is not limited to the use of BASIC and DIGEST.

4.4.1.2 Access Control

The HTTP protocol (as per RFC 1616) supports the implementation of the Access Control framework by the concept of "Deny" through the status code 403. Which rules are applied on the service hosting side to determine that the request results in a 403 rather a status code of 200 must not be specified. This leaves freedom for any service instance to implement access control according to the individual need per service instance and endpoint operation. However, this introduces a burden to advertise as outlined in a later section.

The independent implementation of the Access Control framework is possible as long authorization decisions are based on the HTTP request context. The context typically includes information about the environment, the calling client and the requested resource. In cases where Access Control shall be based on information about the calling client in terms of authentication or identity information, the Authentication framework must also be implemented. Keep in mind that the implementation of the Access Control framework itself does not require to implement secure communication, but when used in combination with Authentication, secure communication is recommended. So in essence, one should always implement access control plus authentication with HTTP over TLS.

4.4.1.3 Integrity / Confidentiality

The HTTP protocol supports encrypted communication of the transport layer by leveraging the RFC 2818. This introduces the protocol scheme http**s** that essentially is HTTP over TLS. As illustrated in figure 1 (OSI Model), TLS applies encryption to the

Transport Layer of the communication stack. That enables to use HTTP unchanged, as it "lives" on a higher layer in the communication stack; in the Application Layer.

As standardized in HTTP 1.1 (RFC 2616), a HTTP URL has the following structure:

"http:" "//" host [":" port] [abs_path ["?" query]] [12], section 3.2.2

Therefore, the actual OGC Web Service requests using HTTP+KVP does look identical regardless if HTTP or HTTPS is being used!

This "secure channel" communication provides confidentiality limited to the channel and therefore provides confidentiality from point-to-point rather than end-to-end. Also, the URL itself is not confidential, as it is not part of the communication channel.

In cases where the Integrity of the Confidentiality framework shall be applied to the HTTP+KVP, the query string or individual keys or key-value-pairs must be encrypted accordingly.

4.4.2 Category II: HTTP+XML

4.4.2.1 Authentication

The implementation of this framework is identical to the one for Category I.

4.4.2.2 Access Control

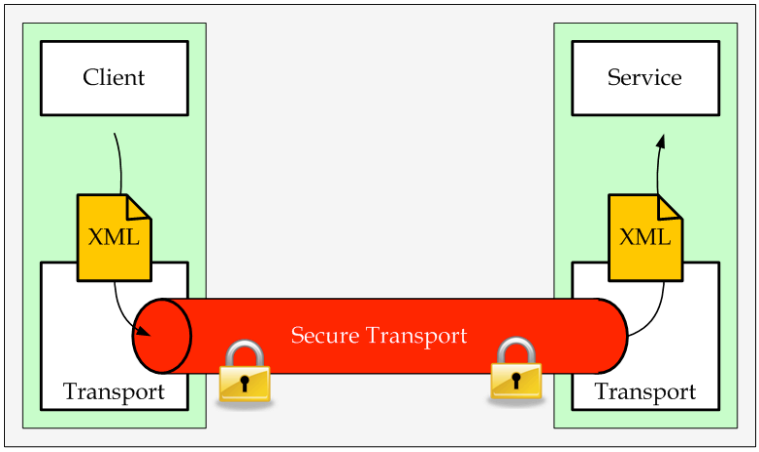
The implementation of this framework is identical to the one for Category I.

4.4.2.3 Integrity / Confidentiality

The implementation of these frameworks can easily be established by leveraging HTTP over TLS, as described for the Category I. The use of XML encoded requests is an improvement over the use of KVP, as the entire request is secretly sent through the secure communication channel. However, limitations in terms of point-to-point apply and must be considered in particular when crafting workflows or communicating over proxies.

The following figure illustrates the basic point-to-point limitation when leveraging secure communication based on TLS. The information item is "secure" as long as it resides in the secure communication channel between the client and service endpoints. Once the information (XML) reaches the receiver (service in the example below), it is available in clear text.

Figure 2 — Transport Layer Security



4.4.3 Category III: HTTP+SOAP

4.4.3.1 Authentication

For the implementation of this framework, two options exist. The first is identical to the one for Category I by leveraging the HTTP protocol options. The second is based on the implementation of the Integrity Framework. When using WS-Security to apply Digital Signatures to ensure integrity of SOAP encoded requests, it is not required that the key can be associated with an identity. For implementing the Authentication framework on top of the Integrity framework, a private key would be used to apply a digital signature, and the key is associated with the senders (service calling client's) identity.

4.4.3.2 Access Control

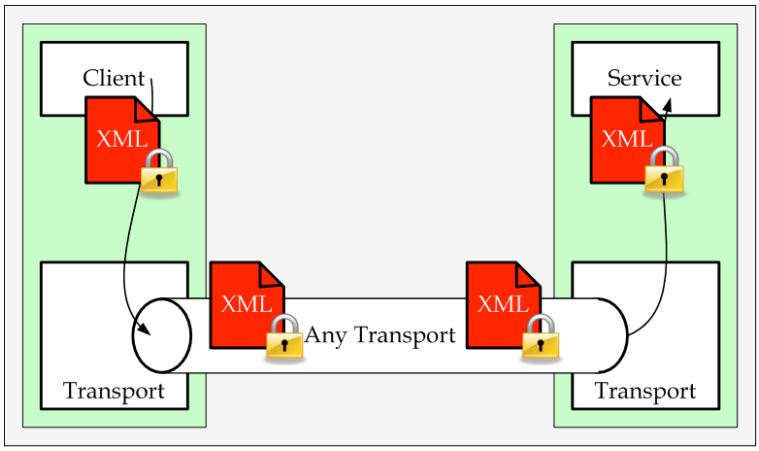
The implementation of this framework is identical to the one for Category I.

4.4.3.3 Integrity / Confidentiality

The implementation of these frameworks can (and should) take place independent from the HTTP options using TLS. The strength of SOAP encoded requests over simple XML encoded requests is that WS-Security (and other applicable standards of that stack) can be used to individually ensure integrity and/or confidentiality of the request as a whole or in parts. The use of W3C's XML Digital Signatures and XML Encryption provide a large flexibility with the risk of lack of interoperability.

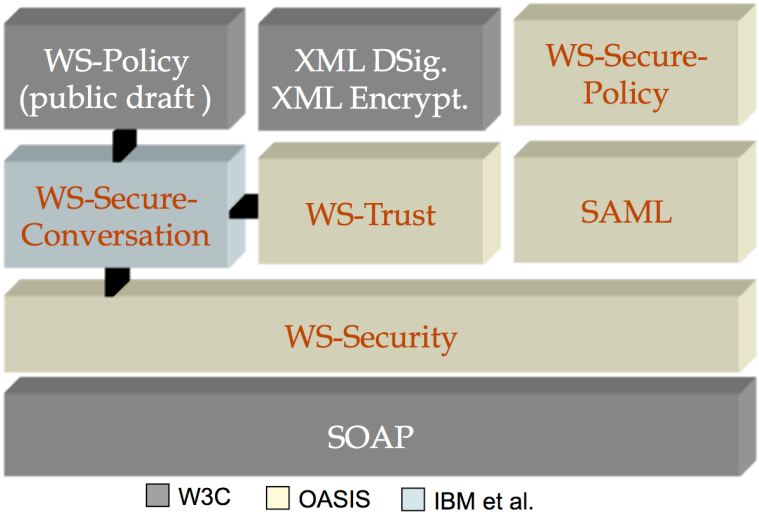
Compared to HTTP over TLS, the encrypted XML stays with integrity and confidentiality, even when it has reached the receiver (service in the example below):

Figure 3 — Message Level Security



However, when using WS-Security, a strong need exists to advertise the constraints regarding integrity and confidentiality to the calling client. This can be undertaken by using additional standards of the WS-* family like WS-Policy, WS-SecurityPolicy and WS-PolicyAttachment. The following figure illustrates major building blocks of the WS-* family.

Figure 4 — WS-* Family (major building blocks)



A complete example for implementing WS-Security can be obtained here: http://java.globinch.com/enterprise-java/web-services/jax-ws/secure-metro-jax-ws-usernametoken-web-service-signature-encryption/

4.5 Implications from the different implementation variations

An OGC Web Service endpoint that is capable of supporting HTTP+KVP and HTTP+XML or even HTTP+SOAP introduces a noticeable challenge to security,

because the state transfer from HTTP+KVP request to HTTP+XML to HTTP+KVP must be maintained and the implementation of the different frameworks must ensure identical assurance. This seems to be difficult considering the fact that XML encoded requests are much more expressive then KVP encoded requests. In that sense, a secured implementation of an OGC Web Service should advertise the appropriate HTTP method: **HTTP Get** or **Post** and if applicable both. However for the client side, it is recommended to only use one HTTP method (GET or POST) once a security session is initiated. This relaxes the conditions to implement a security framework.

The use of HTTP+SOAP introduces the option to leverage message security a là WS-Security which introduces multiple options with regard to which parts of the request XML must be encrypted, which must be digitally signed, which algorithm to use, which key length to use, etc. etc. This seems to immediately break the hardly achieved interoperability with OGC Web Service standards that describe how to call a service for a particular purpose. From that perspective, if you wish to implement OGC Web Services using the Integrity and Encryption frameworks and want simple interoperability, then you should **not use SOAP and WS-Security** unless all services run inside a controlled security domain and you have full control over the "what" gets encrypted and how. In addition, a detailed service description using WSDL and WS-Policy must be available to instrument proper implementation on the client. For a complete example on all the parts that must play together, please use the link below the figure 4 (above).

With the new hype of mobile clients to be used as geospatial platforms, the use of SOAP and even XML seems to raise additional concerns that should be explored further in future research.

