OGC Web Services Security

Table of Contents

Open Geospatial Consortium

Submission Date: 2018-05-17

Approval Date: 2018-08-28

Publication Date: 2019-01-28

External identifier of this OGC® document: http://www.opengis.net/doc/IS/security/1.0

URL for this OGC® document: http://docs.opengeospatial.org/is/17-007r1/17-007r1.html

Additional Formats (informative): https://docs.opengeospatial.org/is/17-007r1/17-007r1.pdf

Please refer to the errata for this document.

Internal reference number of this OGC® document: 17-007r1

Version: 1.0

Category: OGC® Implementation Standard

Editor: Andreas Matheus

OGC Web Services Security

Warning

This document is an OGC Member approved international standard. This document is available on a royalty free, non-discriminatory basis. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type:	OGC® Standard
Document subtype:	
Document stage:	Approved
Document language:	English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER’S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR’s sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize

you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

i. Abstract

Information Assurance (IA) [1: **Information assurance** (**IA**) is the practice of assuring information and managing risks related to the use, processing, storage, and transmission of information or data and the systems and processes used for those purposes. Wikipedia, March 7, 2018] Controls for OGC Web Services (OWS) have been implemented for years. However, these implementations break interoperability, as they are not standardized by OGC Web Service standards. Interoperability between secured OGC Web Services and clients is limited to systems custom built to work with an IA implementation.

The goal of the OWS Common Security Standard is to allow the implementation of IA controls and to advertise their existence in an interoperable way with minimal impact to existing implementations using a backwards-compatible approach. That goal is being pursued in two ways:

1. Identify and document IA Controls for supporting authentication in a register maintained through the OGC.
2. Specify how a service can advertise their IA controls through the Service Capabilities Document.

This OGC standard applies to OWS deployed on HTTPS. It specifies how conformant OWS Services shall advertise their IA Controls and additional security features. The advertisement uses XML elements that are already part of the Capabilities document structure. This ensures that existing client implementations will not break.

The standard also describes the governance process for the IA Control registers, examples of register contents, and descriptions on how this information should be used.

Next, this standard defines conformance classes and requirements classes to be used for reaching compliance and their validation via conformance tests.

Finally, this standard defines client behavior to ensure interoperable processing of advertised security controls.

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, Security, OWS Common, OWS Common Security, OGC Web Services Security, OAuth2, OpenID Connect, SAML2, HTTPS, WS-Security, WS-Policy, SOAP, WMS, WFS, WCS, WMTS, XACML, GeoXACML, Authentication, Access Control

iii. Preface

This is version 1.0 of the OGC Web Services Security standard submitted to the Technical Committee by the OWS Common – Security Standards Working Group.

This document standardizes an annotation mechanism for Capabilities documents or responses to the GetCapabilities request, ensuring interoperability between a secured OGC Web Service instance deployed on HTTPS and a client application. It further overrides existing HTTP protocol limitations and exception handling for existing OGC Web Services standards for the purpose of achieving interoperability with main stream IT security standards and their implementations. To achieve this,

no changes to existing OGC Abstract specifications and OGC Web Services standards are required.

This standard has no direct precursor document but can be seen as the result of previous OGC Testbeds, documented in different Testbed Engineering Reports (see bibliography).

The annotation approach and the service behavior regarding security is standardized in a backwards compatible way to ensure it can be applied to **existing** OGC Web Services with no change to the deployments.

The implication to be compliant with this standard is that some requirements from existing OGC standards are superseded. Because this standard defines the compliance, it is not required to incorporate the requirements into the existing standards. Therefore no change requests to the existing OGC standards are required!

Uptake of the standardized approach by **new** OGC Web Services standards will ensure security interoperability.

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.

iv. Submitting organizations

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

Organization name(s)

- University of the Bundeswehr
- NGA
- Geonovum
- WiSC
- DigitalGlobe

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name	**Affiliation**
Andreas Matheus	University of the Bundeswehr
Dave Wesloh	NGA
Frank Terpstra	Geonovum
Chuck Heazel	WiSC

Michael Leedahl	DigitalGlobe

Chapter 1. Scope (informative)

This standard applies to a deployed OGC Web Service instance for which the protocol scheme of all operation endpoint URLs, exposed in the Capabilities document, is ‘https’ as defined in RFC 7230, section 2.7.2.

A security-annotated Capabilities document is one which uses the <Constraint> element(s) to express the existence of security controls on the operation of the service instance or support for a particular security feature. Applying the tests as defined in the Annexes can validate compliance for a service, the client and the OGC management process. Basically, a service can be described by a Capabilities document that includes security annotations as defined in this standard. A client loading these Capabilities and parse for the <Constraint> element(s) can determine the security controls implemented for each operation of the service instance. The string value of this element’s name attribute contains the identifier of the implemented requirements class.

How the client obtains the security-annotated capabilities is out of scope for this standard.

This standard defines one common abstract Requirements Class and three Capabilities document structure specific Requirements Classes. The structure specific classes address how the requirements are implemented for WMS 1.1.1, WMS 1.3 and OWS Common based service Capabilities documents.

Requirements Class **Common Security**: This abstract class ensures that the service instance is implementing HTTPS as specified by the IETF RFCs [e.g. RFC 7230]. This is the minimum capability required to be interoperable with mainstream IT security technology. Common Security bundles mandatory requirements classes that address issues which inhibit operating an OGC compliant web service over HTTPS. This Requirements Class also provides a method for the client to use either the service exception handling compliant with OWS Common (for the OWS layer) or exception handling compliant with the HTTP specification for the security layer. This method ensures the elimination of unnecessary limitations regarding the HTTP protocol and exception handling from OWS Common and other OGC Web Service standards. This standard also defines other optional requirements classes that address the description of further IAs to be able to convey as much information on existing security controls as possible.

The following Requirements Classes are concerned with how to apply the actual security annotations to the Capabilities document that is associated with a service endpoint. There are three different Requirements Classes because the way to insert security annotations into the Capabilities document differs based on the underlying XML schema or DTD.

Requirements Class **OWS Common**: This class defines how the security metadata is to be inserted into the OGC Web Service Capabilities document for any service instance based on OWS Common XML schema.

Requirements Class **WMS 1.1.1**: This class defines how the security metadata is to be inserted into the OGC Web Service Capabilities document for a WMS 1.1.0 service instance based on the WMS 1.1.1 DTD.

Requirements Class **WMS 1.3.0**: This class defines how the security metadata is to be inserted into the OGC Web Service Capabilities document for a WMS 1.3.0 service instance based on the WMS

1.3.0 XML schema.

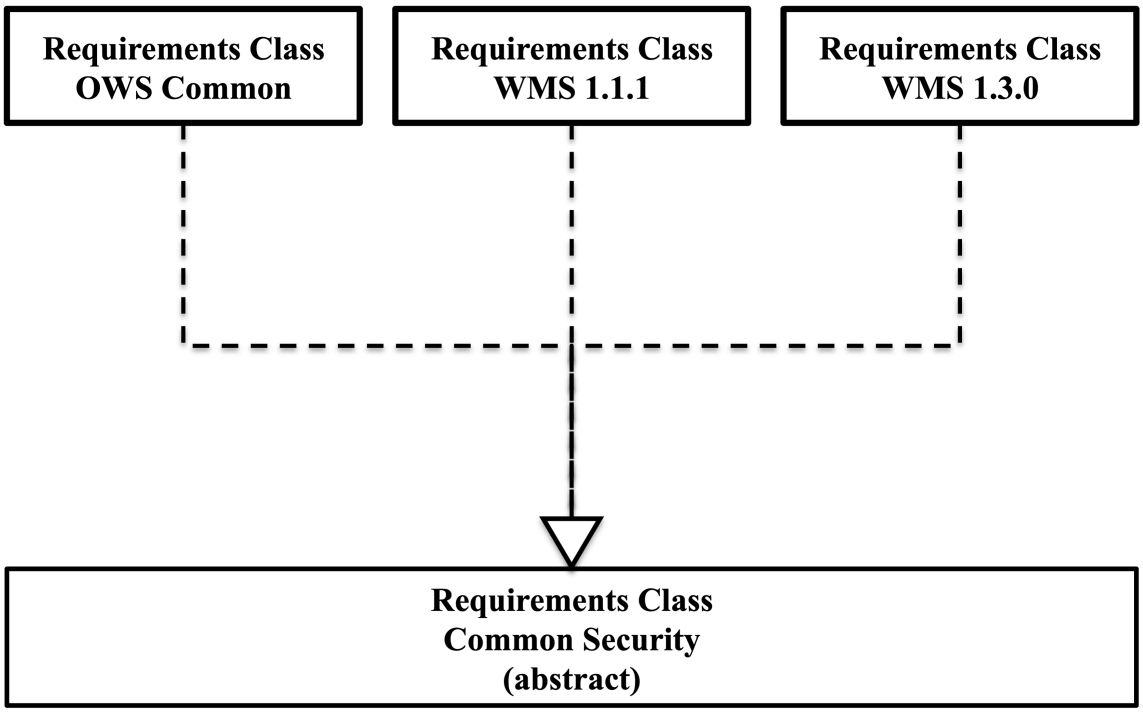

Figure 1. Relationships between requirements classes (simplified overview)

The implication to be compliant with this standard is that some requirements from existing OGC standards are superseded. Because this standard defines the compliance, it is not required to incorporate the requirements into the existing standards. Therefore no change requests to the existing OGC standards are required!

The following OGC standards are directly affected:

1. OWS Common 1.1.0, OGC 06-121r3 *OGC Web Services Common Specification, OGC® Implementation Standard*
2. OWS Common 2.0.0, OGC 06-121r9 *OGC Web Services Common Specification, OGC® Implementation Standard*
3. WMS 1.1.1, OGC 01-068r3 *Web Map Service Implementation Specification*
4. WMS 1.3.0, OGC 06-042 *OpenGIS Web Map Service (WMS) Implementation Specification*

The following OGC standards are impacted because they inherit from OWS Common.

1. WFS 1.1.0, OGC 04-094 *OpenGIS Web Feature Service (WFS) Implementation Specification*
2. WFS 2.0, OGC 09-025r1 *OpenGIS Web Feature Service 2.0 Interface Standard (also ISO 19142)*
3. WFS 2.0.2, OGC 09-025r2 *OGC® Web Feature Service 2.0 Interface Standard – With Corrigendum*
4. WCS 2.0, OGC 09-147r3 *OGC® WCS Interface Standard - KVP Protocol Binding Extension, version 1.0.1*
5. WCS 2.0, OGC 09-148r1 *OGC® WCS - XML/POST Protocol Binding Extension, version 1.0.0*
6. WCS 2.0, OGC 09-149r1 *OGC® Web Coverage Service 2.0 Interface Standard - XML/SOAP Protocol Binding Extension, version 1.0.0*

7. WMTS 1.0, OGC 07-057r7 *OpenGIS Web Map Tile Service Implementation Standard*
8. WPS 1.0.0, OGC 05-007r7 *Web Processing Service*
9. WPS 2.0, OGC 14-065 *OGC® WPS 2.0 Interface Standard*
10. SOS 2.0, OGC 12-006 *OGC® Sensor Observation Service Interface Standard*
11. SPS 2.0, OGC 09-000 *OGC® Sensor Planning Service Implementation Standard*
12. CSW 2.0.2, OGC 07-006r1 *OpenGIS Catalogue Service Implementation Specification*
13. CSW 3.0, OGC 12-176r7 *OGC® Catalogue Services 3.0 Specification - HTTP Protocol Binding*

One typical way to realize compliance without modifying the existing service implementations is via a security gateway or proxy. This proxy would have the duty to implement the compliance by injecting security annotations into the GetCapabilities response, operate the service endpoint on HTTPS but also support all HTTP methods and correct the OWS Common error code handling. Testbed 12 ER OGC16-048 describes a practical approach of a security proxy.

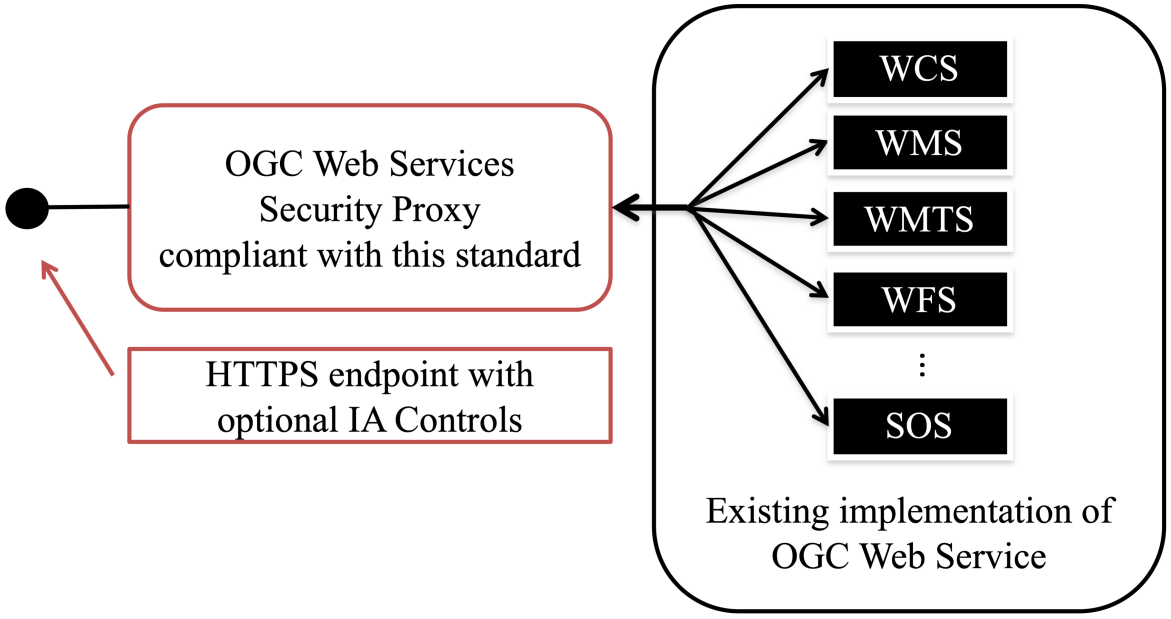

Figure 2. Security Proxy to make Geoserver deployment compliant with this standard

Chapter 2. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of 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.

2.1. Normative references

1.	J. Franks et.al.: HTTP Authentication: Basic and Digest Access Authentication, IETF,,https://tools.ietf.org/html/rfc2617[https://tools.ietf.org/html/rfc2617]
2.	Fielding, R.: Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing [2: Obsoletes RFC 2616 – HTTP 1.1],,IETF, https://tools.ietf.org/html/rfc7230
3.	Rescorla, E.: HTTP over TLS – RFC 2818, IETF, https://tools.ietf.org/html/rfc2818
4.	Sakimura, N.: OpenID Connect, OpenID Foundation, http://openid.net/specs/openid-connect-core-1_0.html
5.	Sakimura, N.: OpenID Connect Discovery, OpenID Foundation, https://openid.net/specs/openid-connect-discovery-1_0.html
6.	Nottingham, M.,Well Known URIs, IANA, https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml
7.	Hardt, D.: The OAuth 2.0 Authorization Framework, IETF, https://tools.ietf.org/html/rfc6749
8.	Jones, M.: The OAuth 2.0 Authorization Framework: Bearer Token Usage, IETF, https://tools.ietf.org/html/rfc6750
9.	Cantor, S.: Security Assertion Markup Language (SAML) v2.0, OASIS, https://www.oasis-open.org/standards#samlv2.0
10.	Kemp, J.: Authentication Context for the OASIS, Security Assertion Markup Language (SAML) V2.0, OASIS, https://www.oasis-open.org/standards#samlv2.0
11.	Moses, T.: eXtensible Access Control Markup Language, OASIS, https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
12.	Matheus, A: Geospatial eXtensible Access Control Markup Language, OGC, http://www.opengeospatial.org/standards/geoxacml
13.	van Kesteren, A.W3C CORS (Common Object Resource Sharing), W3C, https://www.w3.org/TR/cors/
14.	Ralphson, M.: OpenAPI 3.0, https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md
15.	OGC: OWS Common 1.0 Operations Metadata XSD, http://schemas.opengis.net/ows/1.0.0/owsOperationsMetadata.xsd
16.	OGC: OWS Common 1.1.0 Operations Metadata XSD, http://schemas.opengis.net/ows/1.1.0/owsOperationsMetadata.xsd
17.	OGC: OWS Common 2.0 Operations Metadata XSD, http://schemas.opengis.net/ows/2.0/owsOperationsMetadata.xsd
18.	OGC: WMS 1.1.1 DTD, http://schemas.opengis.net/wms/1.1.1/capabilities_1_1_1.dtd

19.	OGC: OGC Policy Directives, https://portal.opengeospatial.org/public_ogc/directives/directives.php
20.	Dierks, T. and Rescorla, E.: RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2, IETF, https://tools.ietf.org/html/rfc5246

2.2. Informative references

1.	Vretanos, P.: WFS 1.1.0, OGC 04-094, OpenGIS Web Feature Service (WFS) Implementation Specification
2.	Vretanos, P.: WFS 2.0, OGC 09-025r1, OpenGIS Web Feature Service 2.0 Interface Standard (also ISO 19142)
3.	Vretanos, P.: WFS 2.0.2, OGC 09-025r2, OGC® Web Feature Service 2.0 Interface Standard – With Corrigendum
4.	Baumann, P.: WCS 2.0, OGC 09-147r3, OGC® WCS Interface Standard - KVP Protocol Binding Extension, version 1.0.1
5.	Baumann, P.: WCS 2.0, OGC 09-148r1, OGC® WCS - XML/POST Protocol Binding Extension, version 1.0.0
6.	Baumann, P.: WCS 2.0, OGC 09-149r1, OGC® Web Coverage Service 2.0 Interface Standard - XML/SOAP Protocol Binding Extension, version 1.0.0
7.	Maso, J.: WMTS 1.0, OGC 07-057r7, OpenGIS Web Map Tile Service Implementation Standard
8.	Mueller, M.: WPS 2.0, OGC 14-065, OGC® WPS 2.0 Interface Standard
9.	Bröring, A.: SOS 2.0, OGC 12-006, OGC® Sensor Observation Service Interface Standard
10.	Simonis, I., Echterhoff, J.: SPS 2.0, OGC 09-000, OGC® Sensor Planning Service Implementation Standard
11.	Nebert D., Whiteside A., Vretanos, P.: CSW 2.0.2, OGC 07-006r1, OpenGIS Catalogue Service Implementation Specification
12.	Nebert D., Voges U., Vretanos P., Bigagli L., Westcott, B.: CSW 3.0, OGC 12-176r7, OGC® Catalogue Services 3.0 Specification - HTTP Protocol Binding
13.	Whiteside A.: OWS Common 1.1.0, OGC 06-121r3, OGC Web Services Common Specification, OGC® Implementation Standard
14.	Whiteside A., Greenwood J.: OWS Common 2.0.0, OGC 06-121r9, OGC Web Services Common Specification, OGC® Implementation Standard

2.3. Bibliography

See the Bibliography appendix.

