Draft ETSI TS 119 432 V0.0.8 (2018-12)

Electronic Signatures and Infrastructures (ESI);

Protocols for remote digital signature creation

<

**technical Standard**

Reference

DTS/ESI-0019432

Keywords

 electronic signature, security, trust services, remote, protocol

***ETSI***

650 Route des Lucioles

F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C

Association à but non lucratif enregistrée à la

Sous-préfecture de Grasse (06) N° 7803/88

***Important notice***

The present document can be downloaded from:
<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>.

If you find errors in the present document, please send your comment to one of the following services:
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2018.

All rights reserved.

**DECT**TM, **PLUGTESTS**TM, **UMTS**TM and the ETSI logo are trademarks of ETSI registered for the benefit of its Members.
**3GPP**TM and **LTE**™ are trademarks of ETSI registered for the benefit of its Members and
of the 3GPP Organizational Partners.
**oneM2M** logo is protected for the benefit of its Members.
**GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Contents

Intellectual Property Rights 7

Foreword 7

Modal verbs terminology 7

Introduction 7

1 Scope 8

2 References 8

2.1 Normative references 8

2.2 Informative references 9

3 Definitions and abbreviations 10

3.1 Definitions 10

3.2 Abbreviations 11

4 Signature creation process, service decomposition 12

4.1 Signature creation process steps and data elements 12

4.2 Service main components and interfaces 13

4.3 Signature Creation Application 13

4.3.1 Signer’s document and hashing 13

4.3.2 DTBS composition and formatting 13

4.3.3 DTBS preparation 14

4.3.4 SDO composer 14

4.4 Server Signing Application 14

4.4.1 Signature creation 14

4.4.1.1 Signature activation 14

4.4.1.2 Signature creation by SCDev 14

5 Architectures for server signing 15

5.1 Overview 15

5.2 Introduction to architectures 15

5.3 Remote signing services with SCAL1 15

5.4 Remote signing services with SCAL2 17

5.5 Security, integrity and confidentiality 19

6 Protocol profiles specification 19

6.1 Introduction 19

6.2 OASIS DSSX TC XML related protocol 19

6.3 CSC JSON related protocol 20

7 Protocol components definitions 20

7.1 Introduction 20

7.2 Component for asynchronous/synchronous operation mode selection 20

7.2.1 Component semantics 20

7.2.2 JSON related component 21

7.2.3 XML related component 21

7.3 Component for identification of the request 22

7.3.1 Component semantics 22

7.3.2 JSON related component 22

7.3.3 XML related component 22

7.4 Component for credential authorization 22

7.4.1 Component semantics 22

7.4.2 JSON related component 22

7.4.3 XML related component 22

7.5 Component for defining optional data to be returned 23

7.5.1 Component semantics 23

7.5.2 JSON related component 23

7.5.3 XML related component 23

7.6 Component for defining the validity period for asynchronous requests 23

7.6.1 Component semantics 23

7.6.2 JSON related component 24

7.6.3 XML related component 24

7.6.4 Processing model 24

7.7 Component for service authentication 24

7.7.1 Component semantics 24

7.7.2 JSON related component 24

7.7.3 XML related component 24

7.8 Component for identifying signature credentials 24

7.8.1 Component semantics 24

7.8.2 JSON related component 25

7.8.3 XML related component 25

7.9 Component for language and region selection 25

7.9.1 Component semantics 25

7.9.2 JSON related component 25

7.9.3 XML related component 25

7.10 Component for specifying the contents from certificate info to be returned 26

7.10.1 Component semantics 26

7.10.2 JSON related component 26

7.10.3 XML related component 26

7.10.4 Processing model 27

7.11 Component for managing digital signatures transactions 27

7.11.1 Component semantics 27

7.11.2 JSON related component 27

7.11.3 XML related component 28

7.12 Component for service policy selection 28

7.12.1 Component semantics 28

7.12.2 JSON related component 28

7.12.3 XML related component 28

7.13 Component for signature creation policy selection 28

7.13.1 Component semantics 28

7.13.2 JSON related component 28

7.13.3 XML related component 29

7.14 Component for optional signature attributes/properties selection 29

7.14.1 Component semantics 29

7.14.2 JSON related component 29