4.6 Need for Interoperable Description of the *Common Security* Constraints

In the context of OGC Web Services it is important that the calling client understands the constraints of a service instance implied by the implementation of the introduced security frameworks. As summarized in table 1, it should be possible to describe the existence of those security framework implementations that have an impact on the client. In particular, the ability to describe the existence of a single framework being implemented or any (meaningful) combination must be possible.

From table 1, the client needs to "know" about the existence of the authentication, access control and integrity / confidentiality frameworks. The existence of the non-repudiation and the audit and alarms frameworks is not of concern to be reported to the client.

In addition to the general ability to define the existence of a security framework, it must certainly be possible to describe the existence of security framework(s) per service instance. For OGC Web Services, you must be able to describe security implications per service operation. Examples of unrestricted (unprotected) operations may include GetCapabilities, GetLeganedGraphics, DescribeFeatureType. Examples of service operations that are target for getting protected are GetMap, GetFeature, Transaction, etc.

5 Determine the best place for advertising *Common Security* constraints

In order to determine the best place for advertising security constrains, reflecting the security in place at a service instance, understanding the general IT Publish-Find-Bind paradigm and in particular the OGC implementation of that paradigm is required.

5.1 Publish-Find-Bind using Universal Description Discovery and Integration (UDDI)

Universal Description Discovery and Integration (UDDI) is a model that uses XML based registries where a service provider can publish (UDDI) compliant descriptions of a web offering, e.g. a Web Service. The description usually comprises a set of documents, each crafted for a different purpose. In order to support best the find and bind, UDDI provides Yellow, White and Green pages. Of particular interest are the Green Pages, which include the technically relevant documents to implement the interface of the service.

Without going into the full detail of the UDDI process, please note in the context of this report that the final outcome of the Find process is a **WSDL** document that describes the offering properly to build a client that is capable to execute the service in the Bind phase.

Security constraints of the offering that effect the Bind is therefore (obviously) best included into the WSDL document. This can take place by using XLinks to external descriptions using WS-Policy.

The use of WS-Policy closes the loop with the original concept to use SOAP for Web Service communication; so the use of WSDL, SOAP encoded service requests, integrity and confidentiality implemented using WS-Security and WS-Policy to describe the SOAP message requirements is the natural choice.

The following examples illustrate an example WS-Policy and a WSDL that uses the WS-Policy:

Table 4 — WS-Policy example[4]

```
<wsp:Policy wsu:Id=" TutorialWebServiceSOAP ">
    <wsp:ExactlyOne>
        <wsp:All>
            <sp:AsymmetricBinding>
                <wsp:Policy>
                    <sp:InitiatorToken>
                        <wsp:Policy>
                            <sp:X509Token
sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/Alway
sToRecipient">
                                <wsp:Policy>
                                    <sp:WssX509V3Token11 />
                                </wsp:Policy>
                            </sp:X509Token>
                        </wsp:Policy>
```

[4] http://concentricsky.com/blog/2012/dec/implementing-ws-security-cxf-wsdl-first-web-service

```
                    </sp:InitiatorToken>
                    <sp:RecipientToken>
                        <wsp:Policy>
                            <sp:X509Token
sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/Never
">
                                <wsp:Policy>
                                    <sp:WssX509V3Token11 />
                                    <sp:RequireIssuerSerialReference />
                                </wsp:Policy>
                            </sp:X509Token>
                        </wsp:Policy>
                    </sp:RecipientToken>
                    <sp:Layout>
                        <wsp:Policy>
                            <sp:Strict />
                        </wsp:Policy>
                    </sp:Layout>
                    <sp:IncludeTimestamp />
                    <sp:OnlySignEntireHeadersAndBody />
                    <sp:AlgorithmSuite>
                        <wsp:Policy>
                            <sp:Basic128 />
                        </wsp:Policy>
                    </sp:AlgorithmSuite>
                    <sp:EncryptSignature />
                </wsp:Policy>
            </sp:AsymmetricBinding>
            <sp:Wss11>
                <wsp:Policy>
                    <sp:MustSupportRefIssuerSerial />
                </wsp:Policy>
            </sp:Wss11>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

Table 5 — WSDL with WS-Policy reference example[5]

```
<wsdl:binding name="TutorialWebServiceSOAP" type="tns:TutorialWebService">
    <wsp:PolicyReference URI="#TutorialBindingPolicy" />
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="sendTutorialMessage">
        <soap:operation soapAction="http://example.com/tutotial/sendTutorialMessage" />
        <wsdl:input>
            <wsp:PolicyReference URI="#TutorialInputBindingPolicy"/>
            <soap:body use="literal" parts="parameters" />
            <soap:header use="literal" part="source" message="tns:TutorialRequest"/>
        </wsdl:input>
        <wsdl:output>
            <wsp:PolicyReference URI="#TutorialOutputBindingPolicy"/>
            <soap:body use="literal" parts="response"/>
            <soap:header use="literal" part="acknowledgment"
message="tns:TutorialResponse"/>
        </wsdl:output>
        <soap:address location="http://localhost/" />
        </wsdl:port>
        </wsdl:service>
    </wsdl:operation>
</wsdl:binding>
```

[5] http://concentricsky.com/blog/2012/dec/implementing-ws-security-cxf-wsdl-first-web-service

5.2 Publish-Find-Bind in the OGC World

When studying the concept of OGC Web Services and the approach to Publish-Find-Bind, one will find that the established procedure is different than UDDI.

Because the goal of this ER is to find options to describe Common Security across all OGC Web Service standards and make recommendations on the best option(s) is(are), studying the OGC way of Publish-Find-Bind in more detail is essential.

5.2.1 OGC Publish

For OGC Web Service instances, typically two types of descriptions are uploaded to a registry:

1) An ISO-19115 compliant metadata document that focuses on describing the dataset

2) An ISO compliant metadata document that focuses on describing the actual service instance with a link to the ISO metadata for served the data set

From the perspective of *Common Security*, the description of the data set is not of any concern, as it has no implications to the actual Bind process. However, license and use restrictions on the data set might be in place and must be honored when receiving the data via the providing service.

The interesting bit from the perspective of *Common Security* is the ISO metadata description for the service instance. The options available to include or link to security constraint description of implemented frameworks are addressed in more detail in a later section of this ER.

5.2.2 OGC Find

The find process usually involves querying a CSW, which is able to search one or multiple registries of ISO-19115 compliant metadata records. The result of a find process is one or many ISO metadata file(s). Each file either describes characteristics of a data set or a service instance.

Notice that the result of the OGC Find process is **not** a WSDL document that enables a developer to implement a client. Instead, the ISO metadata file contains the service base URL that enables the client to execute the GetCapabilities operation. The result of the GetCapabilities operation (the Capabilities document) enables the client to fully bind with the service.

In case that *Common Security* is in place for the service, these constraints can be advertised in both documents: (i) the ISO metadata and (ii) the OGC Capabilities document. However, taking under consideration that the metadata file is used of human consumption, it is not necessary to encode the constraints introduced by the Common Security to be machine-readable. However, this can't hurt as using machine-readable encoding enables to properly visualize the service characteristics to the user, in particular the existing security constraints. Because the client always uses the service base URL and the GetCapabilities operation to bind with the service instance, the ability to describe Common Security constraints inside the Capabilities document in a machine-readable fashion seems to be mandatory.

5.2.3 OGC Bind

In the OGC world, the actual Bind to an OGC Web Services does not take place using a WSDL as for the UDDI approach. Instead, a service instance specific Capabilities file is used. Beside other pieces of information, the Capabilities file outlines the options to execute service operations. The Capabilities file provides insight into the accepted options for packaging a service request (HTTP+KVP, HTTP+POST, HTTP+SOAP) and outlines the possible data types returned. This information is provided per service operation, similar to WSDL.

As the capabilities document is encoded as an XML instance document containing all the information for a calling client to bind to the service, this seems to be the natural place to put the security constraints as required to describe the implemented frameworks regarding *Common Security*. There is not an option for all different versions of the Capabilities document (as we will see in a later section) to describe or hook security constraints descriptions in the Capabilities document.

6 Exploring existing options in ISO Metadata based on an Example

Note: The intention of this chapter is not to criticize but to make clear the difficulties of using the currently available options!

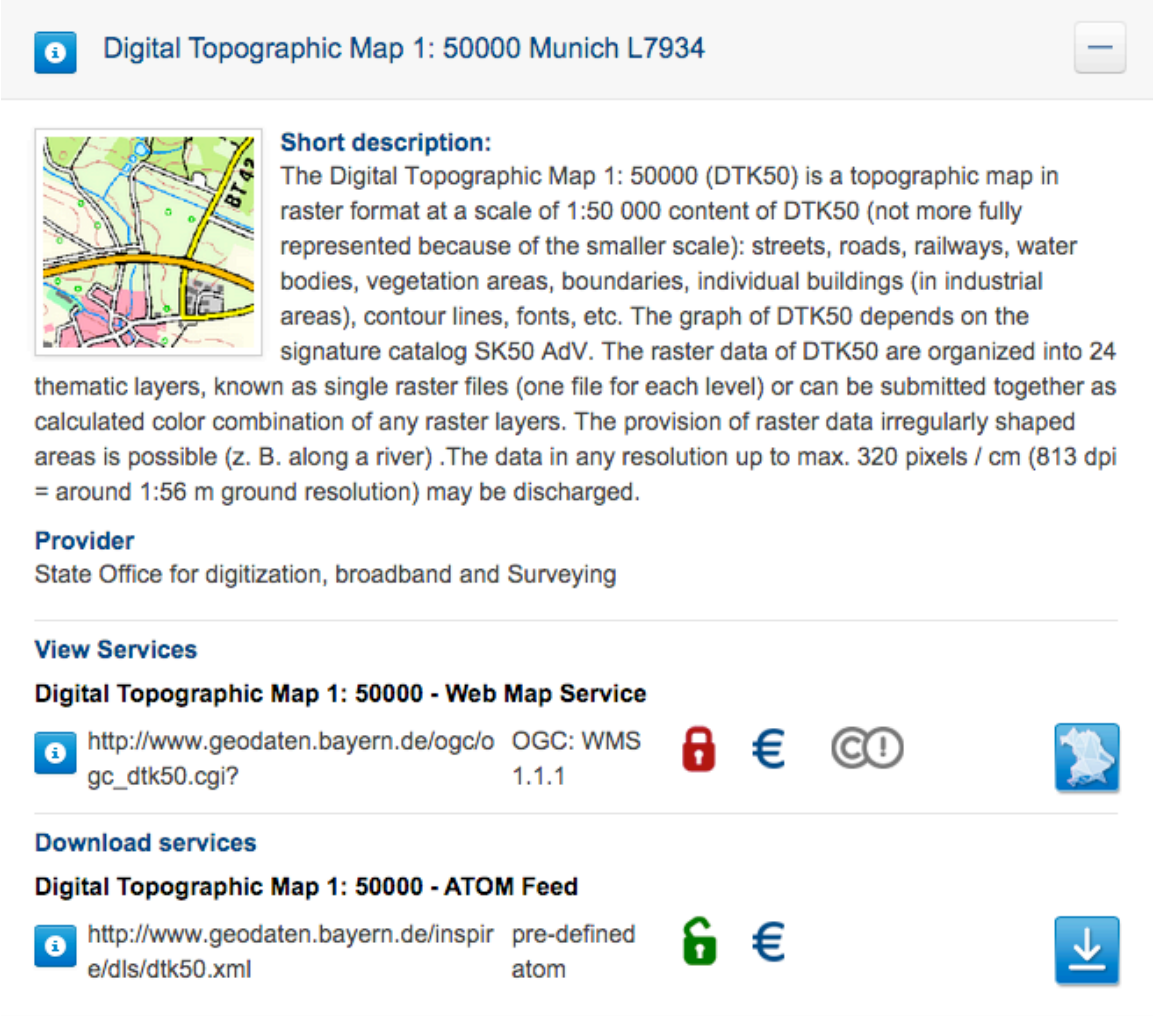
This chapter illustrates an example approach that leverages the ISO 19115 metadata options for outlining constraints. In order to understand the limitations of existing options, and to better understand the need / requirements to improve the options and the process, let's study an INSPIRE[6] offering from the Bavarian SDI.

[6] http://inspire.ec.europa.eu/

As the Publish element of the web services model is not relevant for this ER, let's start with the Find step. The search can be conducted from the Bavarian SDI portal[7]: http://geoportal.bayern.de/geoportalbayern/

For this example, we are looking for the topographic map of Munich in the scale 1:50.000. Typing "DTK50 München" into the search box can fetch the results. The following figure is a screenshot, displaying the response.

Figure 5 — GDI Bavaria example to illustrate WMS restrictions



As we can see, the portal page illustrates various characteristics for two options how to obtain a topographic map for Munich in the scale of 1:50.000:

[7] Please note that the portal is only available in German and that all screenshots showing English language result from Google automatic translation.

1) The first response is an OGC WMS of version 1.1.1. From the symbols used, you can clearly determine that access control is in place (red lock). You can also determine that there is a fee involved to use it (EUR symbol) and that there are use restrictions (Creative Commons symbol).

2) The second response is an ATOM Feed (INSPIRE Download Service). From the symbols used, you can clearly determine that it is open but that a fee is involved.

Where do the triggers for displaying the symbols come from? The editor expects that the visualization for the View Service (first hit) is based on the ISO metadata document for that service instance that can be obtained from this URL[8]:
http://geoportal.bayern.de/geoportalbayern/detailpage?10-1.ILinkListener-detailOverview-downloadLink&resId=9690a6e4-8903-4ff5-b739-355b4b03d72f

Taking a closer look at the metadata, we find the following XML snippet that is responsible for the different symbols:

```
<gmd:resourceConstraints>
    <gmd:MD_LegalConstraints>
        <gmd:accessConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/re
sources/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="copyright">copyright</gmd:MD_RestrictionCode>
        </gmd:accessConstraints>
    </gmd:MD_LegalConstraints>
</gmd:resourceConstraints>
<gmd:resourceConstraints>
    <gmd:MD_LegalConstraints>
        <gmd:useLimitation>
            <gco:CharacterString>Nutzungsbedingungen: Für den Zugang zum
kostenpflichtigen Dienst benötigen Sie eine Kennung und ein Passwort. Diese
Zugangsdaten erhalten Sie über den Kundenservice. Informationen zu den Preisen des
Dienstes entnehmen Sie der Preisliste unter
https://geoportal.bayern.de/geodatenonline/seiten/preise. Es gelten die
Nutzungsbedingungen der Bayerischen Vermessungsverwaltung
(https://geoportal.bayern.de/geodatenonline/seiten/nutzungsbedingungen).</gco:Character
String>
        </gmd:useLimitation>
        <gmd:useConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/re
sources/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="license">license</gmd:MD_RestrictionCode>
        </gmd:useConstraints>
        <gmd:useConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/re
sources/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="otherRestrictions">otherRestrictions</gmd:MD_RestrictionCode>
        </gmd:useConstraints>
        <gmd:otherConstraints>
            <gco:CharacterString>Nutzungsbedingungen: Für den Zugang zum
kostenpflichtigen Dienst benötigen Sie eine Kennung und ein Passwort. Diese
Zugangsdaten erhalten Sie über den Kundenservice. Informationen zu den Preisen des
Dienstes entnehmen Sie der Preisliste unter
https://geoportal.bayern.de/geodatenonline/seiten/preise. Es gelten die
```

[8] The full ISO metadata is available in the Annex

```
Nutzungsbedingungen der Bayerischen Vermessungsverwaltung
(https://geoportal.bayern.de/geodatenonline/seiten/nutzungsbedingungen).</gco:Character
String>
        </gmd:otherConstraints>
    </gmd:MD_LegalConstraints>
</gmd:resourceConstraints>
```

6.1 The Red Lock Symbol

One could simply conclude that the red lock symbol indicates Access Control, which is quite right even though the mouse over help reads “with authentication”. From the ISO metadata document, the following snippet must have triggered that symbol:

```
<gmd:MD_RestrictionCode codeList=".../ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="otherRestrictions">otherRestrictions</gmd:MD_RestrictionCode>
```

This construct helps to visualize a “red lock” but as there is no semantics to explain what a user must do or a developer must implement to overcome the security constraint, this XML element has not enough information. One could link the next XML element even though the name is very general “otherConstraints” with the “otherRestriction”:

```
<gmd:otherConstraints>
            <gco:CharacterString>Nutzungsbedingungen: Für den Zugang zum
kostenpflichtigen Dienst benötigen Sie eine Kennung und ein Passwort. Diese Zugangsdaten
erhalten Sie über den Kundenservice. Informationen zu den Preisen des Dienstes entnehmen
Sie der Preisliste unter https://geoportal.bayern.de/geodatenonline/seiten/preise. Es
gelten die Nutzungsbedingungen der Bayerischen Vermessungsverwaltung
(https://geoportal.bayern.de/geodatenonline/seiten/nutzungsbedingungen).</gco:CharacterS
tring>
        </gmd:otherConstraints>
```

The obvious limitation is that in an international ISO metadata document, you find German text explaining that you need an account with them, which will require a fee according to their terms of use and fees. A non-German speaker would have difficulty understanding what the issue is. Therefore, this approach is incomplete and lacks in interoperability.

6.2 The € Symbol

This symbol clearly implies that you need to pay money to use the service. Or is the fee for obtaining the actual map? From the ISO metadata document, the following snippet might have triggered that symbol:

```
<gmd:distributionOrderProcess>
    <gmd:MD_StandardOrderProcess>
        <gmd:fees>
            <gco:CharacterString>geldleistungspflichtig</gco:CharacterString>
        </gmd:fees>
```



```
    </gmd:MD_StandardOrderProcess>
</gmd:distributionOrderProcess>
```

Or, maybe this XML snippet:

```
<gmd:useConstraints>
    <gmd:MD_RestrictionCode codeList=".../ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="license">license</gmd:MD_RestrictionCode>
</gmd:useConstraints>
```

Similar to the description of "otherRestrictions", there is no fine-grained semantics with this element. But assuming that the previous XML element in the metadata is linked even tough the element name reads "useLimitation" and not "fee", we get some semantics:

```
<gmd:useLimitation>
    <gco:CharacterString>Nutzungsbedingungen: Für den Zugang zum kostenpflichtigen
Dienst benötigen Sie eine Kennung und ein Passwort. Diese Zugangsdaten erhalten Sie über
den Kundenservice. Informationen zu den Preisen des Dienstes entnehmen Sie der
Preisliste unter https://geoportal.bayern.de/geodatenonline/seiten/preise. Es gelten die
Nutzungsbedingungen der Bayerischen Vermessungsverwaltung
(https://geoportal.bayern.de/geodatenonline/seiten/nutzungsbedingungen).</gco:CharacterS
tring>
</gmd:useLimitation>
```

The provided text (response) is written in German (again) and does not really explain to the user what the fee is. As for the "Red Lock", this approach seem incomplete with a lack of interoperability.

6.3 The CC Symbol

As for the previous symbols, one can easily find a trigger in the ISO metadata:

```
<gmd:useConstraints>
    <gmd:MD_RestrictionCode codeList=".../ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="license">license</gmd:MD_RestrictionCode>
</gmd:useConstraints>
```

However, determining which CC license this service has in place is not easy. Or is the data set served with a particular CC license? From the ISO metadata, we find no additional information and in particular not a link to a CC license as we would have expected.

If the above assumption is correct, how can we link the following XML snippet to a meaningful security constraint?