Chapter 3. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r9], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

For the purposes of this document, the following additional terms and definitions apply:

- ADR XACML Authorization Decision Request
- CSW Catalogue Service for the Web
- DGIWG Defence Geospatial Information Working Group
- DTD Document Type Definition
- GMX ISO TC211 XML namespace http://www.isotc211.org/2005/gmx
- GeoXACML Geospatial eXtensible Access Control Markup Language
- HTTP Hypertext Transfer Protocol
- HTTPS Hypertext Transfer Protocol Secure
- IA Information Assurance
- IANA Internet Assigned Numbers Authority
- IETF Internet Engineering Task Force
- ISO International Organization for Standardization
- JSON JavaScript Object Notation
- MIME Multipurpose Internet Mail Extensions
- OASIS Organization for the Advancement of Structured Information
- OAuth OAuth
- OGC Open Geospatial Consortium
- OpenID Connect OpenID Connect
- OWS OGC Web Service
- PAP XACML Policy Administration Point
- RFC Request For Comments
- SAML Security Assertion Markup Language
- SDI Spatial Data Infrastructure
- SOS Sensor Observation Service
- SPS Sensor Planning Service
- SWG Standards Working Group
- URL Uniform Resource Locator
- URN Uniform Resource Name

- W3C World Wide Web Consortium
- WCS Web Coverage Service
- WFS Web Feature Service
- WMS Web Map Service
- WMTS Web Map Tile Service
- WPS Web Processing Service
- XACML eXtensible Access Control Markup Language
- XML eXtensible Markup Language

Chapter 4. Conventions

This section provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

All sections in this document are normative unless otherwise indicated.

4.1. Identifiers for this Standard

The normative provision of this standard is available by this URL

https://www.opengis.net/doc/IS/security/1.0

All Conformance Classes in this standard are identified by a URI with this base:

https://www.opengis.net/spec/security/1.0/conf/cc

All Conformance Tests in this standard are identified by a URI with this base:

https://www.opengis.net/spec/security/1.0/conf/cct

A URN identifier with this base identifies all Requirements Classes in this standard:

urn:ogc:specification:security:1.0:rc

Requirement Class URN identifiers are used in the <ows:Constraint> element to identify the nature of each constraint. These URNs serve as identifiers only and are not expected to be resolvable.

Requirements Classes in this standard are also identified by a URL with this base:

https://www.opengis.net/spec/security/1.0/req/rc

Requirement Class URL identifiers are used in the <ows:Meaning> element which is a child element of <ows:Constraint>. The purpose of this element is to provide a resource identifier for the Requirement Class.

All Requirements Class Tests in this standard are identified by a URI with this base:

https://www.opengis.net/spec/security/1.0/req/rct

A URL with this base identifies requirements for the service implementation:

https://www.opengis.net/spec/security/1.0/req/sr

A URL with this base identifies requirements for the client implementation:

https://www.opengis.net/spec/security/1.0/req/cr

A URL with this base identifies requirements for the OGC implementation:

https://www.opengis.net/spec/security/1.0/req/mr

All requirements are sequentially numbered as defined in Directive #43 [19.].

The following URI resolves to the Authentication Codelist as specified by this standard:

https://www.opengis.net/def/security/1.0/codelist/authentication

The following URI defines the namespace for the SecurityExtendedCapabilities element defined as WMS 1.1.1 DTD and WMS 1.3.0 schema element:

http://www.opengis.net/security/1.0

4.2. Versioning

The version of this standard can be maintained independent from the version of the Authentication Codelist, a Conformance or Requirements Class. Including the version in the standard URI as well as in the URNs ensures this.

4.3. Backwards Compatibility

This standard leverages the existing OWS Constraint element to enable the annotation of service capabilities with IA controls present on operation(s) of the service instance. This solution ensures backwards compatibility, as a Capabilities document that includes security annotations is valid against the existing OWS Common schema. For WMS 1.1.1 and WMS 1.3.0, which do not make use of OWS Common, a similar approach is standardized ensuring backwards compatibility.

All approaches ensure that a service endpoint can, independent from anything else, provide service Capabilities with security annotations. Client applications not capable of interpreting the annotations will simply ignore them but will not return expected results. Clients however, that properly interpret the security annotation can use that information to ensure interoperable functioning with secured OGC Web Services.

Chapter 5. Use Cases (informative)

NOTE	This standard does not require the use of a catalogue service to obtain the service metadata. This section includes a catalogue to fully illustrate the find-bind process for protected OGC Web Services.

The following use cases provide an overview on how to use annotated Capabilities. The term “annotated Capabilities” refers to the extension of a Capabilities document as defined by this standard. In the OGC world of services, the paradigm Publish-Find-Bind is based on a service / data provider describing the service with metadata, e.g. using ISO 19139 and registering that description. For the purpose of this standard, we assume that the metadata is stored in a catalogue service. From the catalogue service, the metadata for the service’s data and for the service itself can be found. For this specification, it is out of scope to describe the security annotations in ISO 19139 metadata. However, note that the service metadata, in the catalogue, contains a link to the service capabilities. This is usually the GetCapabilities operation of the actual service. For the use of annotated capabilities, this link must be freely accessible and therefore perhaps is a different URL (not the GetCapabilities operation of the actual secured service). Knowledge of the service “content” is restricted. Therefore public access to the annotated capabilities must be in compliance with the “need-to-know” principle. The following use cases illustrate the different combinations that might exist:

5.1. Use Case 0: Public Service / Public Data / Public Catalogue / Public Communication

This is the current standardized use of OGC Web Services – no security. Therefore, there are no implications for this standard.

Services can be discovered through a catalogue that has no security.

The client application can bind to the service instance via the Capabilities document. This reflects common practice for today’s SDIs.

5.2. Use Case I: Authenticated Public Service / Public Data / Public Catalogue / Secure Communication

This is another common use of OGC Web Services – no client authentication or authorization is used – only server authentication and a protected communication channel. The server authentication assures the client has the authentic source of the public data. The secure communication channel ensures privacy to the client. Outsiders cannot determine what data the client is receiving.

As server authentication methods are not currently specified in OGC Web Services, there are implications for this standard. In the most common case (HTTPS) server authentication and protected communications is not strictly compliant with existing OGC Web Services standards. This standard corrects that oversight.

The client application can bind to the service instance via the Capabilities document. This reflects

common practice for today's SDIs. The client needs to support the server authentication method while accessing the public data on the server, e.g. support for HTTPS.

5.3. Use Case II: Protected Service / Open Data / Public Catalogue / Secure Communication

For this use case, a public catalogue holds data and service metadata for a protected service. The public access to the catalogue implies that authentication is not required. It is therefore not possible to provide user specific responses. The user can discover service metadata that point to the public version of the annotated Capabilities.

Open data implies that the need-to-know principle does not apply. Therefore, the annotated Capabilities, accessible from a free and open URL, must contain all information relevant for the client application to bind to the service. In particular, this requires that the "content" section of the annotated capabilities list all accessible resources.

Any service that fits the description of use case II should include a "content" section in the annotated Capabilities that lists all accessible resources.

A client application that has implemented all the published security requirements is able to bind to the service and work with the Capabilities document in the usual fashion – likewise to use case 0.

The main difference with use case I is that both user and server are authenticated. Data is open but anonymous access is not allowed.

5.4. Use Case III: Protected Service / Private Data / Public Catalogue

This use case is similar to use case II with the exception that the service provides access to private data. This must trigger the need-to-know constraint, which implies that the "content" section of the annotated Capabilities is empty or other indications exist from which the client application can conclude that the service is providing private resources. One example is the Defence Geospatial Information Working Group (DGIWG) profile for the WFS Web Service that uses the <AccessContraints> element of the Capabilities to indicate the highest level of classification.

NOTE	The Capabilities document can have multiple <AccessContraints> elements.

In this case, the client application can only expect that the annotated Capabilities contain the <OperationsMetadata> element. Therefore, the client application is first required to determine its ability to comply with the security requirements outlined in the annotations. In cases where the application is interoperable with the security on the protected service instance, the client must execute the advertised GetCapabilities operation and comply with the security constraints to fetch the full Capabilities that include the "Content" section.

If the annotated Capabilities document does not contain a "content" section (<Layer> element for WMS or a <FeatureTypeList> element for a WFS or a <Contents> element for WMTS, WCS or WPS) or the <AccessContraints> element does not contain the literal "None", the application shall [3: This

will be stated as a requirement in the normative section of this document.] execute the GetCapabilities operation advertised (within the Capabilities document) to fetch the full service capabilities.

NOTE	The description above assumes that the client is able to function on the advertised security requirements for the GetCapabilities operation.

5.5. Use Case IV: Protected Service / Private Data / Protected Catalogue / Secure Communication

This use case assumes that managed access to the catalogue is in place. It also assumes that the user has sufficient credentials to access the catalogue. The catalogue would return sufficient information to enable the client to determine the protected access point and the required security credentials needed to obtain the service capabilities document.

NOTE	The credentials to access the catalogue might be different from the credentials to access the service.

5.6. Use Case V: Use of cached Capabilities instance documents

This use case does not build upon the existence of a catalogue service. It is assumed that a client loads the Capabilities instance document stored on a file system to configure access to a protected OGC Web Service. This approach is not recommended as outlined in section “Security Considerations” (see 10), because the integrity of such a document is not guaranteed. As a result, the Capabilities instance document might violate the need-to-know principle, as an arbitrary “Content” section could be included.

From a client application’s point of view, the entire content might have been tampered with. Therefore, it can only be safe to ignore these capabilities. The application should not even request a fresh copy of the Capabilities document from the endpoint contained in the Capabilities document as that endpoint could have also been changed to point to some malicious server.

5.7. Use Case VI: Use of Capabilities instance documents hosted on a Web Server

This use case does assume that a public Capabilities instance document exists that can be loaded by the application. In this case, the client application should fetch a trusted fresh copy from the endpoint of the protected service as outlined in the use the public version of the capabilities document.

Chapter 6. Conformance

Conformance with this standard shall be checked using the applicable requirements of each conformance and requirements class. Compliance can be verified via the normative set of conformance tests, defined in Annex A for the service, in Annex B for the client and Annex C for the OGC management process to maintain authentication codes and to operate an authentication code resolver to obtain their definitions. Section 7 defines the requirements for a service implementation to become compliant; section 8 defines the requirements for a client implementation and section 9 defines the requirements regarding the OGC management.

The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

All requirements and Requirements Classes described in this document are owned by the standard(s) identified.

NOTE	Special attention should be taken as this standard supersedes some requirements defined in OWS Common v1.0, v1.1.0 and v2.0 as well as existing OGC Web Service Implementation Specifications where applicable to ensure establishing of interoperable secured service instances.

The described approach to annotate Capabilities documents for secured service instances is backwards compatible as only existing OWS Common elements from the Capabilities document structure are leveraged. This backwards compatibility enforces a particular use of the element <ows:Constraint>, which is slightly different from the original intent.

In order to conform to this OGC standard, a software implementation shall be compliant to one of the three implementable Requirements Classes defined in the next sections.

The following figure provides a structural overview of the Requirements Classes defined in this standard to implement a complaint service.

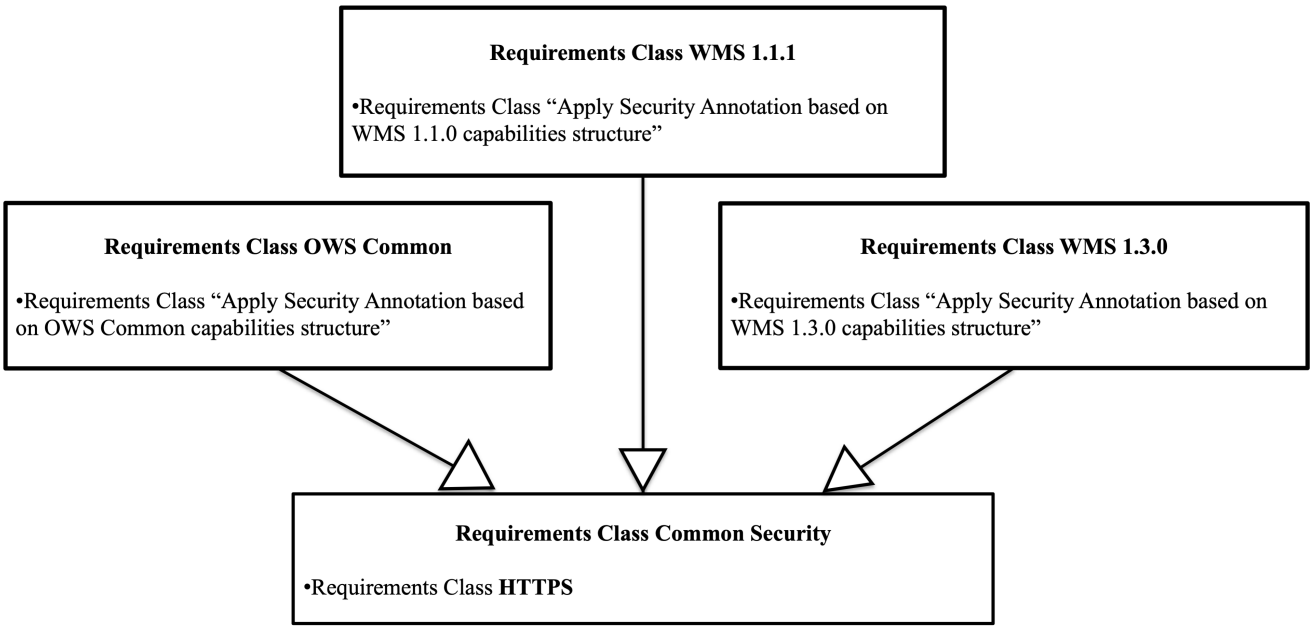

Figure 3. Requirements Classes of this standard

6.1. Requirements Class Common Security

Table 1. Requirements for the Requirements Class Common Security

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity**
Target type	Service Implementation
Dependency	None
Requirement	1
Requirement	2

Requirement 1	https://www.opengis.net/spec/security/1.0/req/sr/commonSecurity/1
	A service instance SHALL implement the following mandatory Requirements Class: https://www.opengis.net/spec/security/1.0/req/rc/https

Requirement 2	https://www.opengis.net/spec/security/1.0/req/sr/commonSecurity/2
	A service instance SHALL return a Capabilities response that includes the adequate <Content> section for the user if all implemented IAs are satisfied.

6.2. Requirements Class OWS Common

Table 2. Requirements for the Requirements Class OWS Common

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/owsCommon**

Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity**
Requirement	3
Requirement	4
Requirement	5
Requirement	6

Requirement 3	https://www.opengis.net/spec/security/1.0/req/sr/owsCommon/1
	This Requirements Class (**OWS Common**) SHALL be applied if the service endpoint provides security annotations in the Capabilities document and the capabilities are based on OWS Common Schema.