```
<gmd:MD_LegalConstraints>
    <gmd:accessConstraints>
        <gmd:MD_RestrictionCode codeList=".../ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="copyright">copyright</gmd:MD_RestrictionCode>
```

```
    </gmd:accessConstraints>
</gmd:MD_LegalConstraints>
```

The XML element tag reads “accessConstraints” but the ISO restriction code points to “copyright”, which might not be the expected content for everyone.

6.4 Study of the Security Indicators for the Download Service

The second response in the screenshot above shows a green and open lock, which indicates open and unconstrained access. There also is a fee symbol. How does that fit together: An open service with fee constraints?

In order to understand the mechanics behind, let’s take a closer look at the ISO metadata document that can be obtained via this URL: http://geoportal.bayern.de/geoportalbayern/detailpage?11-1.ILinkListener-detailOverview-downloadLink&resId=b6fca3d5-21b1-33c5-ab45-c67708751076

From that metadata document, the only XML snippet that could have triggered the € symbol is the following:

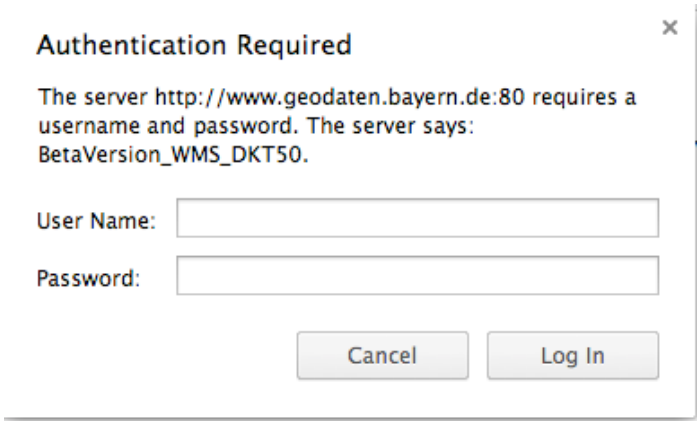
```
<gmd:distributionOrderProcess>
    <gmd:MD_StandardOrderProcess>
        <gmd:fees>
            <gco:CharacterString>geldleistungspflichtig</gco:CharacterString>
        </gmd:fees>
        <gmd:orderingInstructions>
            <gco:CharacterString>Der Downloaddienst auf die Digitale Topographische Karte
1:50000 steht auf Anfrage zur Verfügung. Zur Nutzung des Dienstes benötigen Sie eine
Kennung und ein Passwort. Wenden Sie sich hierfür bitte an unseren
Kundenservice.</gco:CharacterString>
        </gmd:orderingInstructions>
    </gmd:MD_StandardOrderProcess>
</gmd:distributionOrderProcess>
```

The existence of the `<gmd:fees>` element is a clear indication. Please note that there are no further “Constraint” elements in the metadata document. In particular, there is no access, legal or use constraint. However, the other element right after the `<gmd:fees>` element contains important information, unfortunately in German only! The text clearly states that this Download Service will provide links to obtain a topographic map with scale 1:50000 via a Viewing Service. And for that access you do need an account. Further information can be obtained from customer service but no contact details are provided!

When following the Download Service using the provided link (http://www.geodaten.bayern.de/inspire/dls/dtk50.xml) and a specific link for the Download Service for Munich (http://www.geodaten.bayern.de/inspire/dls/dtk50.DEBY_89fd9293-9c78-39f5-a34d-914ff4443d22.xml), a list of View Services becomes available. Selecting the View

Service for obtaining the topographic map of Munich[9], the service returns a HTTP status code 401. This indicates that a username and password is required with the service provider, as we can derive from the login box.

Figure 6 — HTTP Basic Login



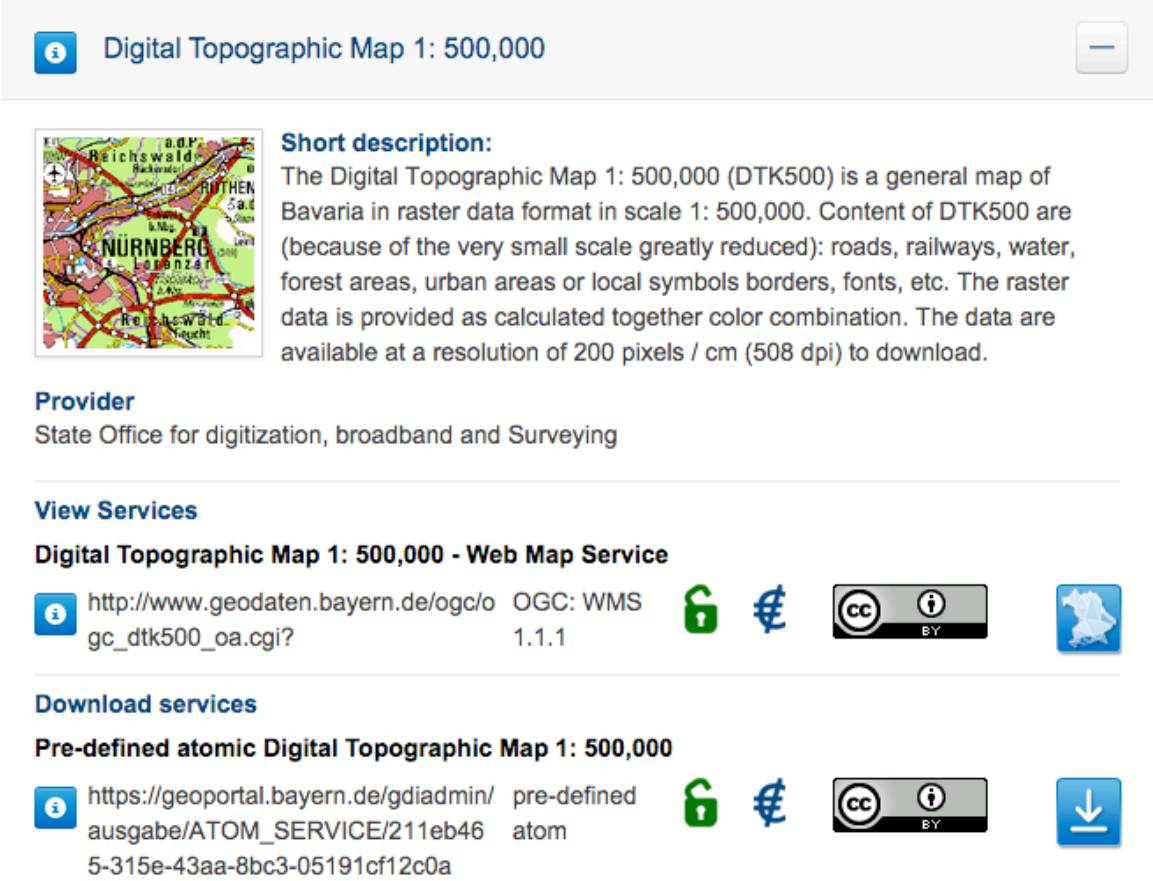
This example seems to be incomplete, as at the end the same restrictions apply as per information for the service itself. However, these restrictions were not displayed in the beginning.

6.5 Study of the ISO Metadata for an open View Service

The following screenshot illustrates that the View Service is a WMS of version 1.1.1 that has no security constraints:

9 http://www.geodaten.bayern.de/ogc/ogc_dtk50.cgi?service=WMS&version=1.1.1&request=GetMap&width=1997&height=1791&srs=EPSG:31468&bbox=4470230.554,5335856.672,4475224.21,5340336.09&layers=by_dtk50&styles=&format=image/tiff&

Figure 7 — GDI Bavaria example to illustrate Atom Feed restrictions



The green lock indicates no access control and no authentication; the negative EUR symbol indicates that no fees are involved. However, there are supposed to be license restrictions as indicated by the Creative Commons symbol. Taking a closer look at the underlying ISO metadata (http://geoportal.bayern.de/geoportalbayern/detailpage?4-1.ILinkListener-detailOverview-downloadLink&resId=29b23ebc-16ff-46d9-88c8-c5141db30547) unveils that there are "Constraint" elements, which we did not expect to find due to the green lock:

```
<gmd:resourceConstraints>
    <gmd:MD_LegalConstraints>
        <gmd:accessConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/resou
rces/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="copyright">copyright</gmd:MD_RestrictionCode>
        </gmd:accessConstraints>
    </gmd:MD_LegalConstraints>
</gmd:resourceConstraints>
<gmd:resourceConstraints>
    <gmd:MD_LegalConstraints>
        <gmd:useLimitation>
            <gco:CharacterString>Nutzungsbedingungen: Der Datensatz/Dienst steht unter der
folgender Lizenz: Creative Commons Namensnennung (CC BY). Die Namensnennung hat in
folgender Weise zu erfolgen: "Datenquelle: Bayerische Vermessungsverwaltung –
```