Requirement 4	https://www.opengis.net/spec/security/1.0/req/sr/owsCommon/2
	In order for a service instance operation to be compliant, requirements from the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity** SHALL be implemented.

Requirement 5	https://www.opengis.net/spec/security/1.0/req/sr/owsCommon/3
	For Capabilities based on OWS Common Schema, the service instance inherits the Capabilities structure from OWS Common (any version). The security annotation SHALL use the <ows:Constraint> element as defined in the <ows:ServiceMetadataType> definition.

Requirement 6	https://www.opengis.net/spec/security/1.0/req/sr/owsCommon/4
	The annotated Capabilities document SHALL be valid XML according to the underlying OWS Common schema.

6.3. Requirements Class WMS 1.1.1

Table 3. Requirements for the Requirements Class WMS 1.1.1

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/wms111**
Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity**
Requirement	7
Requirement	8
Requirement	9
Requirement	10
Requirement	11
Requirement	12

Requirement	13
Requirement	14

Requirement 7	https://www.opengis.net/spec/security/1.0/req/sr/wms111/1
	This Requirements Class (WMS 1.1.1) SHALL be applied if the service endpoint provides security annotations in the Capabilities document and the Capabilities structure is based on WMS 1.1.1 DTD.

Requirement 8	https://www.opengis.net/spec/security/1.0/req/sr/wms111/2
	In order for a service instance operation to be compliant, requirements from the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity** SHALL be implemented.

Requirement 9	https://www.opengis.net/spec/security/1.0/req/sr/wms111/3
	For Capabilities based on WMS 1.1.1, the DTD defined as ExtendedSecurityCapabilities SHALL be used to provide the security annotation. The normative definition can be obtained from this URL: http://schemas.opengis.net/wms/1.1.1/ExtendedSecurityCapabilities.dtd

Table 4. Definition of the ExtendedSecurityCapabilities element for WMS 1.1.1

```
<!DOCTYPE WMT_MS_Capabilities SYSTEM
"http://schemas.opengis.net/wms/1.1.1/WMS_MS_Capabilities.dtd"[
<!--
============================================================
OWS Common Security Extension to annotate security
Definition of element ows:OperationsMetadata replicating the
definition from the OWS Common Schema to become available as DTD
============================================================
-->
<!ELEMENT VendorSpecificCapabilities (ows_security:ExtendedSecurityCapabilities)>

<!ELEMENT ows_security:ExtendedSecurityCapabilities (ows:OperationsMetadata+)>
<!ATTLIST ows_security:ExtendedSecurityCapabilities xmlns:ows_security CDATA #FIXED
"http://www.opengis.net/security/1.0">

<!ELEMENT ows:OperationsMetadata (ows:Operation*)>
<!ATTLIST ows:OperationsMetadata xmlns:ows CDATA #FIXED
"http://www.opengis.net/ows/1.1">

<!ELEMENT ows:Operation (ows:DCP+) >
<!ATTLIST ows:Operation name CDATA #REQUIRED>

<!ELEMENT ows:DCP (ows:HTTP) >
<!ELEMENT ows:HTTP (ows:Get | ows:Post)+ >

<!ELEMENT ows:Get (ows:Constraint+)>
<!ATTLIST ows:Get xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink" xlink:type
CDATA #FIXED "simple" xlink:href CDATA #REQUIRED >

<!ELEMENT ows:Post (ows:Constraint+)>
<!ATTLIST ows:Post xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink" xlink:type
CDATA #FIXED "simple" xlink:href CDATA #REQUIRED >

<!ELEMENT ows:Constraint (ows:AllowedValues | ows:NoValues | ows:ValuesReference |
ows:Meaning)+>
<!ATTLIST ows:Constraint name CDATA #REQUIRED>

<!ELEMENT ows:AllowedValues (ows:Value+)>

<!ELEMENT ows:Value (#PCDATA)>

<!ELEMENT ows:NoValues EMPTY>

<!ELEMENT ows:ValuesReference (#PCDATA)>
<!ATTLIST ows:ValuesReference reference CDATA #REQUIRED>

<!ELEMENT ows:Meaning (#PCDATA)>
<!ATTLIST ows:Meaning reference CDATA #REQUIRED>
]>
```

Installing the <ows_security:ExtendedSecurityCapabilities> as a valid element into the Capabilities document, it must become a child to the element VendorSpecificCapabilities.

Requirement 10	https://www.opengis.net/spec/security/1.0/req/sr/wms111/4
	The <ows_security:ExtendedSecurityCapabilities> element SHALL be added as a LAST child to the <ows:VendorSpecificCapabilities> element.

Requirement 11	https://www.opengis.net/spec/security/1.0/req/sr/wms111/5
	If the actual capabilities document for the service instance does **not** include the element <ows:VendorSpecificCapabilities>, then the capabilities document SHALL use the DTD snippet as defined in Table 5 to enable security annotation.

Table 5. Normative use of ExtendedSecurityCapabilities element with WMS 1.1.1 Capabilities and no other vendor specific capabilities

```
<!--
============================================================
OWS Common Security Extension to annotate security requires

adding the element ows_security:ExtendedSecurityCapabilities to

your Vendor Specific Capabilities definition
If you do not define your own VendorSpecificCapabiltiies then

use this element
<!ELEMENT VendorSpecificCapabilities (ows_security:ExtendedSecurityCapabilities) >
============================================================
-->

<!ELEMENT VendorSpecificCapabilities (ows_security:ExtendedSecurityCapabilities) >
```

Requirement 12	https://www.opengis.net/spec/security/1.0/req/sr/wms111/6
	For Capabilities where the service instance **does** include <VendorSpecificCapabilities> then the <ows_security:ExtendedSecurityCapabilities> element SHALL be included into the vendor specific capabilities definition.

Informative Example: Assuming the following <VendorSpecificCapabilities> exists as illustrated in Table 6.

Table 6. Example definition of vendor specific capabilities for WMS 1.1.1 Capabilities

```
<!ELEMENT VendorSpecificCapabilities (Profiles) >
<!ELEMENT Profiles (Profile*) >
<!ELEMENT Profile (#PCDATA) >
```

Then, the security annotated version of the Capabilities should use the following DTD

Table 7. Use of ExtendedSecurityCapabilities element with WMS 1.1.1 Capabilities and vendor specific capabilities

```
<!--
============================================================
OWS Common Security Extension to annotate security
requires adding the element < ows_security:ExtendedSecurityCapabilities> to
your Vendor Specific Capabilities definition
If you do not define your own VendorSpecificCapabiltiies
then use this element
<!ELEMENT VendorSpecificCapabilities (<ows_security:ExtendedSecurityCapabilities>) >
============================================================
-->

<!ELEMENT VendorSpecificCapabilities (Profiles,
ows_security:ExtendedSecurityCapabilities) >
<!ELEMENT Profiles (Profile*) >
<!ELEMENT Profile (#PCDATA) >
```

Requirement 13	https://www.opengis.net/spec/security/1.0/req/sr/wms111/7
	The <ows:OperationsMetadata> elements of the <ows:ExtendedSecurityCapabilities> SHALL contain all operations metadata for the secured service endpoint. *Note: This might be a duplication of the operations metadata originally contained in the Capabilities document.*

Requirement 14	https://www.opengis.net/spec/security/1.0/req/sr/wms111/8
	The annotated Capabilities document SHALL be valid XML according to the underlying WMS 1.1.0 DTD.

6.4. Requirements Class WMS 1.3.0

Table 8. Requirements for the Requirements Class WMS 1.3.0

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/wms130**
Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity**
Requirement	**15**

Requirement	**16**
Requirement	**17**
Requirement	**18**

Requirement 15	https://www.opengis.net/spec/security/1.0/req/sr/wms130/1
	This Requirements Class (WMS 1.3.0) SHALL be applied if the service endpoint provides security annotations in the Capabilities document and the Capabilities structure is based on WMS 1.3.0 Capabilities Schema.

Requirement 16	https://www.opengis.net/spec/security/1.0/req/sr/wms130/2
	In order for a service instance operation to be compliant, requirements from the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity** SHALL be implemented.

Requirement 17	https://www.opengis.net/spec/security/1.0/req/sr/wms130/3
	For Capabilities based on WMS 1.3.0 Schema, the service instance SHALL use the schema element definition ExtendedSecurityCapabilities as a substitution for the _ExtendedCapabilities element when expressing the security annotation(s). The normative schema can be obtained from this URL: http://schemas.opengis.net/wms/1.3.0/ExtendedSecurityCapabilities.xsd

Table 9. Definition of the ExtendedSecurityCapabilities element for WMS 1.3.0

```
<schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ows_security="http://www.opengis.net/security/1.0"
  xmlns:ows="http://www.opengis.net/ows/1.1"
    xmlns:wms="http://www.opengis.net/wms"
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    targetNamespace="http://www.opengis.net/security/1.0"
    elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0.0">
    <import namespace="http://www.opengis.net/wms"
schemaLocation="http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd"/>
    <import namespace="http://www.opengis.net/ows/1.1"
schemaLocation="http://schemas.opengis.net/ows/1.1.0/owsOperationsMetadata.xsd"/>
    <xs:complexType name="ExtendedSecurityCapabilitiesType">
      <sequence>
        <element ref="ows:OperationsMetadata"/>
      </sequence>
    </xs:complexType>
    <element name="ExtendedSecurityCapabilities"
type="ows_security:ExtendedSecurityCapabilitiesType"
substitutionGroup="wms:_ExtendedCapabilities"/>
</schema>
```

Requirement 18	https://www.opengis.net/spec/security/1.0/req/sr/wms130/4
	The <ows:OperationsMetadata> elements of the <ows_security:ExtendedSecurityCapabilities> SHALL contain all operations metadata for the secured service endpoint.

Requirement 19	https://www.opengis.net/spec/security/1.0/req/sr/wms130/5
	The annotated Capabilities document SHALL be valid XML according to the underlying WMS 1.3.0 Schema. *Note: This might be a duplication of the operations metadata originally contained in the Capabilities document.*

Chapter 7. Conformance for a Service Implementation

The following requirements classes define what a service endpoint must implement to be compliant. Usually, this is two-fold: (i) What is the actual functionality that the service must provide when implementing an IA and (ii) How shall the annotation be done to indicate that the functionality is implemented.

7.1. Requirements Class HTTPS

Table 10. Requirements for Requirements Class HTTPS

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/https urn:ogc:specification:security:1.0:rc:https
Target type	Service Implementation
Dependency	
Requirement	20

Requirement 20	https://www.opengis.net/spec/security/1.0/req/sr/https/1
	Any OGC Web Service that is deployed (hosted) on HTTPS (HTTP over TLS) SHALL expose service Capabilities in which the URL protocol scheme for each operation is equal to the https URI scheme as defined in RFC 7230, section 2.7.2.

7.2. Requirements Class Identifiers

Table 11. Requirements for Requirements Class Identifiers

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/identifiers urn:ogc:specification:security:1.0:rc:identifiers
Target type	Service Implementation
Dependency	
Requirement	21
Requirement	22
Requirement	23

This standard uses URNs in the name attribute of the <ows:Constraint> element to identify the security control. The use of a URN here is sufficient, because they are used for comparison only.

The URIs are to be registered with the OGC Naming Authority. The OWS Common – Security Standards Working Group (SWG) triggers this action following an approved submission.

Requirement 21	https://www.opengis.net/spec/security/1.0/req/sr/identifiers/1
	Other URLs and URNs, not specified in this standard, SHALL be submitted via an OGC Change Request to the OWS Common – Security SWG for consideration.

Requirement 22	https://www.opengis.net/spec/security/1.0/req/sr/identifiers/2
	In the case where the implementer wishes to provide a resolvable URL of the URN, the <ows:Meaning> element and the ows:reference attribute SHALL be used for a WMS 1.1.1, WMS 1.3.0 as well as a OWS Common v1.1 or v2.0 based Capabilities structure to provide that resolvable URL.

Requirement 23	https://www.opengis.net/spec/security/1.0/req/sr/identifiers/3
	In the case where the implementer wishes to provide a resolvable URL of the URN, the <ows:Metadata> element and the href attribute SHALL be used for a OWS Common v1.0 based Capabilities structure to provide that resolvable URL.

7.3. Requirements Class HTTP Methods

Table 12. Requirements for Requirements Class HTTP Methods

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/http-methods urn:ogc:specification:security:1.0:rc:http-methods
Target type	Service Implementation
Dependency	
Requirement	24
Requirement	25

Requirement 24	https://www.opengis.net/spec/security/1.0/req/sr/http-methods/1
	The service endpoint SHALL list all supported HTTP methods (likely a subset of the methods defined in HTTP 1/1 recommendation from the IETF - RFC 2616).

Requirement 25	https://www.opengis.net/spec/security/1.0/req/sr/http-methods/2
	The URN identifier SHALL be used for the <ows:Constraint> element. For OWS Common Version 1.0 each supported HTTP method SHALL be put into a <Value> element. For OWS Common Version 1.1.0 or 2.0 the sub-element <ows:AllowedValues> SHALL be used to list each supported HTTP method using a <Value> element. The value of the element SHALL be in all uppercase the name of the method as identified in RFC 2616.

Table 13. Informative example indicating support for the methods GET, POST and OPTIONS for OWS

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:http-methods">
<ows:Value>GET</ows:Value>
<ows:Value>POST</ows:Value>
<ows:Value>OPTIONS</ows:Value>
</ows:Constraint>
```

Table 14. Informative example indicating support for the methods GET, POST and OPTIONS for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:http-methods">
<ows:AllowedValues>
<ows:Value>GET</ows:Value>
<ows:Value>POST</ows:Value>
<ows:Value>OPTIONS</ows:Value>
<ows:AllowedValues>
</ows:Constraint>
```

7.4. Requirements Class W3C CORS

Table 15. Requirements for Requirements Class W3C CORS

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/cors urn:ogc:specification:security:1.0:rc:cors
Target type	Service Implementation
Dependency	https://www.opengis.net/spec/security/1.0/req/rc/http-methods
Requirement	26

Requirement 26	https://www.opengis.net/spec/security/1.0/req/sr/cors/1
	The URN identifier urn:ogc:specification:security:1.0:rc:cors SHALL be used for the <Constraint> element to signal that the service endpoint operation is compliant with the W3C recommendation “Cross Origin Resource Sharing” (see normative references). The only valid sub-element SHALL be <ows:Value/> for OWS Common Version 1.0 and <ows:NoValues/> for OWS Common Version 1.1.0 or 2.0.

Table 16. Informative annotation for expressing compliance with W3C CORS for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:cors">
<ows:Value/>
</ows:Constraint>
```

Table 17. Informative annotation for expressing compliance with W3C CORS for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:cors">
<ows:NoValues/>
</ows:Constraint>
```

7.5. Requirements Class HTTP Exception Handling

Table 18. Requirements for Requirements Class HTTP Exception Handling

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/http-exception-handling urn:ogc:specification:security:1.0:rc:http-exception-handling
Target type	Service Implementation
Dependency	
Requirement	27
Requirement	28
Requirement	29

Requirement 27	https://www.opengis.net/spec/security/1.0/req/sr/http-exception-handling/1
	The URN identifier urn:ogc:specification:security:1.0:rc:http-exception-handling SHALL be used to identify the <ows:Constraint> element. The only valid sub-element SHALL be <ows:Value/> for OWS Common Version 1.0 and <ows:NoValues/> for OWS Common Version 1.1.0 or 2.0.