```
www.geodaten.bayern.de".</gco:CharacterString>
        </gmd:useLimitation>
        <gmd:useConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/resou
rces/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="license">license</gmd:MD_RestrictionCode>
        </gmd:useConstraints>
        <gmd:useConstraints>
            <gmd:MD_RestrictionCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/resou
rces/codelist/ML_gmxCodelists.xml#MD_RestrictionCode"
codeListValue="otherRestrictions">otherRestrictions</gmd:MD_RestrictionCode>
        </gmd:useConstraints>
        <gmd:otherConstraints>
            <gco:CharacterString>Nutzungsbedingungen: Der Datensatz/Dienst steht unter der
folgender Lizenz: Creative Commons Namensnennung (CC BY). Die Namensnennung hat in
folgender Weise zu erfolgen: "Datenquelle: Bayerische Vermessungsverwaltung –
www.geodaten.bayern.de".</gco:CharacterString>
        </gmd:otherConstraints>
        <gmd:otherConstraints>
            <gco:CharacterString>{ "id": "cc-by", "name": "Creative Commons Namensnennung
(CC BY)", "quelle": "Datenquelle: Bayerische Vermessungsverwaltung –
www.geodaten.bayern.de", "url": "http://creativecommons.org/licenses/by/3.0/deed.de"
}</gco:CharacterString>
        </gmd:otherConstraints>
    </gmd:MD_LegalConstraints>
</gmd:resourceConstraints>
```

We find the exact same elements in the ISO metadata document as for the View Service with the Red Lock and the € symbol. In particular, we find the constraint "otherRestrictions" present as well as "license" and "copyright". However, the content one of the "otherConstraints" element now contain a cryptic content that seem to express a particular CC license for the "Bayerische Vermessungsverwaltung", even though this organization doesn't exist any longer (their successor organization is the "Landesamt für Digitalisierung, Breitband und Vermessung") and the provided link does result in HTTP 404.

Taking a look at the Capabilities document for the service unveils that both constraints are in place: Fees and AccessConstraints.

6.6 Misuse of <Fees> and <AccessConstraints> in the Capabilities document

As shown in this example, a common practice is that the XML elements Fees and AccessConstraints in the Capabilities document are not used in a standard compliant way.

For this example, where the portal screenshot indicates no access constraints (green lock), the Capabilities document indicates something different:

```
<Fees>Kostenfrei (mit allen Rechten)</Fees>

<AccessConstraints>Siehe Nr. 3.1 unter
http://geoportal.bayern.de/geoportalbayern/inhalte/nutzungsbedingungen.html</AccessConst
raints>
```

But according to the OGC WMS 1.1.1 specification, the only valid content of the element <Fees> and <AccessContraints> in case there are no fees or restrictions imposed is the string literal *none*. Interesting, also note that the content of the <AccessContraints> element refers to use restrictions and not to the actual access restrictions such as authentication or access control limitations.

This leads to the conclusion that the existing hooks for describing the security constraints in an ISO metadata document exist but are difficult to use, in particular in a consistent fashion. Also, the use of options in the capabilities response document that there are not proper hooks provided to describe common security constraints.

6.7 Lessons learned and Shortcomings of the example approach

The first observation that has nothing to do with the actual advertisement of security, license, use and fee constraints is that the entire process of obtaining information is very difficult to understand and requires a certain understanding of the bits and pieces that link together to provide all the puzzle pieces to get the complete understanding. One particular observation indicates that the process is also difficult for the provider side as different metadata files provide different details at different places that partly contradict or are incomplete.

Regarding the use of security constraints, the use of provided options from the ISO metadata seem to be difficult to understand, may not be interoperable with descriptions from other providers. The provided information is not sufficient to properly bind to the server.

For the given example, it is difficult to understand the requirements for the Bind. For the Download Service, there is no Capabilities document but an Atom Feed XML document. And for the Viewing Service, there is no Capabilities document available either as the GetCapabilities operation is not open; in fact that operation also requires an account to login.

The Atom feed document leverages yet another mechanism to advertise constraints using the following construct in the instance document of the feed:

```
<rights>
Es gelten die Nutzungsbedingungen und die Preisliste der Bayerischen
Vermessungsverwaltung. Diese können Sie auf der Internetseite der Bayerischen
Vermessungsverwaltung unter http://vermessung.bayern.de/service/Nutzungshinweise.html
nachlesen.
</rights>
```

Again, we find terms of use information and fee information (in German) that differs from the text used in the ISO metadata.

The general conclusion we can make from the example is that there is confusion on how to properly describe security constraints for OGC Web Services based on the currently available options. Description of security constraints is difficult to understand, insufficient or not fit for purpose and scattered among different documents: Pieces can be found in ISO metadata, Atom feed description documents and in the Capabilities document and these are inconsistent.

Further, consider that the description of security constraints is pretty useless when it comes to users that prefer Google for the shortcut to bind to OGC Web Services, even knowing that this is not the proper process. The simplistic use of the Capabilities document (either via bookmark or GetCapabilities) operations is used. This shortcut is also supported by the general desktop GIS implementations. Setting up OGC Web Services in mapping projects does not involve assessing the ISO metadata. Setup can simply happen from the Capabilities document. These observations recommend that any security constraints be advertised solely in the Capabilities document.

7 The Capabilities document and the options to describe security constraints

The general problem with the approach to describe security constraints in the ISO metadata is that shortcuts in the GetCapabilities document as the general OGC Bind is implemented in clients. This voids the entire approach to use the ISO metadata to advertise security constraints.

The important question at hand is if there are options in the Capabilities file to describe Common Security, in particular describe implementation details for the different frameworks outlined in the ISO 10181 series. Additionally, if there were options to describe the security constraints, is the information sufficient and fit for purpose?

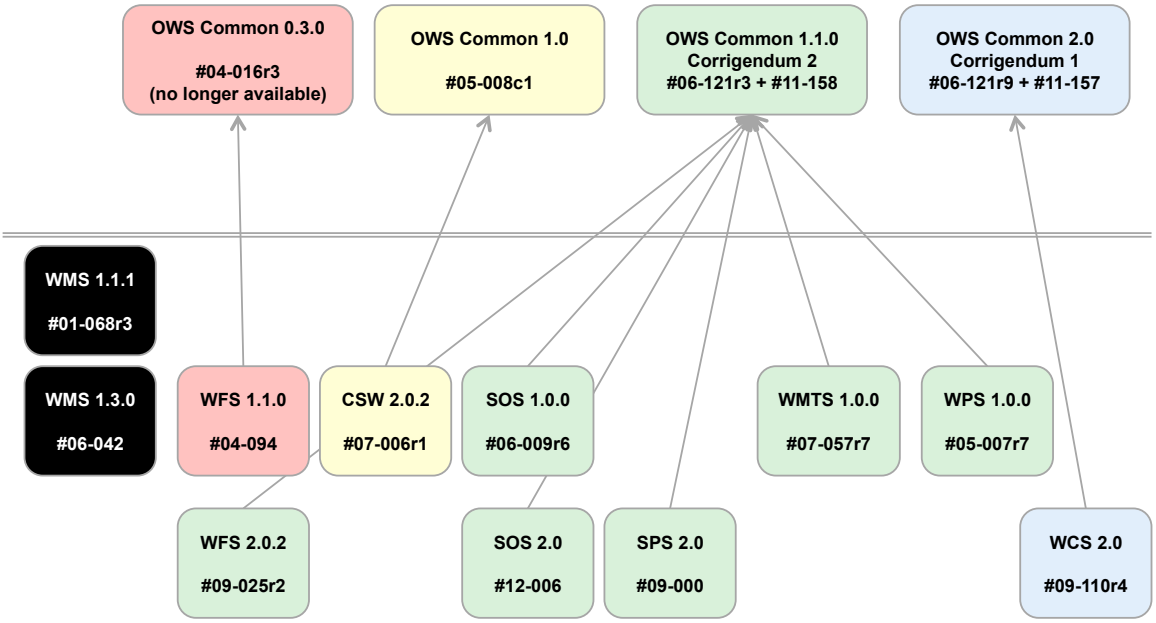
In order to determine the options in the Capabilities document, we first study the basic mechanics of OGC standardization for Web Services.

7.1 Commonalities across OGC Web Services

The first essential question is to evaluate if a common place could exist in the OGC Web Services standards world that defines how to advertise security constraints in a Capabilities document. Such a common specification would automatically provide all service standards the ability to inherit from that specification.

Those familiar with OGC Web Service standards would expect that the natural place for commonalities across the Web Service standards is OWS Common. A closer look at this issue quickly shows that this is not quite so.

Figure 8 — Use of OWS Common version across OGC Web Service specifications



Currently, there are four different version of OWS Common in use one of which is deprecated. This clearly indicates that the description of *Common Security* cannot simply be integrated into one single, commonly used base standard. For example, WMS 1.3 version – as an ISO standard – does not even link to OWS Common; WMS 1.3 has relevant parts from OWS Common restated in the normative text in the WMS document. However, from reading the standard, which version of OWS Common was used is not obvious.

7.2 Study of OGC Standards regarding *Common Security*

For describing *Common Security* requirements with OGC Web Services, we need to determine if any security related standards are normatively referenced from the involved standards. Further we need to check that no general IT requirements are overwritten which would prevent the use of general IT security / communication security standards as well as common libraries.

7.2.1 OGC Web Services Common Specifications

As OGC OWS Common is meant as a placeholder standard for “common things” across the Web Services standards, let’s study what the baseline for general IT standards and communication is about. This is the fundament necessary to build on top the *Common Security*.

7.2.1.1 OWS Common Version 1.0.0

OWS Common, version 1.0 is skipped here because only CSW 2.0.2 is linking it.

7.2.1.2 OWS Common Version 1.1.0

OWS Common, version 1.1.0 is a normative base for the following OGC Web Services: SOS 1.0, SOS 2.0, SPS 2.0, WFS 2.0.2, WMTS 1.0.0, WPS 1.0.0

For OWS Common, version 1.1.0 the general HTTP communication behavior is based on a normative reference to IETF RFC 2616. This limits communication to HTTP/1.1 and supporting secure communication via HTTPS (HTTP + TLS) is not mandatory for an implementation.

This version of OWS Common does not have a normative reference to IETF RF 2109 (HTTP Cookies). The fact that a service shall be stateless seems to be the goal, but the lack of HTTP Cookies disables the ability to establish communication/service sessions in a common and practical manner. As secured services are usually no longer stateless (there typically is a user or client context to maintain), the option of using HTTP Cookies is not given.

Exception reporting is limited to XML encoded exceptions using the "exceptionCode" mechanism with a pre-defined set of error codes with the HTTP code 200. It is unclear if the use of other status codes from the referenced RFC 2616 can be leveraged. In particular, 101, 302, 401, 403, 404, 500 are important to consider[10].

7.2.1.3 OWS Common Version 2.0

The main difference of version 2.0 over version 1.1.0 seems to be the option to use SOAP encoded messages for service in/output. So the limitation of HTTP communication (no TLS and not Cookies) and status codes remains.

Even though SOAP is enabled for OWS Common 2.0, the lack of enabling the WS-* stack of standards disable the implementation of confidentiality and integrity on service instances.

7.2.2 OGC Web Map Service Implementation Specification, version 1.3

Even though it is unclear if elements of WMS 1.3 were copied from a particular version of the OWS Common specification or introduced as the "common requirements", the same HTTP limitations as in OWS Common 1.1.0 exist: No HTTPS, no HTTP Cookies and no use of HTTP status codes other than 200.

7.2.3 Exception Codes

Common to OWS Common and WMS 1.3 is that the defined exception codes (to be encoded in XML) lack any security related definitions. So, for example, there is no code 403 "access denied" or 401 "authentication required". In difference to (or as an

[10] A detailed study was not undertaken for this ER.

improvement) to OWS Common 1.1.0, WMS 1.3 introduces its own extension to the exception handling from OWS Common:

"*Errors may arise in software modules other than those which implement the WMS, and may result in exception messages other than those defined by this International Standard. For example, when an error condition occurs in the local computing environment of the WMS server instance (e.g. out of memory or disk space), the server may be unable to process the WMS request and may issue an error message. Or, upon receiving a request that is invalid according to the rules of the Distributed Computing Platform in use, the server **may** issue a service exception of a type valid in that DCP (e.g. if the URL prefix is incorrect, an HTTP 404 status code (IETF RFC 2616) may be sent).*"[WMS 1.3]

The last sentence is unclear: Does the use of HTTP status codes overwrite the use of XML encoded WMS exception reporting? The term "may" is not helpful here!

And, how about the use of security related HTTP status codes, like 401, 403?

7.2.4 Expressiveness of Capabilties document of different OGC Web Services

Based on the many different versions of OWS Common and the OGC WMS standard not linking to OWS Common, there are many different versions with different levels of expressiveness when it comes to the OGC Capabilities documents.

7.2.4.1 WMS 1.1.1

WMS version 1.1 was standardized before OWS Common.

The WMS Capabilities for this version is based on DTD; no schema. This DTD includes elements to express access constraints and fees via the following elements:

```
<!ELEMENT Fees (#PCDATA)>
<!ELEMENT AccessConstraints (#PCDATA)>
```

Furthermore, it is possible to define vendor specific parameters according to the following element definitions:

```
<!ELEMENT Capability
          (Request, Exception, VendorSpecificCapabilities?,
       UserDefinedSymbolization?, Layer?) >
```

In particular, the VendorSpecificCapabilities is not defined in detail.

7.2.4.2 WMS 1.3.0

The WMS Capabilities for this version is based on an XML Schema[11] that is not linked to OWS Common.

The schema allows, as for 1.1.1, the definition of Fees and AccessContraints as well as extended Capabilities (vendor specific capabilities). This mechanism is used to advertise the INSPIRE specific version of the Capabilities.

Further, note that the WMS XSD constrains the definition of the protocol scheme for service instance endpoints to “http” and allows methods to “Get” and “Post”. The constraint to use HTTP prevents to service a WMS instance on HTTP**S**. This has the implication that you cannot even put a security proxy in from of the WMS 1.3 that mandates HTTP**S**, as you cannot define that in a standards compliant way.

7.2.4.3 WMTS 1.0

The WMTS Capabilities seem to have same expressiveness as the WMS 1.3.0 regarding Fees, AccessContraints. Further, it restricts the HTTP protocol to scheme http; and does not allow the scheme https.

When describing the actual service endpoints, the structure looks different and there is the ability to use elements named ows:Constraint. The definition is this:

```
<complexType name="RequestMethodType">
    <annotation>
        <documentation>Connect point URL and any constraints for this HTTP request method
for this operation request. In the OnlineResourceType, the xlink:href attribute in the
xlink:simpleAttrs attribute group shall be used to contain this URL. The other attributes
in the xlink:simpleAttrs attribute group should not be used. </documentation>
    </annotation>
    <complexContent>
        <extension base="ows:OnlineResourceType">
            <sequence>
                <element name="Constraint" type="ows:DomainType" minOccurs="0"
maxOccurs="unbounded">
                    <annotation>
                        <documentation>Optional unordered list of valid domain
constraints on non-parameter quantities that each apply to this request method for this
operation. If one of these Constraint elements has the same "name" attribute as a
Constraint element in the OperationsMetadata or Operation element, this Constraint
element shall override the other one for this operation. The list of required and
optional constraints for this request method for this operation shall be specified in the
Implementation Specification for this service. </documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

[11] http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd

Even though it is difficult to understand the proper use of the Constraint element, note that a definition exists and is from OWS 1.1.0 schema: owsOperationsMetadata.xsd

However, the WMTS Capabilities document introduces two new features: (i) reference a WSDL document and (ii) reference ISO metadata document:

```
<element name="WSDL" type="ows:OnlineResourceType" minOccurs="0" maxOccurs="unbounded">
    <annotation>
        <documentation>Reference to a WSDL resource</documentation>
    </annotation>
</element>
<element name="ServiceMetadataURL" type="ows:OnlineResourceType" minOccurs="0"
maxOccurs="unbounded">
    <annotation>
        <documentation>
            Reference to a ServiceMetadata resource on resource
            oriented architectural style
        </documentation>
    </annotation>
</element>
```

One side note on the complexity of the WMTS Capabilities schemas: There is a requirement for a number of individual schema files to work together, to be loaded, to validated, and as a result generate the correct Capabilities document.

One conclusion is that the WMTS Capabilities document may not contain a vendor specific part (or simply an extension) as was possible for the WMS 1.1.1 and WMS 1.3.

7.2.4.4 WFS 1.1.0

The WFS 1.1.0 implementation specification uses a normative reference to OWS Common 0.3.0 but the WFS Schema (http://schemas.opengis.net/wfs/1.1.0/wfs.xsd) includes the OWS Common 1.0.0 schema. This includes the OWS owsOperationsMetadata.xsd as of version 1.0.0.2. Please note that this version already introduces the option for ows:Constraint as explained in the next chapter.

7.2.4.5 WFS 2.0, WCS 2.0, SOS 2.0, SPS 2.0

The Capabilities document for the WCS 2.0 is based on a schema that includes OWS Common v 2.0. The other service Capabilities documents include the OWS Common 1.1.0 schema. Because OWS Common 1.1.0 and 2.0 provide the ows:Constraint, it is possible to use that element for each operation. Therefore, this seems to be the correct existing hook for including security description towards *Common Security*.

The following annotation of an existing WFS 2.0 Capabilities document illustrates an example approach for describing *Common Security* regarding Authentication and Access Control. The original part is shaded 10% grey.

```
<ows:Operation name="GetFeature">
    <ows:DCP>
        <ows:HTTP>
            <ows:Get xlink:href="http://maps.dgs.udel.edu:80/geoserver/dgs/wfs"/>
            <ows:Post xlink:href="http://maps.dgs.udel.edu:80/geoserver/dgs/wfs"/>
        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="AcceptVersions">
        <ows:AllowedValues>
            <ows:Value>1.0.0</ows:Value>
            <ows:Value>1.1.0</ows:Value>
            <ows:Value>2.0.0</ows:Value>
        </ows:AllowedValues>
    </ows:Parameter>
    <ows:Parameter name="AcceptFormats">
        <ows:AllowedValues>
            <ows:Value>text/xml</ows:Value>
        </ows:AllowedValues>
    </ows:Parameter>
```

```
    <ows:Constraint name="authentication">
        <ows:AllowedValues>
            <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:browser</ows:Value>
            <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:ecp</ows:Value>
        </ows:AllowedValues>
        <ows:Metadata
xlink:href="http://www.unibw.de/.../inspire/authCodelists.xml#AuthenticationCode"/>
    </ows:Constraint>
    <ows:Constraint name="access">
        <ows:AllowedValues>
            <ows:Value>urn:tbd:policy:GetFeature</ows:Value>
        </ows:AllowedValues>
        <ows:Metadata
xlink:href="http://www.unibw.de/.../inspire/accessCodelists.xml#AccessCode"/>
    </ows:Constraint>

</ows:Operation>
```

7.2.4.6 CSW 2.0.2

The Capabilities document for the CSW 2.0.2 is based on a schema that includes OWS Common v 1.0. Regarding the option to describe ows:Constraint for each operation, this is possible as for the OWS Common 1.0 schema.

7.2.5 Implications to existing approaches to secure OGC Service instances

After the study of the OGC Web Services standards and their common base OWS Common, apparently there is no standardized support to host an OGC Web Service instance over HTTP**S** and by that to implement the Confidentiality and Integrity frameworks. To the editor's understanding, the use of HTTPS (HTTP over TLS) would require a normative reference to RFC 2818.

Also, the approach for implementing the Authentication Framework via HTTP Authentication seem to be invalid, because all service requests require that the HTTP Authorization Header element is present or the service responds with an invalid HTTP status code. To the editors understanding, the use of HTTP Authentication that enables the use of HTTP status code 401 requires a normative reverence to RFC 2617.

In addition to the above, putting a complete service instance (including all operations) behind HTTP Authentication has the implication that it breaks the OGC Find / Bind mechanics. As explained in an earlier section of this ER, common practice for a client is to connect to a service using the GetCapabilities operations. So in a case where the entire service and all operations are locked, it is not possible to base a description of *Common Security* inside the Capabilities document. In order to overcome this, an implementation can host the Capabilities as a publicly accessible XML instance document. In other words, the Find / Bind must be based on a publically accessible Capabilities document.