Table 19. Informative annotation for expressing compliance with HTTP Exception Handling for OWS Common 1.0

```
<ows:Constraint  name="urn:ogc:specification:security:1.0:rc:http-exception-handling">
<ows:Value/>
</ows:Constraint>
```

Table 20. Informative annotation for expressing compliance with HTTP Exception Handling for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:http-exception-handling">
<ows:NoValues/>
</ows:Constraint>
```

Different versions of OWS Common as well as various OGC Web Services standards define slightly different exception handling. The following figure illustrates the exception handling via ExceptionReport as defined in OWS Common 2.0 – OGC #06-121r9, section 8.6, table 28.

Table 21. OWS Common Exception Codes (source: OWS Common 1.2)

	HTTP Status Code	
exceptionCode value	**Code**	**Message**
OperationNotSupported	501	Not Implemented
MissingParameterValue	400	Bad request
InvalidParameterValue	400	Bad request
VersionNegotiationFailed	400	Bad request
InvalidUpdateSequence	400	Bad request
OptionNotSupported	501	Not Implemented
NoApplicableCode	3xx, 4xx, 5xx	Internal Server Error

OWS Common 2.0 differentiates between an exception that arises inside or outside the service implementation. In the case where the root cause of the error is inside an OGC service implementation, then the HTTP status code and ExceptionReport according to Figure 4 above shall be used. In the case where the root cause of the error is outside the OGC service implementation, then the HTTP status codes – with no ExceptionReport – is to be used. Unfortunately, OWS Common 1.1 – which is the mostly used version – does not differentiate the origin of the root cause. It is also unclear, which HTTP status code is to be used when delivering the ExceptionReport.

In addition, WMS introduces the EXCEPTION/EXCEPTIONS parameter that allows the client to control what the mime type of the exception returned is going to be. This allows including the ExceptionReport into an image.

Regardless of these variations to exception handling, this standard defines a clear separation between the actual OGC Web Services for which exception handling is defined, and the extra exception handling introduced by applying security to a service instance.

OGC web services do not exist in a vacuum. They are built on a set of services and standards, which define the underlying, distributed computing environment as illustrated in Figure 5 below.

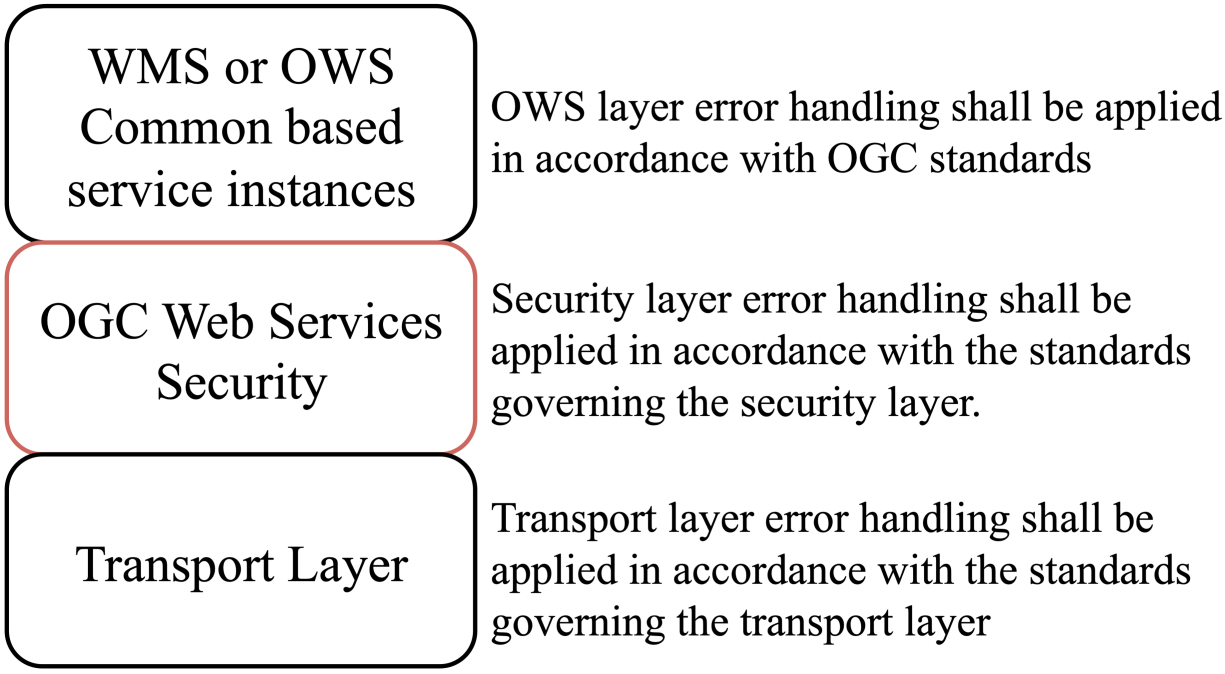

Figure 4. Processing Metaphor how to achieve interoperability with this standard

Not only do OGC Web Service need to correctly produce and handle exceptions defined in the OGC Standards but they also must correctly produce and handle exceptions in accordance with the standards for each service and protocol layer that the OGC Service is built upon. Therefore, an OGC Web Service which is compliant with this standard must comply with not just the exception handling requirements as defined in this standard, but also with the exception handling requirements of all of the supporting capabilities (protocols and services) which are advertised through the Capabilities document.

Requirement 28	https://www.opengis.net/spec/security/1.0/req/sr/http-exception-handling/2
	OWS services SHALL respect and implement exception handling for all supporting capabilities (protocols and services) identified through the Capabilities document in accordance with their governing standards.

Requirement 29	https://www.opengis.net/spec/security/1.0/req/sr/http-exception-handling/3
	A compliant implementation SHALL implement exception handling to use HTTP status codes as mandated by the relevant security standards if the origin is an implemented IA control.

7.6. Requirements Class HTTP POST Content-Type

Table 22. Requirements for Requirements Class HTTP POST Content-Type

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/content-type urn:ogc:specification:security:1.0:rc:content-type
Target type	Service Implementation
Dependency	
Requirement	30
Requirement	31
Requirement	32

Requirement 30	https://www.opengis.net/spec/security/1.0/req/sr/content-type/1
	The URN identifier urn:ogc:specification:security:1.0:rc:content-type SHALL be used to signal that the service endpoint operation is compliant with the Conformance Class “HTTP POST Content-Type”. The only valid sub-element SHALL be <ows:Value/> for OWS Common Version 1.0 and <ows:NoValues/> for OWS Common Version 1.1.0 or 2.0.

Table 23. Informative annotation expressing compliance with HTTP POST Content-Type for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:content-type">
<ows:Value>text/xml</ows:Value>
<ows:Value>application/xml</ows:Value>
<ows:Value application/soap+xml</ows:Value>
<ows:Value>application/x-www-form-urlencoded</ows:Value>
</ows:Constraint>
```

Table 24. Informative annotation expressing compliance with HTTP POST Content-Type for OWS Common 1.1.0 and 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:content-type">
<ows:AllowedValues>
<ows:Value>text/xml</ows:Value>
<ows:Value>application/xml</ows:Value>
<ows:Value application/soap+xml</ows:Value>
<ows:Value>application/x-www-form-urlencoded</ows:Value>
</ows:AllowedValues>
</ows:Constraint>
```

Requirement 31

	https://www.opengis.net/spec/security/1.0/req/sr/content-type/2
	If the service instance supports HTTP POST requests (as declared in the capabilities document), the service instance SHALL support the mime-type “application/x-www-url-form-encoding” as registered with IANA (https://www.iana.org/assignments/media-types/application/x-www-form-urlencoded).

Requirement 32	https://www.opengis.net/spec/security/1.0/req/sr/content-type/3
	The name of the POST parameter for the application/x-www-form-urlencoded mime-type shall be OWSR carrying the text/xml formatted request.

Note: For additional information see the OGC CR #388 (http://ogc.standardstracker.org/show_request.cgi?id=388)

7.7. Requirements Class Authorization

Table 25. Requirements for Requirements Class Authorization

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/authorization urn:ogc:specification:security:1.0:rc:authorization
Target type	Service Implementation
Dependency	
Requirement	33
Requirement	34
Requirement	35
Requirement	36

Requirement 33	https://www.opengis.net/spec/security/1.0/req/sr/authorization/1
	The URN identifier urn:ogc:specification:security:1.0:rc:authorization SHALL be used as the name of the <ows:Constraint> element to signal that the service endpoint operation is on access control. For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose.

Table 26. Informative annotation expressing compliance with Authorization for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authorization">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 27. Informative annotation expressing compliance with Authorization for OWS Common 1.1.0 and 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authorization">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

Requirement 34	https://www.opengis.net/spec/security/1.0/req/sr/authorization/2
	The URL provided with the urn:ogc:specification:security:1.0:rc:authorization constraint SHALL resolve to a XACML or GeoXACML policy.

Note: The URL provided might be protected and returns a user specific instance of the general access control policy.

Requirement 35	https://www.opengis.net/spec/security/1.0/req/sr/authorization/3
	For an XACML policy, the mime-type SHALL be used as registered with IANA and published informational by the IETF in RFC 7061 (https://tools.ietf.org/html/rfc7061): application/xacml+xml

Requirement 36	https://www.opengis.net/spec/security/1.0/req/sr/authorization/4
	For a GeoXACML policy, the mime-type SHALL be used as registered with IANA and published at https://www.iana.org/assignments/media-types/application/geoxacml+xml: application/geoxacml+xml

The implementation leveraging the URL to fetch the access policy must observe the Content-Type of the response to identify whether the policy is XACML or GeoXACML and which version.

_Note: It is not required that the access control layer at the service actually operates on a XACML or GeoXACML policy. However, to ensure interoperability and the ability of the client to determine the access denied case before executing a service, a standards compliant description is required. A proprietary language must not be used. An example where the client could leverage the obtained policy is before uploading tons of features to a WFS-T. In cases where the client has determined "deny" this would simply conserve bandwidth.

7.8. Requirements Class WS-Policy

Table 28. Requirements for Requirements Class WS-Policy

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/policy urn:ogc:specification:security:1.0:rc:policy
Target type	Service Implementation
Dependency	
Requirement	37

Requirement	38
Requirement	39

Requirement 37	https://www.opengis.net/spec/security/1.0/req/sr/ws-policy/1
	The URN identifier urn:ogc:specification:security:1.0:rc:policy SHALL be used as the name of the <ows:Constraint> element to signal that the service endpoint operation is on a WS-Security control. For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose. The URL provided SHALL resolve to the WS-Policy that defines the SOAP security conditions implemented.

Table 29. Informative annotation expressing compliance with WS-Policy for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:policy">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 30. Informative annotation expressing compliance with WS-Policy for OWS Common 1.1.0 and 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:policy">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

Requirement 38	https://www.opengis.net/spec/security/1.0/req/sr/ws-policy/2
	The service endpoint shall use the annotation to provide insight information about the SOAP + WS-Security based security. The URL SHALL resolve to the WS-Policy used to describe the WS-Security conditions.

Requirement 39	https://www.opengis.net/spec/security/1.0/req/sr/ws-policy/3
	For returning a WS-Policy, the official WS-Policy mime-type SHALL be used as registered with IANA and published by the W3C in the Web Services Policy 1.5 Framework (https://www.w3.org/TR/2006/WD-ws-policy-20061117/): application/wspolicy+xml

7.9. Requirements Class OpenAPI

Table 31. Requirements for Requirements Class OpenAPI

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/openapi urn:ogc:specification:security:1.0:rc:openapi
Target type	Service Implementation
Dependency	
Requirement	40

Requirement 40	https://www.opengis.net/spec/security/1.0/req/sr/openapi/1
	The identifier urn:ogc:specification:security:1.0:rc:openapi SHALL be used to inform that there is an OpenAPI compliant description of the service endpoint(s). For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose. The provided URL SHALL resolve to a valid OpenAPI instance document.

Note: A referenced description may leverage OpenAPI extensions.

Table 32. Informative annotation expressing compliance with OpenAPI for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:openapi">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 33. Informative annotation expressing compliance with OpenAPI for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:openapi">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

The use of OpenAPI for describing an API can also be used to describe the communication protocols used by an interface of an OGC Web Service. We recognize that this description is **not** a replacement for the Capabilities document.

However, the main driver for using the OpenAPI format is to provide security constraints for the service instance using a format that is well known in mainstream IT. Particularly important is the ability to provide information about existing security controls. The example below illustrates how to provide additional (meta) information for an OAuth2 protected service instance [4: https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md#implicit-oauth2-sample].

Table 34. Example annotation expressing OAuth2 requirement with OpenAPI

type: oauth2

flows:

implicit:

authorizationUrl: https://example.com/api/oauth/dialog

scopes:

write:pets: modify pets in your account

read:pets: read your pets

authorizationCode:

authorizationUrl: https://example.com/api/oauth/dialog

tokenUrl: https://example.com/api/oauth/token

scopes:

write:pets: modify pets in your account

read:pets: read your pets

This example from the OpenAPI v3.0 specification [5: https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md] indicates that the service endpoint requires OAuth2 Access Token released for particular scopes.

7.10. Requirements Class Authentication

Table 35. Requirements for Requirements Class Authentication

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/authentication urn:ogc:specification:security:1.0:rc:authentication
Target type	Service Implementation
Dependency	
Requirement	41
Requirement	42

Requirement 41	

	https://www.opengis.net/spec/security/1.0/req/sr/authentication/1
	The URN identifier urn:ogc:specification:security:1.0:rc:authentication SHALL be used for the name of the <ows:Constraint> element to signal that the service endpoint operation requires authentication. For OWS Common 1.1.0 and 2.0 the sub-element SHALL be <ows:ValuesReference> where the reference attribute value contains the URN referencing the authentication code as identified in the Authentication Codelist.

The following informative example illustrates the security annotation to indicate that the authentication method client side TLS certificate is in place.

Table 36. Informative example annotation expressing client authentication via certificate for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication">
<ows:Value>
urn:ogc:def:security:authentication:ietf:5246:client_certificate
</ows:Value>
<ows:Metadata
href="https://www.opengis.net/def/security/1.0/codelist/authentication/TLS_CLIENT_CERT
IFICATE"/>
</ows:Constraint>
```

Table 37. Informative example annotation expressing client authentication via certificate for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication">
<ows:ValuesReference ows:reference=
"urn:ogc:def:security:authentication:ietf:5246:client_certificate"/>
<ows:Meaning ows:reference=
"https://www.opengis.net/def/security/1.0/codelist/authentication/TLS_CLIENT_CERTIFICA
TE"/>
</ows:Constraint>
```

Requirement 42	https://www.opengis.net/spec/security/1.0/req/sr/authentication/2
	In the case where the implementer wishes to provide a resolvable URL to the definition of the authentication method, for OWS Common 1.1.0 and 2.0 the <ows:Meaning> element and the reference attribute SHALL be used to provide that resolvable URL. The URL SHALL fetch the definition from the Authentication CodeList that corresponds to the name attribute used with the <ows:Constraint> element as defined in Requirement 68

7.11. Requirements Class SAML2

Table 38. Requirements for Requirements Class SAML2

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/authentication/saml2 urn:ogc:specification:security:1.0:rc:authentication:saml2
Target type	Service Implementation
Dependency	
Requirement	43
Requirement	44
Requirement	45

Requirement 43	https://www.opengis.net/spec/security/1.0/req/sr/saml2/1
	The URN identifier urn:ogc:specification:security:1.0:rc:authentication:saml2 SHALL be used to provide additional information if required by the identified authentication method. For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose. The provided URL SHALL resolve to the SAML2 metadata file for the federation of which this service endpoint is a member off.

Table 39. Informative annotation expressing compliance with SAML2 for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:saml2">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 40. Informative annotation expressing compliance with SAML2 for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:saml2">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

In addition to annotating the authentication method as defined in above, additional information can be provided. For example, the service provider might want to let the client know to which SAML2 federation the service belong. This could be achieved by using the identifier urn:ogc:specification:security:1.0:rc:authentication:saml2

Requirement 44

	https://www.opengis.net/spec/security/1.0/req/sr/saml2/2
	The <ows:Constraint> element SHALL have the identifier urn:ogc:specification:security:1.0:rc:authentication:saml2 to indicate additional SAML2 metadata information is available.

Requirement 45	https://www.opengis.net/spec/security/1.0/req/sr/saml2/3
	The <ows:ValuesReference> element and the reference attribute SHALL have the value of the URL which allows to fetch the SAML2 compliant metadata for the federation in which the service is participating in.

NOTE	Before starting the authentication handshake, the client should check if the advertised SAML authentication method (see section above) is supported.

7.12. Requirements Class OpenID Connect

Table 41. Requirements for Requirements Class OpenID Connect

Requirements Class	https://www.opengis.net/spec/security/1.0/req/rc/authentication/oidc urn:ogc:specification:security:1.0:rc:authentication:oidc
Target type	Service Implementation
Dependency	
Requirement	46
Requirement	47

Requirement 46	https://www.opengis.net/spec/security/1.0/req/sr/oidc/1
	The URN identifier urn:ogc:specification:security:1.0:rc:authentication:oidc SHALL be used to provide additional information if required by the identified authentication method. For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose. The provided URL SHALL resolve to the well-known description of the relevant OAuth2 Authorization Server implementing the OpenID Connect extension.

Table 42. Informative annotation expressing compliance with OpenID Connect for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:oidc">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 43. Informative annotation expressing compliance with OpenID Connect for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:oidc">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

Similar to SAML2, the client does need additional information to start the authentication handshake. For OpenID Connect, this is the metadata of the accepted Authorization Server linked with the secured service endpoint (OpenID Connect Discovery).

For an Authorization Server that is a compliant OpenID Connect implementation, a .well-known description exists as defined by IANA (URL ends with ".well-known/openid-configuration").

Requirement 47	https://www.opengis.net/spec/security/1.0/req/sr/oidc/2
	The <ows:Contraint> element SHALL have the identifier urn:ogc:specification:security:1.0:rc:authentication:oidc to indicate that additional OpenID Connect metadata information is available. For OWS Common 1.0 the href attribute of the <ows:Metadata> sub-element SHALL hold the URL. For OWS Common 1.1.0 and 2.0 the reference attribute of the <ows:ValuesReference> sub-element SHALL serve this purpose. The provided URL SHALL resolve to the .well-known configuration of the OpenID Connect Provider associated with the protected service endpoint.

Table 44. Informative annotation expressing OpenID Connect .well-known URL for OWS Common 1.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:oidc">
<ows:Value/>
<ows:Metadata href="{URL}"/>
</ows:Constraint>
```

Table 45. Informative annotation expressing OpenID Connect .well-known URL for OWS Common 1.1.0 or 2.0

```
<ows:Constraint name="urn:ogc:specification:security:1.0:rc:authentication:oidc">
<ows:ValuesReference ows:reference="{URL}"/>
</ows:Constraint>
```

Chapter 8. Conformance for a Client Implementation

In order for the concept of Capabilities with security annotations to work, these annotated Capabilities must be available to the client application with no security challenges involved. The methods describing how to make the annotated Capabilities for a service instance available to the client application vary and depend on many factors. Methods describing how the annotated Capabilities for the service are actually made available to the client application are outside the scope of this standard.