7.3 OWS Operations Metadata

All OGC Web Services except WMS 1.x and 1.3 include the ability to describe the Operations Metadata in a Capabilities document. The schema to describe the operations metadata is inside the file "owsOperationsMetadta.xsd" which does exist on the OGC Schema Web Server (http://schemas.opengis.net) as local copies in each directory for OWS:

Table 6 — OWS Operations Metadata Schema files

OWS Common 1.0	http://schemas.opengis.net/ows/1.0.0/owsOperationsMetadata.xsd
OWS Common 1.1.0	http://schemas.opengis.net/ows/1.1.0/owsOperationsMetadata.xsd
OWS Common 2.0	http://schemas.opengis.net/ows/2.0/owsOperationsMetadata.xsd

At first sight, all files have a different data and different size. The important aspect is that all version have the same identical definition for ows:Constraint:

```
<element name="Constraint"
    type="ows:DomainType"
    minOccurs="0"
    maxOccurs="unbounded">
    <annotation>
        <documentation>Optional unordered list of valid domain constraints
            on non-parameter quantities that each apply to this server. The
            list of required and optional constraints shall be specified in
            the Implementation Specification for this service.</documentation>
    </annotation>
</element>
```

It is also very important that all version of OWS Operations Metadata define the concept of extended Capabilities that basically introduces the option of inserting XML elements using XSD anyType:

```
<element name="ExtendedCapabilities"
    type="anyType">
    <annotation>
```

```
        <documentation>Individual software vendors and servers can use this
            element to provide metadata about any additional server
            abilities.</documentation>
    </annotation>
</element>
```

8 Missing standards to enable Common Security and implications of their use

As outlined before, there are two typical approaches feasible when implementing common security: (i) The first approach can be based on all tools that the communication layer (HTTP) does provide; (ii) The second approach can be based on message security leveraging SOAP + WS-*.

The following two sub-sections outline which common (main stream) IT standards from other standardization organizations must be normatively referenced from OGC Web Services to enable either approach.

8.1 Implementation of the *Common Security* based on HTTP

If the implementation of the introduced frameworks as Common Security be applied leveraging HTTP specific options only, then should the following list of standards be normatively referenced from an OGC Web Service standard and adoptions in the OGC standards text be applied? The following are specific recommendations.

8.1.1 HTTP/1.1 (IETF RFC 2616)

The communication base is HTTP version 1.1. A OGC Web Service implementation or the hosting Web Server shall support all HTTP verbs and not limit the communication to GET and POST, even though these are the potentially the most dominantly used verbs.

The reporting of exception codes shall take place with precedence for HTTP if communication protocol related. The exception codes listed in OGC Web Service standards, which by majority are related to geospatial, shall be used in all other cases. However, for REST or RESTful implementations, the XML based exception codes should become available as OGC specific HTTP status codes.

8.1.2 HTTP Authentication (IETF RFC 2617)

For support in implementing the Authentication Framework with support of the HTTP communication protocol using Basic and Digest, RFC 2617 shall be normatively referenced from all OGC Web Services standards.

To enable more sophisticated authentication options such as OAuth2 and SAML, other appropriate standards shall be normatively referenced. So for support of OAuth2 Bearer

Tokens, RFC 6749 and in particular the use of Authorization Bearer shall be permitted. For supporting the SAML based (federated) authentication, no authentication specific requirements exist for the service implementation. However, implications exist for the implementation of a (desktop) client.

It should be pointed out that the use of HTTP Authentication is only recommended if HTTP over TLS is enabled.

8.1.3 HTTP + TLS (IETF RFC 2818)

Implementation of the Confidentiality and/or Integrity framework(s) via HTTP over TLS or better HTTPS require that OGC Web Service standards normatively reference RFC 2818. The proper rollout does imply that additional standards and practices are in place to ensure proper Public Key management (PKI), but the details for that are out of scope for this ER, as no implications exist to the OGC standardization. However, it is important to ensure that implementations leverage the CRL mechanism and verify root certificates on SSL certificates properly. Trusting all certificates is not good and in particular not a secure practice. At least timestamp and hostname vs. certificate common name checks should be implemented.

It should be mandated for an OGC Web Service Instance that has implemented Common Security that the only communication scheme is HTTP**S**; any HTTP based communication shall be disallowed.

8.1.4 HTTP Cookies (IETF RFC 2965)

Even though the general state of an OGC Web Service shall be stateless, one common practice is to reference to a security session via HTTP cookies. In particular for high performance solutions, the storage of pointers to a common security context using cookies is an acceptable practice. In order to enable to maintain a communication and security state for a particular client / service interaction, the normative reference of RFC 2965 for all OGC Web Service standards is required.

HTTP Cookies are a particular kind of HTTP header and therefore available for HTTP and HTTP over TLS (HTTPS). In order to ensure that Cookies can only be used for HTTP, they should be tagged for “http” use. In case a Cookie shall be used for HTTP**S** connections only, it should be tagged “secure” in addition.

Therefore, when implementing the Integrity / Confidentiality Framework, clearly state in the OGC Web Service standard that Cookies are to be tagged for “http”, “secure” communications only and in addition be linked with the most exclusive path and are properly time constrained.

The use of persistent Cookies should be prevented, unless they do not introduce a security risk. For example, in the context of an Access Management Federation, a persistent cookie may be used to persistently store the choice of the login entity. The

choice of entity does not introduce a security risk, as the actual login credentials – of course – are never stored in any Cookie.

8.2 Implementation of the *Common Security* based on SOAP

Implementation of the outlined frameworks using SOAP comes with the interoperability limitations as outlined earlier in section 4.4.3.3. However, the use of SOAP (XML) encoded requests introduces the option to leverage WS-Security and related standards to build your own interoperability stack for the Integrity, Confidentiality and Authentication frameworks independent from the communication layer: HTTP. However, the use of HTTP+TLS can be seen as an optional improvement.

In order to support the use of WS-* based implementation of Common Security, the OGC Web Service standards must normatively reference the appropriate suite of Web Security standards. Detailing which standards these are in detail is outside the scope of this ER. However, this ER includes the relevant standards in the Reference section and a basic view of the WS-* family standards are given in figure 4. A comprehensive introduction that may help to conclude is available in [1].

Note that the use of WS-Security applies encryption to XML and therefore, the W3C standards XML Digital Signatures and XML Encryption are also mandatory. In addition, good practice is to remind implementers that the use of XML Signature introduces many pitfalls like XML canonicalization and digital signatures on external transformations that may result in applying integrity to any content. Therefore, we recommend considering the W3C Best Practices on how to use XML Digital Signatures (see [23] for details).

8.3 Implementation of the *Common Security* on the client side

When introducing the options for implementing Common Security on the service side, the standardization must pick-up on the client side as well and give normative guidance what to implement, how to process requests and responses and in particular how to act on exceptions.

Since OWS Common does not reference any security related standards creates a huge disadvantage, as most clients for OGC Web Services do not or only partly support the implementation of the different security frameworks as *Common Security*. This limitation is very important, as on the server side security enabled proxies can be deployed to add *Common Security*.

In order to outline relevant requirements and standards to implement Common Security on clients that are able to interact with secured OGC Web Services, consider functionalities common to main stream IT clients. For a better classification of client types, this ER separates applications that are executed in a Web Browser (Web Browser applications) and applications that are executed on the OS (desktop clients).

8.4 Support for Common Security in (typical modern) Web Browser based applications

Obviously, a Web Browser application is able to leverage general and in particular security related functions provided by the executing container; the Browser. In addition, the execution inside a Browser introduces limitations caused by the security sandboxing to prevent content injection attacks leveraging cross-site scripting. Even though it is outside the scope of this ER to go into details, two dominant limitations exist: (i) Same Origin (JavaScript related functions are limited to call content from "same" domains) and (ii) Mixed-content is either blocked or flagged in the browser (application that was loaded via HTTP uses content loaded over HTTP). The implementation has to overcome these (and other) Browser constraints. There is no explicit guidance or standardization required from OGC.

Modern Web Bowser usually support HTTP (RFC 2616), HTTP over TLS (RFC 2818), HTTP Authentication (RFC 2617) and HTTP Cookies (RFC 2965) which already enables to implement *Common Security* based on HTTP as outlined in sections 4.4.1 and 4.4.2.

Important is also the built-in support of general processing functions that are simply available when implementing a client application to run as a Browser app:

1) JavaScript provides a wide spectrum of important processing functions.

2) Processing of (X)HTML content with support for automatic handling if JavaScript is enabled

3) Processing of HTTP status codes and HTTP Cookies

4) Validation of SSL/TLS certificate when using HTTP+TLS connections, with the option of checking CRLs

5) Support for mine/type specific execution of external applications

Observing the general functions of a Web Browser unveils that XML processing support is not available.

8.5 Support for *Common Security* in desktop Applications

In contrast to web applications that run inside a Web Browser, a desktop application is executed on the Operating System. This implies that there is no pre-existing skeleton of communication and processing functions to leverage. Any function must be integrated by importing the right library. Further, the proper use of the library must be enabled. One non-trivial example among many is the proper validation of HTTPS connections with support of HTTP cookies over HTTP redirects.

Therefore, logically OGC standardization should mandate a particular set of HTTP communication: general and security specific IT functions that are to be implemented

with a client that is capable to interact with an OGC Web Service that offers *Common Security*. Contrary to applications that run inside a Web Browser, the extensive processing of XML is usually part of desktop applications. In that sense, it seems reasonable to believe that these clients should be able to support *Common Security* implemented using SOAP + WS-*.

Desktop applications should be able to support different implementations of the Authentication framework. In particular, an implementation shall support RFC 2617 (HTTP Authentication), RFC 6749 (OAuth2 Tokens), RFC 5246 (TLS mutual) and SAML ECP. The support for SAML as a federated authentication solution is (according to the Gardner Group) gaining momentum for enterprises to interconnect, independent from security domains. Some requirements:

- Validation of SSL/TLS certificates is required.
- Support HTTP Cookies (RFC 2965) to maintain a security context across requests is required.
- Desktop client use of HTTP status codes and processes OGC specific XML encoded exceptions is required. The latter can usually be limited to display the content of the exception to the user rather than programmatic actions.

9 Conclusion, Recommendations to OGC standardization and future work

9.1 Conclusion

The conclusion: Can a method for Common Security for OGC standards be defined? A definition is **not** possible within the current suite of OGC Standards.

The main reason is the lack of an OGC common security model that can be implemented by OGC Web Services. The most dominant example is the limitation to use HTTP as constrained by all OWS Common versions schemas (operationsMetadata.xsd) that prevents service operations implementing HTTP over TLS to address the integrity and confidentiality frameworks:

OWS Version 1.0: http://schemas.opengis.net/ows/1.0.0/owsOperationsMetadata.xsd

OWS Version 1.1.0: http://schemas.opengis.net/ows/1.1.0/owsOperationsMetadata.xsd

OWS Version 2.0: http://schemas.opengis.net/ows/2.0/owsOperationsMetadata.xsd

```
<element name="DCP">
    <annotation>
        <documentation>Information for one distributed Computing Platform (DCP) supported
for this operation. At present, only the HTTP DCP is defined, so this element only
includes the HTTP element.
        </documentation>
    </annotation>
    <complexType>
```

```
        <choice>
            <element ref="ows:HTTP"/>
        </choice>
    </complexType>
</element>
```

9.2 Recommendations for OGC Standardization

This chapter outlines a few short recommendations to address the “missing bits” required to enable security. Please read the entire Engineering Report to find all recommendations[12].

9.2.1 Define a common security architecture

In order to guarantee interoperability for OGC Web Services that have implemented one or multiple of the outlined security frameworks (protected services), the OGC members need to define a *Common Security Architecture* for OGC Web Services and Clients. This could be achieved by establishing a Web Services Security SWG which charter to include to define a Common Security Capabilities extension, WSDL documents including guidance how to embed WS-* and WS-Policy when using SOAP as well as defining a common approach to the OGC Publish / Find / Bind paradigm for protected services.

The crafting of that charter could be take place within the realm of the OGC Security Working Group, perhaps in liaison with the OGC Architecture DWG.

9.2.2 All OGC Web Services standards to include Security considerations

For each OGC Web Service standard that endorses the use of security by including security standards related normative references as outlined before, each and every standard shall contain a section on security considerations. This section shall outline security implications based on the data and processing model and in particular on the operations of the service.

The implications should consider that implementation of all the introduced ISO frameworks are required. Therefore, the implications towards the implementation of the authentication, access control, confidentiality, and integrity but in particular the non-repudiation framework shall be included.

[12] The editor decided not to copy and paste all recommendation into one single section as it would mean to separate the recommendation from the meaningful context.

9.2.3 Define and Describe *Common Security* in Capabilities document

The description for the implementation of the security frameworks, as part of the Common Security, should be in the Capabilities document. In particular, ows:Constraint shall be included for each operation.

```
<ows:Constraint name="authentication">
  <ows:AllowedValues>
    <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:browser</ows:Value>
    <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:ecp</ows:Value>
    </ows:AllowedValues>
  <ows:Metadata
xlink:href="http://www.unibw.de/.../inspire/authCodelists.xml#AuthenticationCode"/>
</ows:Constraint>
```

9.2.4 GetCapabilities operation

In order to provide the common hook ows:Constraint for describing common security constraints on service operations, either:

- the GetCapabilities operation is publically accessible and not protected, or
- a publically accessible Capabilities instance document is hosted on a web server that contains the <Operations> section including the security description.

In the case where the served content is classified, the full list of data offerings is only returned if the user issues the GetCapabilties request as a recognized user. This implies the use of the publically assessable Capabilities instance document that does not contain the classified data offerings but outline the GetCapabilities operation as protected. This ensures that the data offerings are not published to anonymous users, but authorized users can bind to the service by following the protected GetCapabilities operation.

Table 7 — Example Capabilities document with no data offerings

```
<ows:OperationsMetadata>
    <ows:Operation name="GetCapabilities">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get xlink:href="http://maps.dgs.udel.edu:80/geoserver/dgs/wfs"/>
                <ows:Post xlink:href="http://maps.dgs.udel.edu:80/geoserver/dgs/wfs"/>
            </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="AcceptVersions">
            <ows:AllowedValues>
                <ows:Value>1.0.0</ows:Value>
                <ows:Value>1.1.0</ows:Value>
                <ows:Value>2.0.0</ows:Value>
            </ows:AllowedValues>
        </ows:Parameter>
        <ows:Parameter name="AcceptFormats">
            <ows:AllowedValues>
                <ows:Value>text/xml</ows:Value>
            </ows:AllowedValues>
        </ows:Parameter>
        <ows:Constraint name="authentication">
```

```
            <ows:AllowedValues>
                <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:browser</ows:Value>
                <ows:Value>urn:oasis:names:tc:SAML:2.0:profiles:SSO:ecp</ows:Value>
            </ows:AllowedValues>
            <ows:Metadata
xlink:href="http://www.unibw.de/.../authCodelists.xml#AuthenticationCode"/>
        </ows:Constraint>
        <ows:Constraint name="access">
            <ows:AllowedValues>
                <ows:Value>urn:tbd:policy:GetFeature</ows:Value>
            </ows:AllowedValues>
            <ows:Metadata
xlink:href="http://www.unibw.de/.../accessCodelists.xml#AccessCode"/>
        </ows:Constraint>
    </ows:Operation>
</ows:OperationsMetadata>
<!--
    <FeatureTypeList/>
-->
<fes:Filter_Capabilities>
    ...
</fes:Filter_Capabilities>
```

The example Capabilities instance is valid even though the <FeatureTypeList> element is missing.

9.2.5 OWS Common limitations on DCP element

For future versions of the common schema,

- Relax the option to use protocol scheme HTTP**S** of a service operation description.

- Provide guidance when to outline support for the HTTP GET and POST method in a capabilities document, referring to the same operation. As an example, it would be perfectly OK to allow for GetCapabilities just HTTP GET and for GetFeature only HTTP POST, because the security framework is only available for POST but not for GET. Disallow a service to jump from one protocol verb to another for the same operation once a session is established to relax the implementation of a or more security frameworks.

9.2.6 WMS

The 1.3.0 version of the WMS does not support the use of ows:Constraint as it does not reference the OWS Common schema. Therefore, adopt the WMS Capabilities DTD for a WMS 1.3 upon next revision to allow the use of ows:Constraint as in the other OGC Web Services.

For the WMS 2.0 standardization, make sure WMS 2.0 leverages the OWS Common schema to maintain continuity in the use of ows:Constraint.

9.2.7 Use of common IT security standards

OGC Web Service standards shall normatively reference mainstream IT standards on security that are required to implement the ISO frameworks listed above.

9.2.8 Client side standardization

OGC must also specify how a client implementation shall work. In particular that it implements the main stream IT security related standards as outlined in section 8.

9.3 Future work

The following are draft examples into the definition of a description for Common Security to all OGC Web Services. However, as this report focuses on the general options how to base descriptions, the following examples must be considered rudimentary.

9.3.1 Define and Describe *Common Security* in Capabilities document

Based on the recommendation to integrate a description to Common Security inside the operation metadata of a Capabilities document, the details and semantics must be defined. In particular, the available names for ows:Constraint shall be defined. The following names may be used as an indication towards the meaning of a constraint to be indicator for an implementation of a particular framework:

Table 8 — Framework implementation key examples

ISO Security Framework	Constraint Name
Authentication	urn:ogc:def:security:authentication
Access Control	urn:ogc:def:security:access
Integrity	urn:ogc:def:security:https urn:ogc:def:security:ws-security
Confidentiality	urn:ogc:def:security:https urn:ogc:def:security:ws-security

The actual values of the constraint identifiers are defined in the actual ISO compliant code list that is referenced via the ows:Metadata element. In that sense, the ows:AllowedValues are non OGC URNs. They relate to other standardization organizations identifiers and relate to their standardization. The above example references a code list that defines two profiles of OASIS SAML for authentication.

9.3.2 Expressiveness of the Capabilities document to define *Common Security*

As illustrated above, different OGC Web Services are based on different versions of OWS Common. From the perspective to find a common ground how to express Common Security constraints in the OGC Capabilities document, unfortunately WMS 1.1.1 and WMS 1.3 Capabilities are not based on an OWS Common schema.

It is recommended that WMS 2.0, as currently work in progress, shall provide the ability to use the same definition for OperationsMetadata as defined in OWS Common 1.0, 1.1.0 and 2.0. This would ensure that for WMS 2.0 the same description mechanism could be used.

It is also recommended that guidance be created how to use the ows:Constraint element in the Operations Metadata regarding the description of the Common Security. Perhaps the guidance is to not use this element to express the existence of security framework implementations but the capabilities extension mechanism instead.

9.3.3 Define a Codelist for supported Authentication options

In order to guarantee interoperability with authentication, it is recommended to standardize the allowed methods. So basically, define the OGC code list for supported authentication methods.

9.3.4 Define a GeoXACML or XACML policy based codelist for Authorizaiton

The codelist for authorization should outline access rights (or constraints) using a standards based description. It is recommended to base the description on the OASIS XACML or OGC GeoXACML standards.

Annex A Revision history

Date	Release	Editor	Primary clauses modified	Description
22.04.2015	0.1	AM	All	1st draft
05.05.2015	0.2	AM	All	2nd draft
June 9 2015	N.A.	Carl Reed	Various	For publication
12.06.2015	0.3	AM	All	Addressing final comments for publication

Bibliography

[1] http://schemas.opengis.net

[2] http://java.globinch.com/enterprise-java/web-services/jax-ws/secure-metro-jax-ws-usernametoken-web-service-signature-encryption/