The following steps outline the general approach what a client implementation must do with the annotated Capabilities document:

- The client implementation must load the annotated Capabilities. The details how this happens are outside the scope of this standard. But, one typical approach would be that a Capabilities instance document could be downloaded from a public Web Server. Or, the client implementation could load a Capabilities instance document from the file system. Please observe the security considerations as outlined in section 10 of this document.
- The Client implementation should interpret the annotated capabilities by parsing the operations metadata to determine its compatibility with the outlined security controls and features of the service.
- If the “content” section of the annotated Capabilities document is empty, the client should execute the GetCapabilities operation as published in the annotated Capabilities to get the full Capabilities document and in particular the content section.
- If the “content” section is not empty, the client may proceed as usual by calling the service specific operations, e.g. GetMap, GetFeature, etc. The client may call the GetCapabilities operation of the service as outlined in the annotated Capabilities document.

8.1. Client Requirements Classes

A client can implement support for one or more Requirements Classes as defined by this standard:

- https://www.opengis.net/spec/security/1.0/req/rc/owsCommon,
- https://www.opengis.net/spec/security/1.0/req/rc/wms130 or
- https://www.opengis.net/spec/security/1.0/req/rc/wms111.

Each of these Requirements Classes define the parsing of security annotations by obeying the different structures of the Capabilities document.

NOTE	All of these Requirements Classes above have a mandatory dependency to the Requirements Class **Common Security** that implies to implementation of HTTPS.

8.1.1. Requirements Class Client Common Security

Table 46. Requirements for the Requirements Class Client Common Security

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientCommonSecurity**
Target type	Client Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/commonSecurity**
Requirement	48

According to Requirement 1 any service instance must operate on HTTPS. Therefore, any client implementation claiming conformance has to support HTTPS.

Requirement 48	https://www.opengis.net/spec/security/1.0/req/cr/commonSecurity/1
	Any compliant client implementation SHALL support HTTP over TLS as defined by RFC 2818. This includes certificate validation, verification and use of Certificate Revocation Lists.

8.1.2. Requirements Class Client OWS Common

Table 47. Requirements for the Requirements Class Client OWS Common

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientOWSCommon**
Target type	Client Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/clientCommonSecurity**
Requirement	49

Requirement 49	https://www.opengis.net/spec/security/1.0/req/cr/clientOWSCommon/1
	A client implementation SHALL be able to parse the security annotations produced by a service implementation compliant to the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/clientOWSCommon**

8.1.3. Requirements Class Client WMS 1.3.0

Table 48. Requirements for the Requirements Class Client WMS 1.3.0

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientWMS130**
Target type	Client Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/clientCommonSecurity**
Requirement	50

Requirement 50

	https://www.opengis.net/spec/security/1.0/req/cr/clientWMS130/1
	A client implementation SHALL be able to parse the security annotations produced by a service implementation compliant to the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/wms130**

8.1.4. Requirements Class Client WMS 1.1.1

Table 49. Requirements for the Requirements Class Client WMS 1.1.1

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientWMS111**
Target type	Client Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/clientCommonSecurity**
Requirement	51

Requirement 51	https://www.opengis.net/spec/security/1.0/req/cr/clientWMS111/1
	A client implementation SHALL be able to parse the security annotations produced by a service implementation compliant to the Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/wms111**.

8.2. Requirements Class Client Parsing

Table 50. Requirements for the Requirements Class Client Parsing

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientParsing**
Target type	Client Implementation
Dependency	
Requirement	52
Requirement	53
Requirement	54
Requirement	55

A service is already compliant to this standard and has implemented the mandatory Requirements Class HTTPS when all operation endpoints of the service exposed in the Capabilities document are using the URL scheme https. This undoubtedly means that the service is hosted on HTTPS.

Requirement 52	https://www.opengis.net/spec/security/1.0/req/cr/clientParsing/1
	The client SHALL accept a service Capabilities where operation endpoints use the URL protocol scheme https.

Requirement 53

https://www.opengis.net/spec/security/1.0/req/cr/clientParsing/2
The client SHALL parse the Capabilities document for <ows:Constraint> element(s) to find all Requirements Classes implemented by the service. The parsing SHALL acknowledge the structure of the Capabilities that can be determined by the XML namespace and the name of the root element.

8.2.1. Working with Complete Capabilities

According to Requirement 2 the Capabilities document returned by the service operation GetCapabilities, as outlined in the annotated capabilities document, returns a full capabilities document. Therefore, a client can be sure to work on a full capabilities document only in this case. In the case where security controls are implemented for the GetCapabilities operation their existence is indicated by the relevant security annotations.

Note: In case that security controls are indicated for the GetCapabilities operation (thru <ows:Constraint> elements) the client must overcome the security controls to receive the full capabilities document.

8.2.2. Working with Partial Capabilities

As described in section 5.4 (Use Case III: Protected Service / Private Data / Public Catalogue) it is possible that publicly accessible capabilities include none or a partial “content” section. In these cases, the client must execute the GetCapabilities operation as outlined in the publically accessible version of the capabilities to fetch the full capabilities document. The client can determine partial capabilities by parsing for the absence of the “content” section.

Table 51. Section names and their content (source: OGC #06-121r9, p.25)

Section name	Contents
ServiceIdentification	Metadata about this specific server. The contents and organization of this section should be the same for all OWSs.
ServiceProvider	Metadata about the organization operating this server. The contents and organization of this section should be the same for all OWSs.
OperationsMetadata	Metadata about the operations specified by this service and implemented by this server, including the URLs for operation requests. The basic contents and organization of this section shall be the same for all OWSs, but individual services may add elements and/or change the optionality of optional elements.

Section name	Contents
Contents	Metadata about the data served by this server. The contents and organization of this section are specific to each OWS type, as defined by that Implementation Specification. Whenever applicable, this section shall contain a set of dataset descriptions, which should each be based on the MD_DataIdentification class specified in ISO 19115 and used in ISO 19119.
Languages	Languages supported by this server. The contents and organization of this section shall be the same for all OWSs.

The “content” section of the Capabilities is represented by different XML elements for different OGC Web Service types:

1. WMS: <Layer>
2. WMTS: <Contents>
3. WCS: <Contents>
4. WFS: <FeatureTypeList>

In any case, for the annotated capabilities to be present, the Capabilities instance document must at least contain the <ows:OperationsMetadata> element and the mandatory operation GetCapabilities. As illustrated in the use cases in section 2, the “content” part of the capabilities document might be omitted. But how can a client tell that a content section is just partially complete? Based on the current standard, it is not possible for the client to determine whether the content section is just partial. This results in two client side requirements.

Requirement 54	https://www.opengis.net/spec/security/1.0/req/cr/clientParsing/3
	If the annotated Capabilities document does not contain a “content” section (<Layer> element for WMS or a <FeatureTypeList> element for a WFS or a <Contents> element for WMTS, WCS or WPS), the client implementation SHALL execute the GetCapabilities operation advertised (within the Capabilities document) to fetch the full service capabilities. *Note: This assumes that the client implementation is able to function on the advertised security requirements for the GetCapabilities operation.*

Requirement 55
https://www.opengis.net/spec/security/1.0/req/cr/clientParsing/4
If the “content” section of the annotated capabilities document is not empty, the client implementation SHALL call the GetCapabilities operation of the service to ensure the advertised content is complete.

Requirement 55	

	https://www.opengis.net/spec/security/1.0/req/cr/clientParsing/4
	If the “content” section of the annotated capabilities document is not empty, the client implementation SHALL call the GetCapabilities operation of the service to ensure the advertised content is complete.

8.3. Requirements Class Client Exception Handling

Table 52. Requirements for the Requirements Class Client Exception Handling

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/clientExceptionHandling**
Target type	Client Implementation
Dependency	
Requirement	56
Requirement	57
Requirement	58
Requirement	59

According to Requirements Class https://www.opengis.net/spec/security/1.0/req/rc/http-exception-handling

a service instance may advertise its support for HTTP compliant exception handling. According to Requirement <fix me> the default exception handling for a service instance compliant to this standard has to use HTTP status codes. But a client can request a service to react with OWS Common based exception handling by submitting a query parameter as specified by the OWS Common standard applicable to the service instance.

Requirement 56	https://www.opengis.net/spec/security/1.0/req/cr/exceptionHandlingProcessing/1
	The client SHALL expect exception handling compliant to HTTP of a service instance that returns the constraint with identifier urn:ogc:specification:security:1.0:rc:http-exception-handling.

Requirement 57	https://www.opengis.net/spec/security/1.0/req/cr/exceptionHandlingProcessing/2
	For the Requirements Class OWS Common, the client SHALL issue the request to the service to send error responses according to the OWS specification as defined by the underlying OWS Common specification.

Requirement 58	https://www.opengis.net/spec/security/1.0/req/cr/exceptionHandlingProcessing/3
	For the Requirements Class WMS1.1.1, the client SHALL use the KVP Exception as standardized to request from the service to send error responses according to the WMS 1.1.1 specification.

Requirement 59	https://www.opengis.net/spec/security/1.0/req/cr/exceptionHandlingProcessing/4
	For the Requirements Class WMS 1.3.0, the client SHALL use the KVP Exceptions as standardized to request from the service to send error responses according to the WMS 1.3.0 specification.

Chapter 9. OGC Conformance

Table 53. Requirements for the OGC Requirements Class Authentication Codelist

Requirements Class	**https://www.opengis.net/spec/security/1.0/req/rc/authenticationCodelist**
Target type	OGC Naming Authority
Dependency	
Requirement	60
Requirement	61
Requirement	62
Requirement	63
Requirement	64
Requirement	65
Requirement	66
Requirement	67
Requirement	68
Requirement	69
Requirement	70

This Requirements Class describes requirements for the OGC to establish a management process regarding the Authentication Codelist mandated by this standard. OGC is responsible for implementing compliance.

Requirement 60	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/1
	OGC SHALL establish a process to maintain and resolve authentication codes as mandated by this Requirements Class.

For ensuring interoperability with authentication methods implemented on a service operation, this standard defines an Authentication Codelist as a normative reference to identify authentication codes. The Authentication Codelist shall be hosted by the OGC. The maintainer of the codes and values of the Authentication Codelist is the OWS Common – Security SWG.

Regarding the interoperability between secured OGC Web Services and client applications, the most important and critical topic is Authentication. The concept of annotated Capabilities allows authentication methods to be declared based on an Authentication Codelist, maintained by the OGC.

Requirement 61	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/2
	The Authentication Codelist is an ISO 19119 service metadata compliant Authentication Codelist using the GMX namespace as it is defined in ISO 19139:2007.

Using this namespace and schema structure for the authentication codes ensures that the same

authentication codes can be used to annotate service metadata to specify the existence of the IA Control Authentication. In technical terms, this can be achieved to load the Authentication Codelist as an external code list into the metadata document.

Essentially, the Authentication Codelist defined in this standard contains identifiers in different name spaces that can be used in the security annotation for authentication, a human readable description and a link to the standard that defines authentication code. The concept of name spaces is important as it enable the re-use of already defined authentication methods and protocols. For example, HTTP BASIC/DIGEST authentication is defined in the namespace IETF, as defined in RFC 2617. Likewise OAuth Bearer authentication is defined in the IETF namespace and SAML protocols are defined in the OASIS namespace. In case where vendor specific authentication is used, they should be included into the Authentication Codelist and the namespace would indicate that the owner is the 3rd party.

Requirement 62	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/3
	Authentication codes SHALL use the namespace that reflects the maintainer of the authentication code.

The maintainer of the Authentication Codelist is the OWS Common - Security SWG. The approval of new authentication codes must be submitted to this SWG via an OGC Change Request: http://ogc.standardstracker.org/.

Requirement 63	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelis/4
	New authentication codes for the Authentication Codelist hosted by OGC SHALL be submitted via Change Request to the OWS Common Security SWG.

Requirement 64	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/5
	To ensure backwards compatibility of the Authentication Codelist, only new authentication codes SHALL be added to the Authentication Codelist. It is not possible to modify or remove existing codes.

9.1. Authentication Codelist Hosting

A normative version of the Authentication Codelist is hosted and maintained by the OGC.

Requirement 65	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/6
	The OGC SHALL host the Authentication Codelist.

Requirement 66

	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/7
	Hosting URL of the Authentication Codelist SHALL use URL protocol scheme https.

9.2. Initial Authentication Codelist

Requirement 67	https://www.opengis.net/spec/security/1.0/req/mr/authenicationCodelist/8
	The Authentication Codelist from Annex D SHALL be used by OGC to setup the version 1.0 of the Authentication Code List Registry.

Requirement 68	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/9
	The official Authentication Codelist SHALL be resolvable via the following URI: https://www.opengis.net/def/security/1.0/codelist/authentication

9.3. Authentication Codes

For ensuring interoperability with authentication methods implemented on a service instance, this standard defines URNs in an Authentication Codelist. The Authentication Codelist uses the ISO GMX namespace to enable interoperable use within the security annotations of the service capabilities as well as service ISO metadata.

Note: How to use the authentication codes to annotate ISO metadata is out of scope for this standard.

9.3.1. Authentication Codes defined by IETF

Based on the IANA Hypertext Transfer Protocol (HTTP) Authentication Scheme Registry (http://www.iana.org/assignments/http-authschemes/http-authschemes.xhtml) the following HTTP authentication methods are defined based on IETF RFCs:

Table 54. IETF defined authentication methods

Identifier	Namespace	Reference
Basic	IETF	http://www.iana.org/go/rfc7617
Bearer	IETF	http://www.iana.org/go/rfc6750
Digest	IETF	http://www.iana.org/go/rfc7616

As these identifiers are not URNs, this standard defines them in the OGC namespace.

IETF Identifier	OGC Identifier	Namespace
Basic	urn:ogc:def:security:authentication:ietf:2617:Basic	OGC
Bearer	urn:ogc:def:security:authentication:ietf:6750:Bearer	OGC

IETF Identifier	OGC Identifier	Namespace
Digest	urn:ogc:def:security:authentication:ietf:2617:Digest	OGC

9.3.2. Authentication Codes defined by OASIS

Based on OASIS SAML2 Authentication Context definitions, the following authentication URNs are defined in the OASIS namespace.

Table 55. List of Authentication methods defined for SAML2

Identifier	Namespace
urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocol	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocolPassword	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:Kerberos	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:MobileOneFactorUnregistered	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorUnregistered	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:MobileOneFactorContract	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorContract	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:Password	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:X509	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:PGP	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:SPKI	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:XMLDSig	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:Smartcard	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:SmartcardPKI	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:SoftwarePKI	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:Telephony	OASIS

Identifier	Namespace
urn:oasis:names:tc:SAML:2.0:ac:classes:NomadTelephony	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:PersonalTelephony	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:AuthenticatedTelephony	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:SecureRemotePassword	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:TimeSyncToken	OASIS
urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified	OASIS

9.3.3. Authentication Codes defined by OGC

Identifier **urn:ogc:def:security:authentication:ietf:5246:client_certificate** has the meaning equivalent to that defined in RFC 5246: “This type of authentication is an extension to the TLS handshake as outlined in section 7.4.4: "A non-anonymous server can optionally request a certificate from the client, if appropriate for the selected cipher suite. This message, if sent, will immediately follow the ServerKeyExchange message (if it is sent; otherwise, this message follows the server’s Certificate message)."[RFC 5246] In case the client cannot provide a suitable and valid certificate, no TLS connection gets established”[RFC 5246]

Table 56. OGC identifier for IETF authentication method

Identifier	Namespace
urn:ogc:def:security:authentication:ietf:5246:client_certificate	OGC

9.4. Requirements Class “Authentication Codelist Registry”

The definition of the authentication code can be resolved from the Authentication Codelist URI via the pattern defined in Requirement 68. The main purpose of the Authentication Codelist Registry is to enable the lookup of authentication code definitions via URL resolving.

```
<gmx:codelistItem>
  <gmx:CodeListDictionary gml:id="AuthenticationCode">
    <gml:description>identification of authentication methods</gml:description>
    <gml:identifier codeSpace="OGC">urn:ogc:def:security:1.0:authentication</gml:identifier>
    <gmx:codeEntry> [15 lines]
    <gmx:codeEntry> [16 lines]
    <gmx:codeEntry> [11 lines]
    <gmx:codeEntry> [13 lines]
    <gmx:codeEntry>
      <gmx:CodeDefinition gml:id="OAUTH2_BEARER_TOKEN">
       <gml:description>
         In the scenario supported by the OAuth 2.0 SSO profile, a web user or service either accesses a resource
          at a service provider, or accesses an identity provider such that the service provider and desired resource are understoo
          or implicit. The web user authenticates (or has already authenticated) to the identity provider, Which then produces an
          authorization grant which was then used by an authorization service to return an access token.  This access token then
          subtitutes as both authentication and authorization on future requests.</gml:description>
        <gml:identifier codeSpace="IETF">urn:ogc:def:security:authentication:ietf:6750:Bearer</gml:identifier>
      </gmx:CodeDefinition>
    </gmx:codeEntry>
```

Figure 5. Informative example of an authentication code entry

As defined in section 7 of this standard, the <ows:Constraint> element uses the <gml:identifier> of the authentication code (urn:ogc:def:security:authentication:ietf:6750:Bearer in the example) to identify the authentication code. For a WMS 1.1.1, WMS 1.3.0 as well as OWS Common v1.1 or v2.0 based Capabilities structure, the <ows:Meaning> allows to fetch the <gml:description> element of the <codeEntry> by specifying the gml:id attribute of the <gmx:CodeDefinition> element.

Requirement 69	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/10
	For the URL pointing to the <gmx:CodeDefinition>, the resolver SHALL return a human readable definition of the associated authentication code in mime-type text/html.

Requirement 70	https://www.opengis.net/spec/security/1.0/req/mr/authenticationCodelist/11
	The base URL for resolving authentication codes SHALL be https://www.opengis.net/def/security/1.0/codelist/authentication

The recommended URL for resolving an authentication code is to extend the base URI with https://www.opengis.net/def/security/1.0/codelist/authentication/\{gml:id}. For example: https://www.opengis.net/def/security/1.0/codelist/authentication/OAUTH2_BEARER_TOKEN

Chapter 10. Security Considerations (informative)

Applying this standard to a service endpoint provides the opportunity to expose security metadata into the service Capabilities. The main purpose is to provide an interoperability mechanism such that the client can determine whether the security controls at the service are supported.

10.1. Threat “Tampered Capabilities”

The mechanism of including security metadata into the Capabilities works well if the client could trust the Capabilities. For the purpose of the security considerations, it is best to differentiate if the Capabilities are used as an XML instance document or as the direct response from the service to the GetCapabilities request.

Would this threat lead to vulnerability? Yes, this threat could cause a client to wrongly submit user credentials to a malicious site!

Assuming that the attacker would be able to modify the URL of the service endpoint and assert that the authentication method were HTTP BASIC Authentication (as an example). This would cause the client to submit user credentials with the service request. This vulnerability must be considered high risk, as the client has no means to identify the attack.

10.1.1. Mitigations to this threat:

HTTPS is mandatory for any service instance that is compliant to this standard, it is mandatory to have HTTPS in place. However, the Capabilities document being an XML instance document must not be trusted, as it has no means of protection applied.

10.1.2. Approaches to provide a digital signature to the Capabilities document

The W3C XML Digital Signature is a method to provide integrity to an XML instance document. Applying a digital signature can take place using three methods:

1. Enveloping Signature
2. Enveloped Signature
3. Detached Signature

One of the main objectives to this OGC Web Services Security standard was is to ensure backwards compatibility which leads to the standardized approach: Insert security metadata into existing elements of the Capabilities structure and publish the <ows:Constraint> element within the structure for the service metadata. In order to ensure backwards compatibility for the digital signature as well would only allow using the method enveloped signature. However, the Digital Signature element could not be in the usual place (either first or last element of document root) but would rather have to sit inside the <ExtendedCapabilities> element. Even though putting the digital signature element inside the <ExtendedCapabilities> element is compliant with the W3C Digital

Signature recommendation, main stream IT tools would fail, as they look for the signature in the usual / recommended place.

See the Signature element on Line 20 below.

Table 57. Example ExtendedCapabilities element including a Signature element

```
<schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows_security="http://www.opengis.net/security/1.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:wms="http://www.opengis.net/wms"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xlink="http://www.w3.org/1999/xlink"
targetNamespace="http://www.opengis.net/security/1.0"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
version="1.0.0">
<import namespace="http://www.opengis.net/wms"
schemaLocation="http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd"/>
<import namespace="http://www.opengis.net/ows/1.1"
schemaLocation="http://schemas.opengis.net/ows/1.1.0/owsOperationsMetadata.xsd"/>
<import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
<xs:complexType name="ExtendedXMLDSigCapabilitiesType">
<sequence>
<element ref="ds:Signature" minOccurs="1" maxOccurs="1"/>
<element ref="ows:OperationsMetadata" minOccurs="1" maxOccurs="1"/>
</sequence>
</xs:complexType>
<element name="ExtendedXMLDSigCapabilities"
type="ows_security:ExtendedXMLDSigCapabilitiesType"
substitutionGroup="wms:_ExtendedCapabilities"/>
</schema>
```

Conclusion: Applying an enveloped signature as part of the <ExtendedCapabilities> document is possible but not feasible as a specific signature / validation library must be implemented to honor the non-typical position of the signature element.

10.1.3. Recommendation

The client should only trust a GetCapabilities response from a service instance and not a Capabilities instance document obtained from another source. The client can trust the service response, as the communication via HTTPS can be considered secure and that no tampering could have occurred while the response to the GetCapabilities request was submitted to the client.

Perhaps there are other means to secure the Capabilities document, but are considered out of scope for this standard.

10.2. Future Consideration

For the future, it seem to be reasonable to request that a digital signature can be applied to OGC encoding documents; e.g. inside a Capabilities instance document to enable enveloped signatures compliant with the main-stream IT approach (either have the Signature element first or last child of document root). But to secure any OGC instance document, like a FeatureCollection, an OWS Context instance document, etc. it would be necessary to provide an optional element to relevant OGC encoding schemas.

Annex A: Conformance Tests for the Service (normative)

A.1. Conformance Classes

This standard defines three Conformance Classes, illustrated in the figure below.

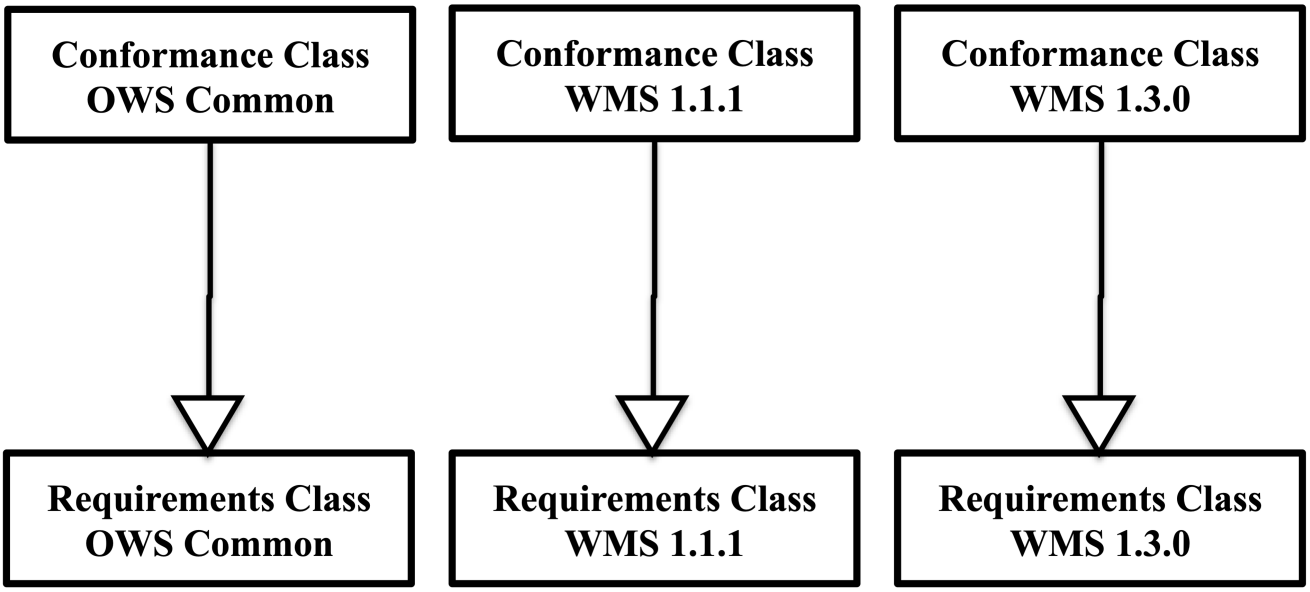

Figure 6. Conformance Classes for a service implementation

Conformance Class	**https://www.opengis.net/spec/security/1.0/conf/cc/owsCommon**
Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/owsCommon**

Conformance Class	**https://www.opengis.net/spec/security/1.0/conf/cc/wms111**
Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/wms111**

Conformance Class	**https://www.opengis.net/spec/security/1.0/conf/cc/wms130**
Target type	Service Implementation
Dependency	**https://www.opengis.net/spec/security/1.0/req/rc/wms130**

A.1.1. Conformance Class Test – Level 1

This mandatory test ensures that a service instance is compliant with one of the defined

Requirements Classes. As discussed in the standard, each Requirements Class reflects a particular Capabilities structure and therefore, this test has three instantiations.

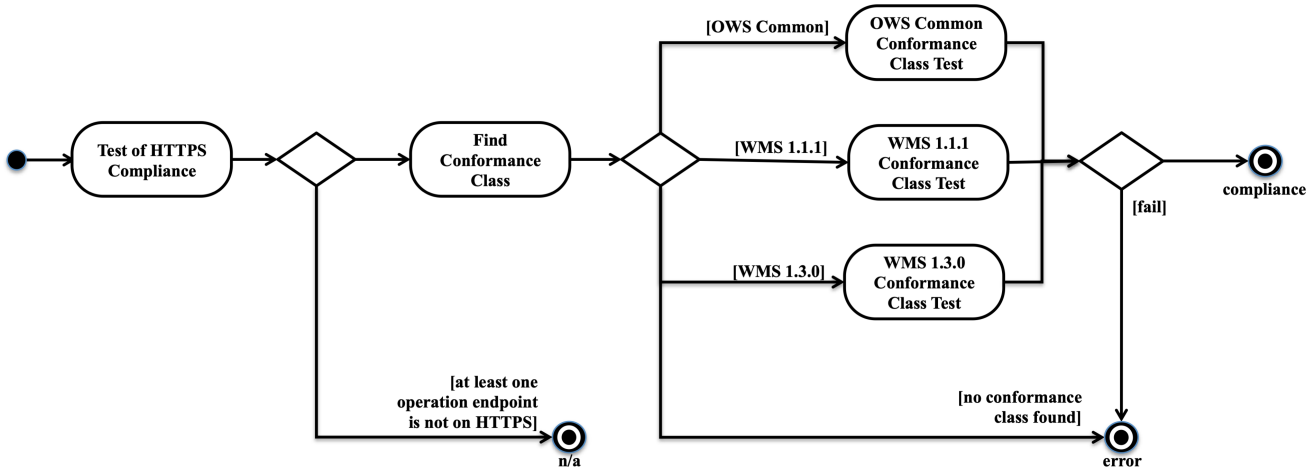

Figure 7. Master activity diagram for testing Compliance

The first test “Test of HTTPS Compliance” verifies that the service instance, described by the Capabilities is compliant to this specification. As stated in requirement 27, a service instance is compliant if all operation endpoint URLs are based on HTTPS.

Table 58. Test if Capabilities refer to compliant service

Conformance Class Test	**https://www.opengis.net/spec/security/1.0/conf/cct/testOfHTTPSCompliance**
Target type	Service Capabilities
Pre-Condition	Service Capabilities obtained
Type	Mandatory
Applicability	Service claims compliance with the OGC Web Services Security standard
Purpose	Verify that all service endpoint URLs of the Capabilities document have scheme ‘https’ (HTTPS).
Test	Use XML parser to fetch **all** operation endpoint URLs exposed in the Capabilities and test if the URL protocol scheme is equal to the literal https.
Pass Condition	If each operation endpoint URL, the URL protocol scheme is https.

Table 59. Find Conformance Class Test

Conformance Class Test	**https://www.opengis.net/spec/security/1.0/conf/cct/findConformanceClass**
Target type	Service Capabilities
Pre-Condition	**https://www.opengis.net/spec/security/1.0/conf/cct/testOfHTTPSCompliance** completed successfully

Type	Mandatory
Applicability	Service is compliant with the OGC Web Services Security standard
Purpose	Find the one Conformance Class to determine where to find the annotations for optional Requirements Class(es)
Test	1. Fetch the namespace and the root element name 2. Execute the individual test realization associated with the Capabilities structure (aka the Conformance Class) **Call Conformance Class Test "WMS 1.1.1" if the name of the root element is WMT_MS_Capabilities** Call Conformance Class Test "WMS 1.3.0" if the name of the root element is WMS_Capabilities ** Call Conformance Class Test "OWS Common" for any other root element name
Pass Condition	Executed Conformance Class Test returns PASS

A.2. Conformance Class Test – Concrete Realization

This Level of Conformance Test is responsible for collecting all exposed Requirements Classes annotated via <ows:Constraint> element(s).

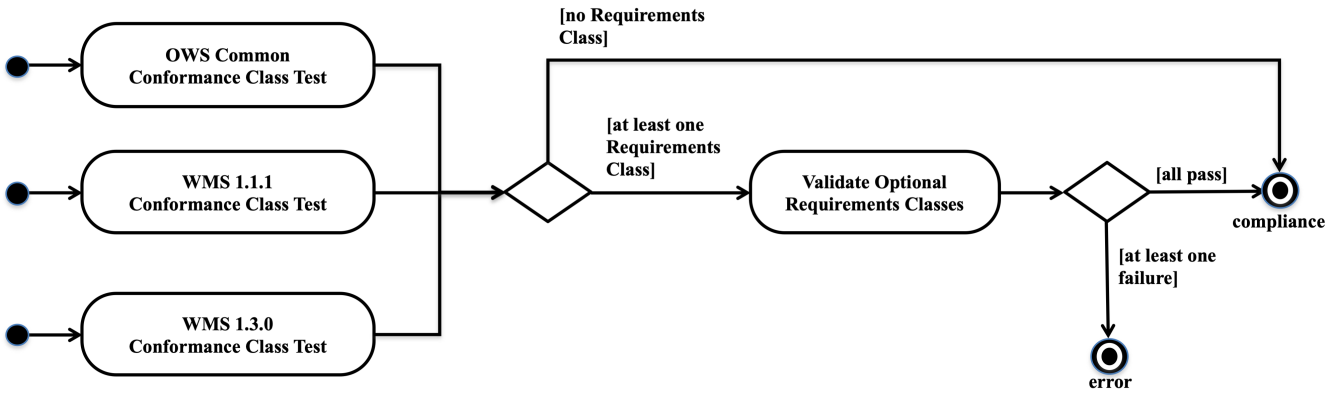

Figure 8. Activity diagram for testing Compliance to an individual Conformance Class

NOTE	The Test "Validate Optional Requirements Classes" is defined in the next section.

Table 60. Conformance Test for OWS Common based Capabilities Structure

Conformance Class Test	**https://www.opengis.net/spec/security/1.0/conf/cct/owsCommon**

Target type	Service Capabilities structure is based on an OWS Common XML schema
Dependency	**https://www.opengis.net/spec/security/1.0/conf/cct/findConformanceClass**
Type	Mandatory
Applicability	Service exposed compliance to Conformance Class **https://www.opengis.net/spec/security/1.0/conf/cc/owsCommon**
Purpose	Fetch all exposed Requirements Classes via <ows:Constraint> element
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element(s) where the element(s) are directly included in the Capabilities structure. 2. Return list of all Requirements Class(es) found or NULL
Pass Condition	Always pass

Table 61. Conformance Test for WMS 1.1.1 based Capabilities Structure

Conformance Class Test	**https://www.opengis.net/spec/security/1.0/conf/cct/wms111**
Target type	Service Capabilities structure is based on an WMS 1.1.1 DTD
Dependency	**https://www.opengis.net/spec/security/1.0/conf/cct/findConformanceClass**
Type	Mandatory
Applicability	Service exposed compliance to Conformance Class **https://www.opengis.net/spec/security/1.0/conf/cc/wms111**
Purpose	Fetch all exposed Requirements Classes via <ows:Constraint> element
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element(s) where the element(s) are included in the DTD extension defined by this standard: <ows_security:SecurityExtendedCapabilities> 2. Return list of all Requirements Classes found or NULL
Pass Condition	Always pass

Table 62. Conformance Test for WMS 1.3.0 based Capabilities Structure

Conformance Class Test	**https://www.opengis.net/spec/security/1.0/conf/cct/wms130**

Target type	Service Capabilities structure is based on WMS 1.3.0 XML schema
Dependency	**https://www.opengis.net/spec/security/1.0/conf/cct/findConformanceClass**
Type	Mandatory
Applicability	Service exposed compliance to Conformance Class **https://www.opengis.net/spec/security/1.0/conf/cc/wms130**
Purpose	Fetch all exposed Requirements Classes via <ows:Constraint> element
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element(s) where the element(s) are included in the XML Schema defined extension by this standard: <ows_security:SecurityExtendedCapabilities> 2. Return list of all Requirements Class(es) found or NULL
Pass Condition	Allways pass

A.3. Testing Optional Requirements Classes

There is only one mandatory Requirements Class to be implemented: HTTPS.

All other Requirements Classes are optional. The existence of a Requirements Class assures that the service has implemented the associated requirements. A Requirements Class can identify the implementation of an IA, support for a security feature or the exposure of additional metadata.

In order to be compliant with this standard, the implementation of one or many optional Requirements Class must be inserted into the Capabilities using the mechanism defined by this standard: Use of the <ows:Constraint> element. This standard also refers to such an element as security annotation.

To assure a particular structure of the <ows:Constraint> element, each Requirements Class imports the requirements defined in the Requirements Class “Identifiers”.

Table 63. Requirements Classes overview (informative)

Requirements Class	… is mandatory	Is dependent upon	…has description
HTTPS	YES	-	HTTP over TLS
Identifiers	NO	-	Use of <ows:Constraint> element to annotate security metadata Use of <ows:Meaning> and <ows:Metadata> to provide URLs for resolving

Requirements Class	… is mandatory	Is dependent upon	…has description
HTTP Methods	NO	-	Support for HTTP 1/1 methods as advertised
HTTP Exception Handling	NO	-	Guarantees HTTP 1/1 compliant error responses incl. HTTP status code Handling Authentication example: 401 ⇒ Authorization Required
W3C CORS	NO	HTTP 1/1	Service supports HTTP header processing according to W3C CORS
Authentication	NO	Exception Handling	To indicate the authentication method used by an operation of the Handling service instance
SAML2	NO	-	To provide URL to SAML2 metadata for supporting client to fetch IdPs
OpenID Connect	NO	-	Required to provide the .well-known URL for the OpenID Provider's configuration
OpenAPI 3.0	NO	-	URL to an OpenAPI 3.0 compliant description of the service instance
Access Control	NO	-	Opportunity to inform client about access constraints — for the purpose of performance to enable client authorization pre-testing
WS-Policy	NO	-	To provide URL for the WS-SecurityPolicy that defines the conditions on accepted SOAP messages
HTTP POST XML Content-Type	NO	-	Support for CR #388 XML Content-Type http://ogc.standardstracker.org/showrequestcgi?id=388

Requirements Classes https://www.opengis.net/spec/security/1.0/req/rc/authentication and

https://www.opengis.net/spec/security/1.0/req/rc/cors have dependencies to other Requirements Classes that must be reflected in the tests.

The Requirements Class https://www.opengis.net/spec/security/1.0/req/rc/authentication depends on the Requirements Class https://www.opengis.net/spec/security/1.0/req/rc/http-exception-handling. This ensures that a service endpoint that has implemented HTTP protocol based Authentication (i.e. HTTP Basic) can return a HTTP status code 401 instead of the OWS Common Exception Report in XML as mandated by the OWS Common specification.

The Requirements Class https://www.opengis.net/spec/security/1.0/req/rc/cors depends on the https://www.opengis.net/spec/security/1.0/req/rc/http-methods Requirements Class. This is necessary to ensure that the service endpoint supports required HTTP methods like OPTIONS and HEAD.

A.4. Test Activity Diagram for Optional Requirements Classes

The following activity diagram illustrates the sequence of tests that SHALL be applied to determine compliance with a particular set of Requirements Classes. The activity diagram takes under consideration the dependency of Requirements Classes.

Note: The Conformance Test for the mandatory Requirements Class HTTPS is already defined in A.2.

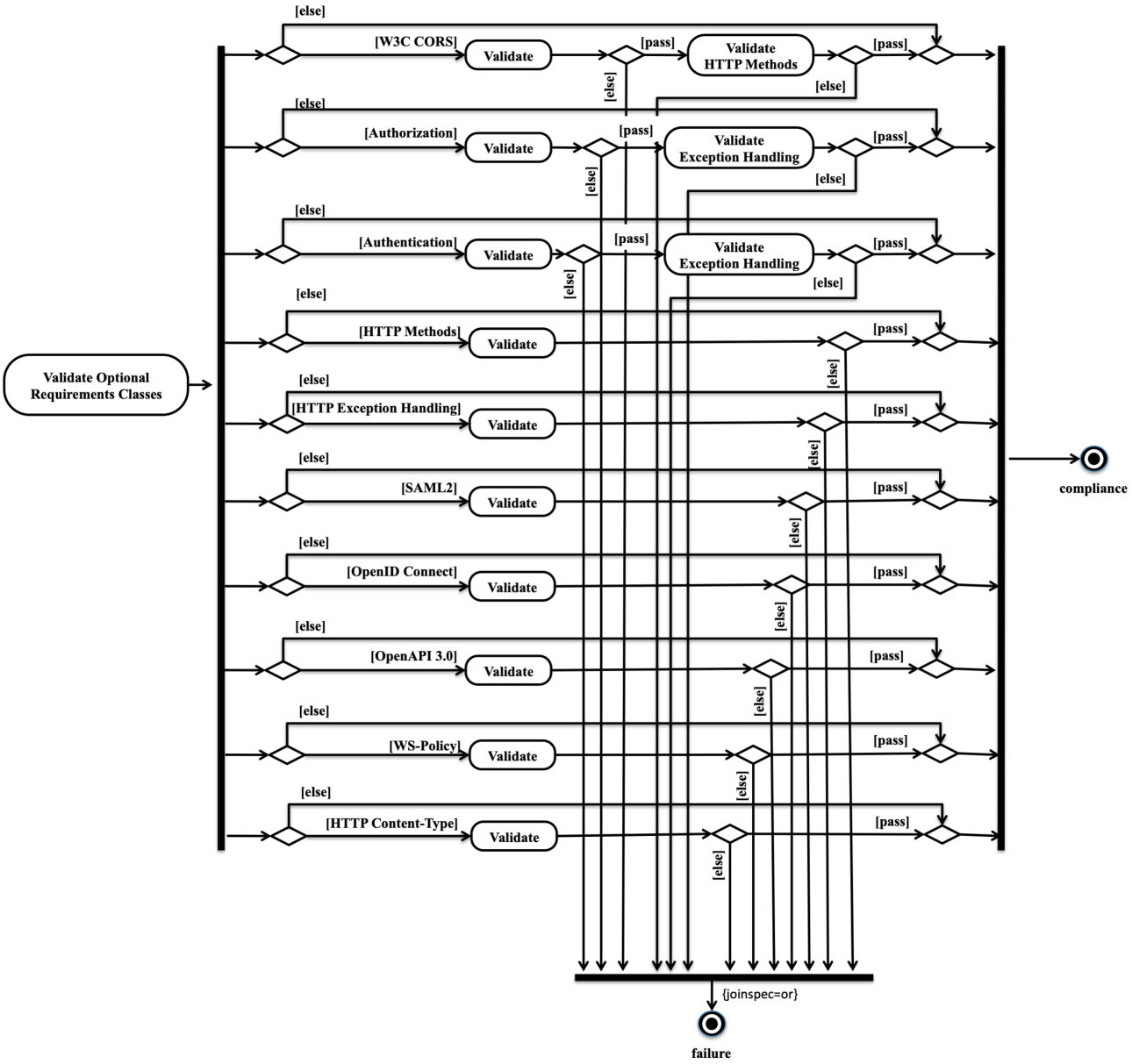

Figure 9. Testing Compliance for optional Requirements Classes

A.4.1. Validate Requirements Class “HTTP Methods”

Table 64. Conformance Test HTTP Methods

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/http-methods**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:http-methods

Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:http-methods
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:http-methods 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:http-methods
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:http-methods

A.4.2. Validate Requirements Class “HTTP Exception Handling”

Table 65. Conformance Test HTTP Exception Handling

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/http-exception-handling**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:http-exception-handling
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:http-exception-handling
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:http-exception-handling 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:http-exception-handling

Pass Condition	Each <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:http-exception-handling

A.4.3. Validate Requirements Class “W3C CORS”

The Requirements Class

https://www.opengis.net/spec/security/1.0/req/rc/cors requires also implementation of https://www.opengis.net/spec/security/1.0/req/rc/http-methods as methods like OPTIONS and HEAD must be supported. Therefore, testing compliance requires to first test the W3C CORS and then the support for HTTP Methods.

Table 66. Conformance Test W3C CORS

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/cors**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:cors
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:cors
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:cors 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:cors 3. Execute Test “Requirements Class HTTP Methods”
Pass Condition	For each operation that is tested towards compliance for Requirements Class urn:ogc:specification:security:1.0:rc:cors must also be compliant to Requirements Class urn:ogc:specification:security:1.0:rc:http-methods

A.4.4. Validate Requirements Class “Authentication”

The Requirements Class https://www.opengis.net/spec/security/1.0/req/rc/authentication requires also implementation of https://www.opengis.net/spec/security/1.0/req/rc/http-exception-handling as

some authentication protocols rely on support for status code other than those defined by superseded OGC standards. Also authentication protocols cannot process OWS Commons XML Exception Reports. Therefore, testing compliance requires to first test the Authentication and then the support for HTTP Exception Handling.

Table 67. Conformance Test Authentication

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/authentication**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:authentication
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:authentication
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:authentication 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:authentication 3. Execute Test “Requirements Class HTTP Exception Handling”
Pass Condition	For each operation that is tested towards compliance for Requirements Class urn:ogc:specification:security:1.0:rc:authentication must also be compliant to Requirements Class urn:ogc:specification:security:1.0:rc:http-exception-handling

A.4.5. Validate Requirements Class “SAML2”

Table 68. Conformance Test SAML2

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/saml2**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes

Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:authentication:saml2
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:authentication:saml2
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:authentication:saml2 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:authentication:saml2
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:authentication:saml2

A.4.6. Validate Requirements Class “OpenID Connect”

Table 69. Conformance Test OpenID Connect

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/oidc**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:authentication:oidc
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:authentication:oidc

Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:authentication:oidc 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:authentication:oidc
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:authentication:oidc

A.4.7. Validate Requirements Class “OpenAPI”

Table 70. Conformance Test OpenAPI

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/openapi**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:openapi
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:openapi
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:openapi 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:openapi
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:openapi

A.4.8. Validate Requirements Class “Authorization”

Table 71. Conformance Test Authorization

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/authorization**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:authorization
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:openapi
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:openapi 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:authorization
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:authorization

A.4.9. Validate Requirements Class “WS-Policy”

Table 72. Conformance Test WS-Policy

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/policy**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:policy
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:policy

Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:policy 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:policy
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:policy

A.4.10. Validate Requirements Class “HTTP Content-Type”

Table 73. Conformance Test HTTP Content-Type

Requirements Class Test	**https://www.opengis.net/spec/security/1.0/req/rct/content-type**
Target type	Service Capabilities
Dependency	Validate Optional Requirements Classes
Type	Mandatory
Applicability	Service provides operation(s) for which it claims compliance to Requirements Class urn:ogc:specification:security:1.0:rc:content-type
Purpose	Verify for a compliant implementation of Requirements Class urn:ogc:specification:security:1.0:rc:content-type
Test	1. Parse the GetCapabilities response for each instance of <ows:Constraint> element where the gml:identifier equals urn:ogc:specification:security:1.0:rc:content-type 2. Verify that the <ows:Constraint> element structure and content conforms to Requirements Class urn:ogc:specification:security:1.0:rc:content-type
Pass Condition	Each tested <ows:Constraint> element is compliant with Requirements Class urn:ogc:specification:security:1.0:rc:content-type

Annex B: Conformance Tests for the Client (normative)

The purpose of the abstract conformance test is to verify client implementations interacting with a test harness, simulating a service compliant to this standard. The mandatory tests defined in this section target on the client functionality and the interface compliance. However, the tests to not take under consideration the correct processing semantics for client function implementations.

The implementation of an actual service test harness is outside the scope of this standard. However, the test harness must be capable to be configured with compliance for each Conformance Class as defined by this standard. It must also be possible to configure the test harness to produce annotated capabilities with any combination of Requirements Classes.

Important for the client to work with a service compliant to this standard is

full support for HTTPS as defined in

1. Requirements Class urn:ogc:specification:security:1.0:rc:https and
2. to work with partial capabilities as defined in Requirements Class **https://www.opengis.net/spec/security/1.0/req/rc/clientParsing**

The inspection of the client test results cannot take place programmatically. Therefore, a human must compare the actual behavior of the client with the expected behavior as defined in the Conformance Tests for the Client.

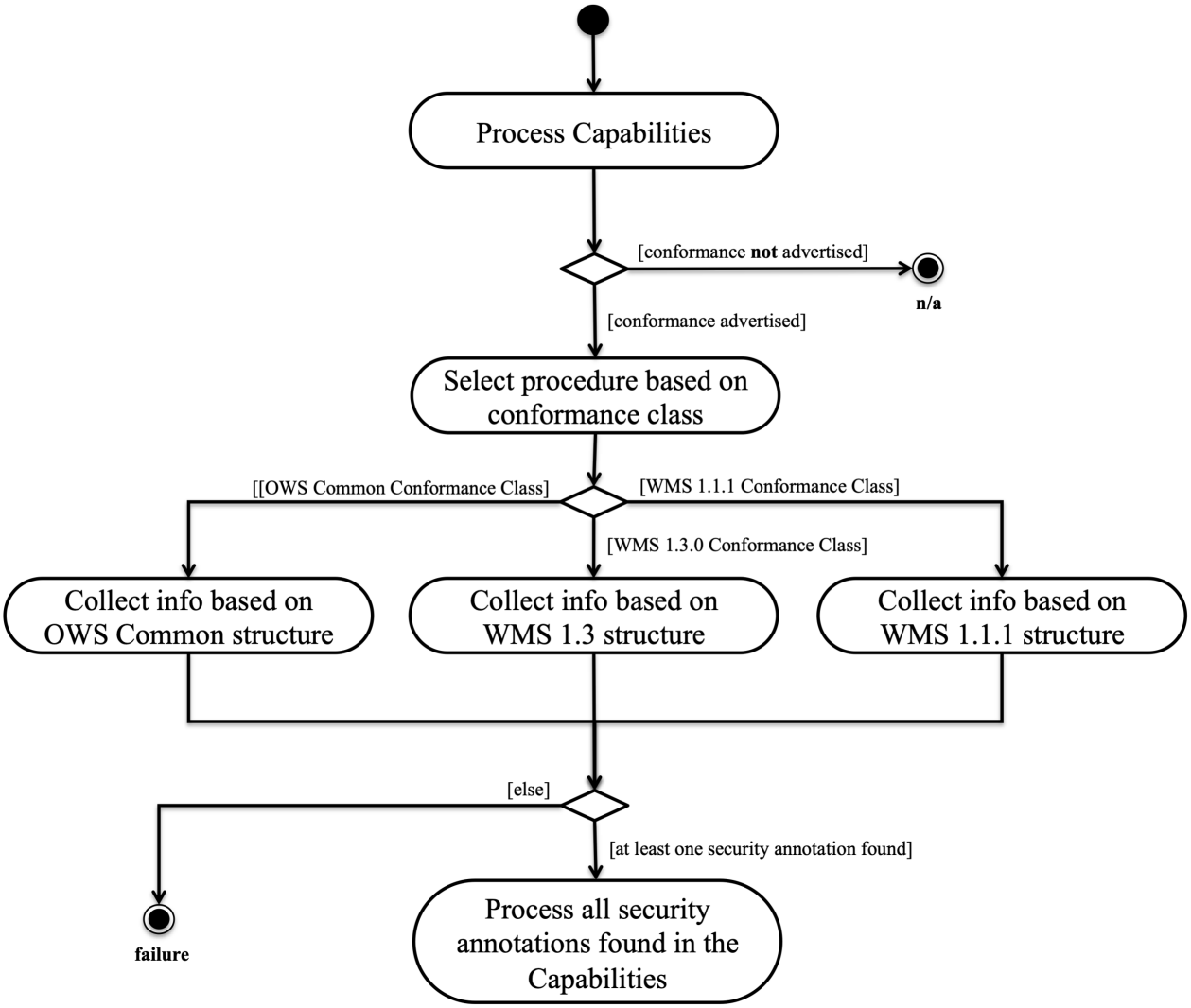

Figure 10. Illustration of client functionality for processing the Capabilities document exposed by a compliant service or the test harness

B.1. Conformance Test HTTPS

The only mandatory Requirements Class defined by this standard is urn:ogc:specification:security:1.0:rc:https

This test SHALL be applied to determine that the client is capable to accept a service connection via HTTPS.

B.2. Conformance Test Working on Capabilities with no Content section

The Requirement **https://www.opengis.net/spec/security/1.0/req/rc/clientParsing** defines a mandatory client behavior that is important to be tested.

When a client receives the Capabilities that does not contain the “content” section, the client must send a GetCapabilities request to the endpoint advertised in the Capabilities.

NOTE

The execution of the GetCapabilities endpoint advertised in the Capabilities might require the client to be compliant with implemented security controls. But assuming the client is a compliant implementation, it must be able to execute the GetCapabilities operation even if protected.

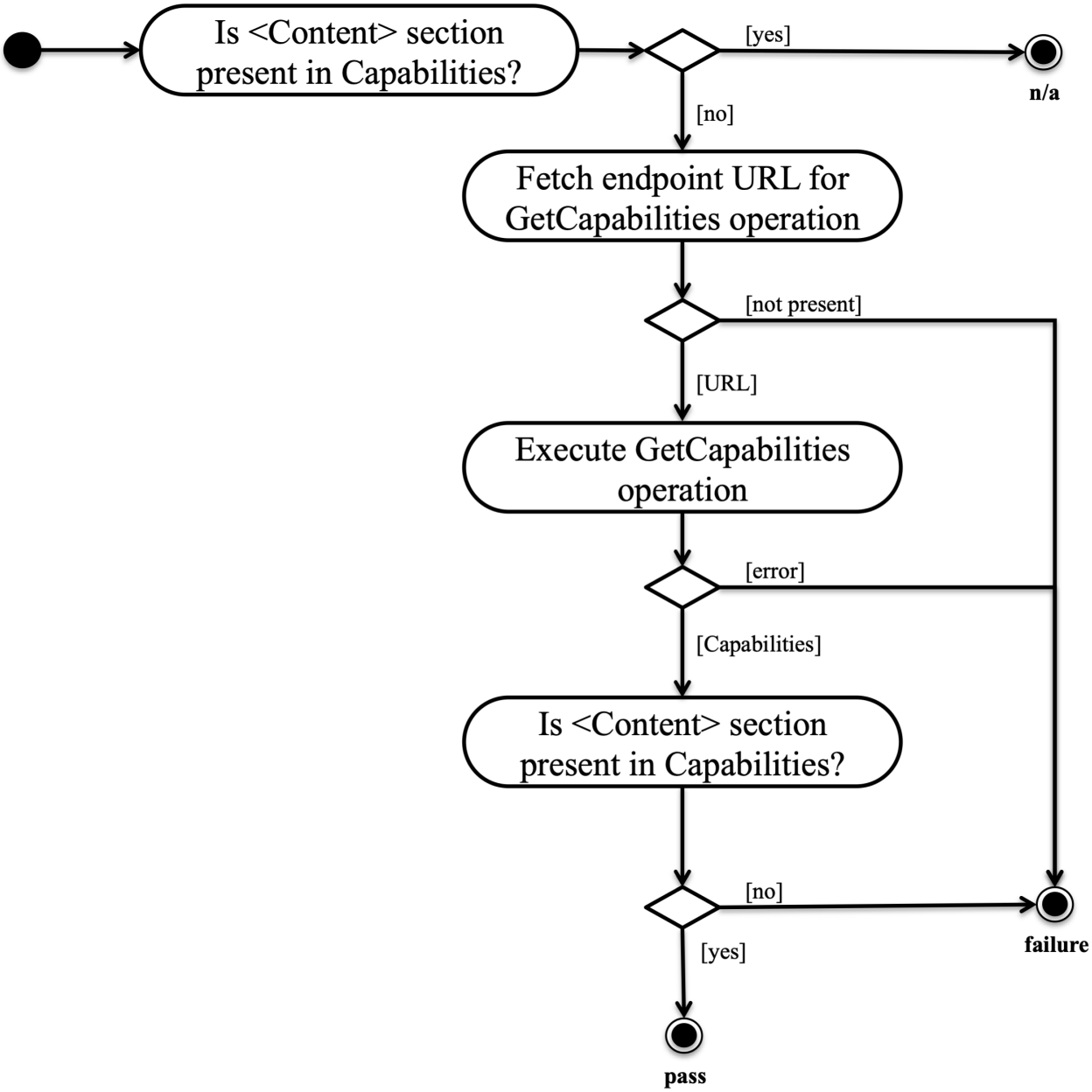

Figure 11. Illustration of the test for a client to work on Capabilities with no <Content> section

The pass or failure of this test can be verified by a human when executing the client on the test harness that produces Capabilities with no <Content> section. The verification is possible via listing the resources that are included in the <Content> element. If the client does not display any Layers, Feature types, etc. to select from, this test fails.

Annex C: Conformance Tests for the Authentication Code Resolver (normative)

The purpose of the tests defined in this section is to verify that an Authentication Code Resolver is compliant with this specification. This is in particular important for testing compliance of your own registry when you cannot use the OGC Authentication Code Resolver using your own.

The use of your own Authentication Code Resolver can be installed such that the service constructs annotated capabilities where the resolvable links to load authentication code definitions is pointing to your own resolver.

Positive Conformance

The Resolver must accept any valid URI for fetching an authentication code.

Test 1: Requesting the URL as defined in Requirement 68, the resolver must return the original Authentication Codelist with Content-Type “text/xml”.

Initiate an HTTP GET request using a resolvable URI for a definition included in your Authentication Codelist. The result must be a HTML page with content-type “text/html” including the human readable definition of the authentication code.

Test 2: When using a resolvable URI for an authentication code, the resulting HTML page contains the single authentication code definition referenced by the authentication code.

Negative Conformance

Test: Non existing Authentication Code

When trying to resolve a non-existing authentication code, the Resolver must return HTTP status code 404 with error “Authentication Code not resolvable”.

Annex D: Initial Authentication Codelist (informative)

The official mandatory Authentication Codelist can be resolved via the URI as specified in Requirement 68.

Table 74. Initial Authentication Codelist (informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<gmx:CT_CodelistCatalogue
xmlns="http://www.opengis.net/def/security/1.0/codelist/authentication"
  xmlns:gmx="http://www.isotc211.org/2005/gmx"
xmlns:gco="http://www.isotc211.org/2005/gco"
  xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
  http://www.isotc211.org/2005/gmx
http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/gmx/gmx.xsd
  http://www.isotc211.org/2005/gco
http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/gco/gco.xsd
  http://www.opengis.net/gml/3.2
http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19136_Schemas/gml.xsd">
  <!--=====Catalogue description======-->
  <gmx:name>
    <gco:CharacterString>authnCodelist</gco:CharacterString>
  </gmx:name>
  <gmx:scope>
    <gco:CharacterString>OGC codelist for description security annotations regarding
authentication</gco:CharacterString>
  </gmx:scope>
  <gmx:fieldOfApplication>
    <gco:CharacterString>OGC</gco:CharacterString>
  </gmx:fieldOfApplication>
  <gmx:versionNumber>
    <gco:CharacterString>1.0</gco:CharacterString>
  </gmx:versionNumber>
  <gmx:versionDate>
    <gco:Date>2019-01-23</gco:Date>
  </gmx:versionDate>
  <!--============================================================================-->
  <!--============================================================================-->
  <!--============================== Codelists
=======================================-->
  <!--=== Authentication ===-->
  <gmx:codelistItem>
    <gmx:CodeListDictionary gml:id="authentication">
      <gml:description>identification of authentication methods</gml:description>
```

```
      <gml:identifier codeSpace="OGC">
urn:ogc:def:security:authentication</gml:identifier>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="HTTP_BASIC">
          <gml:description>The "basic" authentication scheme is based on the model
that the
                    client must authenticate itself with a user-ID and a password for
                    each realm.  The realm value should be considered an opaque string
                    which can only be compared for equality with other realms on that
                    server. The server will service the request only if it can
validate
                    the user-ID and password for the protection space of the Request-
URI.
                    There are no optional authentication parameters.</gml:description>
          <gml:identifier codeSpace="IETF"
>urn:ogc:def:security:authentication:ietf:2617:Basic</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="HTTP_DIGEST">
          <gml:description>
            Like Basic Access Authentication, the Digest scheme is based on a
                simple challenge-response paradigm. The Digest scheme challenges
                using a nonce value. A valid response contains a checksum (by
                default, the MD5 checksum) of the username, the password, the given
                nonce value, the HTTP method, and the requested URI. In this way, the
                password is never sent in the clear. Just as with the Basic scheme,
                the username and password must be prearranged in some fashion not
                addressed by this document.</gml:description>
          <gml:identifier codeSpace="IETF"
>urn:ogc:def:security:authentication:ietf:2617:Digest</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="OAUTH2_BEARER_TOKEN">
         <gml:description>
           In the scenario supported by the OAuth 2.0 SSO profile, a web user or
service either accesses a resource
            at a service provider, or accesses an identity provider such that the
service provider and desired resource are understood
            or implicit. The web user authenticates (or has already authenticated) to
the identity provider, Which then produces an
            authorization grant which was then used by an authorization service to
return an access token.  This access token then
            subtitutes as both authentication and authorization on future
requests.</gml:description>
          <gml:identifier codeSpace="IETF"
>urn:ogc:def:security:authentication:ietf:6750:Bearer</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
```

```
        <gmx:CodeDefinition gml:id="TLS_CLIENT_CERTIFICATE">
          <gml:description>
            This type of authentication is an extension to the TLS handshake as
outlined in section 7.4.4:
            "A non-anonymous server can optionally request a certificate from
            the client, if appropriate for the selected cipher suite.  This
            message, if sent, will immediately follow the ServerKeyExchange
            message (if it is sent; otherwise, this message follows the
            server's Certificate message)."[RFC 5246]
            In case the client cannot provide a suitable and valid certificate, no TLS
            connection gets established</gml:description>
          <gml:identifier codeSpace="IETF"
>urn:ogc:def:security:authentication:ietf:5246:client_certificate</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="USERNAME_TOKEN">
          <gml:description>WSSE UsernameToken as specified in https://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:ogc:def:security:authentication:wsse:username_token</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_InternetProtocol">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocol</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_InternetProtocolPassword">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocolPassword</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_Kerberos">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:Kerberos</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_MobileOneFactorUnregistered">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
```

```
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:MobileOneFactorUnregistered</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_MobileTwoFactorUnregistered">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorUnregistered</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_MobileOneFactorContract">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:MobileOneFactorContract</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_MobileTwoFactorContract">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorContract</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_Password">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:Password</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PasswordProtectedTransport">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PreviousSession">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</gml:identifier>
        </gmx:CodeDefinition>
```

```
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PublicKeyX509">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:X509</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PublicKeyPGP">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS">
urn:oasis:names:tc:SAML:2.0:ac:classes:PGP</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PublicKeySPKI">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:SPKI</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PublicKeyXMLDigitalSignature">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:XMLDSig</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_Smartcard">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:Smartcard</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_SmartcardPKI">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:SmartcardPKI</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_SoftwarePKI">
```

```
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:SoftwarePKI</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_Telephony">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:Telephony</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_TelephonyNomadic">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:NomadTelephony</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_PersonalTelephony">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:PersonalTelephony</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_AuthenticatedTelephony">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:AuthenticatedTelephony</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_SecureRemotePassword">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:SecureRemotePassword</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_TLSClient">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
```

```
>urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_TimeSyncToken">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:TimeSyncToken</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
      <gmx:codeEntry>
        <gmx:CodeDefinition gml:id="SAML2_Unspecified">
          <gml:description>As specified in https://docs.oasis-
open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf</gml:description>
          <gml:identifier codeSpace="OASIS"
>urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified</gml:identifier>
        </gmx:CodeDefinition>
      </gmx:codeEntry>
    </gmx:CodeListDictionary>
  </gmx:codelistItem>
  <!--=== EOF ===-->
</gmx:CT_CodelistCatalogue>
```

Annex E: Using Authentication Codelist in ISO Metadata (informative)

For completeness, this Annex illustrates the use of the authentication Codelist specified by this standard to describe the existence of an authentication IA control. The authentication Codelist can be imported into the ISO service metadata as an external codelist. Based on that import, the authentication codes can be used to describe access constraints.

Table 75. Using the Authentication Codelist in ISO Service Metadata (gmd:resourceConstraints element only)

```
<gmd:resourceConstraints>
  <gmd:MD_LegalConstraints>
    <gmd:accessConstraints>
      <gmd:MD_RestrictionCode
codeList="./resources/Codelist/gmxCodelists.xml#MD_RestrictionCode"
codeListValue="copyright">copyright</gmd:MD_RestrictionCode>
    </gmd:accessConstraints>
    <gmd:accessConstraints>
      <gmd:MD_RestrictionCode
codeList="./resources/Codelist/gmxCodelists.xml#MD_RestrictionCode"
codeListValue="license">license</gmd:MD_RestrictionCode>
    </gmd:accessConstraints>
    <gmd:accessConstraints>
      <gmd:MD_RestrictionCode
codeList="./resources/Codelist/gmxCodelists.xml#MD_RestrictionCode"
codeListValue="otherRestrictions"/>
    </gmd:accessConstraints>
    <gmd:accessConstraints>
      <gmd:MD_RestrictionCode
codeList="https://www.opengis.net/def/security/1.0/codelist/authentication"
codeListValue="urn:ogc:def:security:authentication:ietf:2617:Basic"/>
    </gmd:accessConstraints>
    <gmd:accessConstraints>
      <gmd:MD_RestrictionCode
codeList="https://www.opengis.net/def/security/1.0/codelist/authentication"
codeListValue="urn:ogc:def:security:authentication:ietf:2617:Digest"/>
    </gmd:accessConstraints>
  </gmd:MD_LegalConstraints>
</gmd:resourceConstraints>
<gmd:resourceConstraints>
  <gmd:MD_SecurityConstraints>
    <gmd:classification>
      <gmd:MD_ClassificationCode
codeList="./resources/Codelist/gmxCodelists.xml#MD_ClassificationCode"
codeListValue="unclassified" />
    </gmd:classification>
  </gmd:MD_SecurityConstraints>
</gmd:resourceConstraints>
```

As illustrated in the figure above, the additional access constraints are authentication via:

(i) **https://www.opengis.net/def/security/1.0/codelist/authentication/** urn:ogc:def:security:authentication:ietf:2617:Basic or

(ii) **https://www.opengis.net/def/security/1.0/codelist/authentication/** urn:ogc:def:security:authentication:ietf:2617:Digest

Annex F: Revision History

Date	Release	Author	Paragraph modified	Description
15.02.17	0.1	AM [6: Andreas Matheus]	All	Creation
20.02.17	0.2	AM	All	Incorporation of comments from Frank Terpstra
13.03.17	0.3	AM	All	Andreas, Chuck, Dave, Frank, Michael telecom 13.03.17
27.03.17	0.4	AM	All	Andreas, Chuck, Dave, Michael telecom 27.03.17
08.05.17	0.5	AM	All	Incorporation of comments from Dave Wesloh, Don Sullivan and Frank Terpstra
19.06.17	0.6	AM	All	Incorporation of comments from Dave Wesloh
18.07.17 - 01.08.17	0.7	AM	All	Incorporating results from discussions in meetings and updating the structure of the document for improved readability
24.08.17	0.8	AM	All but mainly 7.5	Revised Exception Handling
18.09.17 - 26.09.17	0.9	AM	Security Considerations	Creation
06.11.17	0.10	AM	All Appendix B	Incorporation of comments from Frank Terpstra

13.11.17	0.11	AM	All Annex C All	Incorporation of comments from Chuck Heazel and Dave Wesloh Including initial Authentication Codelist Preparation for RFC
15.01.18	0.12	AM	Section 4, 6, 7	Incorporation of NA guidance
05.03.18	0.13	AM	Section 7	Harmonizing Constraint element structure for OWS Common 1.0 and 1.1.0 / 2.0
13.03.18	0.14	AM	All	Incorporating comments from Carl Reed Clarification of sections 8 and 9
27.03.18	0.15	AM	All, Section 9 Annex D	Clean-up Clarification on resolving the Authentication Codelist Initial Codelist update
16.04.18	0.16	AM	All	Fix of broken internal links Annex D: initial list of authentication codes updated

24.04.18	0.17	AM	All	Accommodate the requirement to make any HTTPS hosted service required to be compliant with this standard **without** being required to change the service capabilities. Only hosting the service on HTTPS is required.
17.05.18	1.0	AM	All	Finalizing the document for adoption vote.
13.09.18	1.0.1	AM	Annex A	Incorporating comments from NR Canada received from TC voting
15.10.18	1.0.1	AM	All	Final editorial edits • Change requirements numbering to sequential numbers • Change conformance class to requirements class throughout the main document • Moved conformance class definitions from section 6 to Annex A

Annex G: Bibliography

1.	Matheus, A.: OGC 15-022: OGC® Testbed 11 Engineering Report: Implementing Common Security Across the OGC Suite of Service Standards
2.	Matheus, A.: OGC 16-048r1: OGC® Testbed 12 Engineering Report: OWS Common Security Extension
3.	Matheus, A.: OGC 17-021: OGC® Testbed 13 Engineering Report: Security