7.14.3 XML related component 30

7.15 Component for protocol identifier 30

7.15.1 Component semantics 30

7.15.2 JSON related component 30

7.15.3 XML related component 31

7.16 Component for requesting specific signature formats 31

7.16.1 Component semantics 31

7.16.2 JSON related component 31

7.16.3 XML related component 32

7.17 Component for signer identification 33

7.17.1 Component semantics 33

7.17.2 JSON related component 33

7.17.3 XML related component 33

7.18 Component for specifying response URL 33

7.18.1 Component semantics 33

7.18.2 JSON related component 33

7.18.3 XML related component 34

7.18.4 Processing model 34

7.19 Component for submitting document(s) or hash(es) to be signed 34

7.19.1 Component semantics 34

7.19.2 JSON related component 34

7.19.3 XML related component 35

7.20 Component for returning service information 35

7.20.1 Component semantic 35

7.20.2 JSON related component 36

7.20.3 XML related component 37

7.21 Component for returning signed documents or signatures 37

7.21.1 Component semantics 37

7.21.2 JSON related component 37

7.21.3 XML related component 38

7.22 Component for returning signing certificate information 38

7.22.1 Component semantics 38

7.22.2 JSON related component 38

7.22.3 XML related component 40

7.23 Component for returning the list of the signing certificate(s) 40

7.23.1 Component semantics 40

7.23.2 JSON related component 40

7.23.3 XML related component 41

7.24 Component for notifying operation result(s) 41

7.24.1 Component semantics 41

7.24.2 JSON related component 41

7.24.3 XML related component 41

7.25 Component for service policy identification 42

7.25.1 Component semantics 42

7.25.2 JSON related component 42

7.25.3 XML related component 42

7.26 Component for identification of the response 42

7.26.1 Component semantics 42

7.26.2 JSON related component 42

7.26.3 XML related component 42

7.27 Component for signature creation policy identification 43

7.27.1 Component semantics 43

7.27.2 JSON related component 43

7.27.3 XML related component 43

7.28 Component for returning credential authorization mode 43

7.28.1 Component semantics 43

7.28.2 JSON related component 44

7.28.3 XML related component 44

7.29 Component for returning digital signature value(s) 44

7.29.1 Component semantics 44

7.29.2 JSON related component 45

7.29.3 XML related component 45

7.30 Component for returning sole control assurance level required 45

7.30.1 Component semantics 45

7.30.2 JSON related component 45

7.30.3 XML related component 45

8 Profiles for remote signature creation 47

8.1 Introduction 47

8.2 Profile for signature request (A) 47

8.2.1 Profile semantics 47

8.2.2 JSON related component 48

8.2.3 XML related component 48

8.3 Profile for signature response (B) 49

8.3.1 Profile semantics 49

8.3.2 JSON related component 49

8.3.3 XML related component 49

8.4 Profile for DSVs creation request (C) 49

8.4.1 Profile semantics 49

8.4.2 JSON related component 50

8.4.3 XML related component 50

8.5 Profile for digital signature value response (D) 50

8.5.1 Profile semantics 50

8.5.2 JSON related component 51

8.5.3 XML related component 51

8.6 Profile for asynchronous processing (E) 51

8.6.1 Profile semantics 51

8.6.2 JSON related component 52

8.6.3 XML related component 52

8.7 Profile for signing certificates list request (F) 52

8.7.1 Profile semantics 52

8.7.2 JSON related component 52

8.7.3 XML related component 52

8.8 Profile for signing certificates list response (G) 53

8.8.1 Profile semantics 53

8.8.2 JSON related component 53

8.8.3 XML related component 53

8.9 Profile for certificate information retrieval request (H) 53

8.9.1 Profile semantics 53

8.9.2 JSON related component 54

8.9.3 XML related component 54

8.10 Profile for certificate information retrieval response (I) 54

8.10.1 Profile semantics 54

8.10.2 JSON related component 54

8.10.3 XML related component 54

8.11 Profile for service information request (J) 55

8.11.1 Profile semantics 55

8.11.2 JSON related component 55

8.11.3 XML related component 55

8.12 Profile for service information response (K) 55

8.12.1 Profile semantics 55

8.12.2 JSON related component 55

8.12.3 XML related component 55

8.13 Component use summary 56

Annex A (normative): XML and JSON Schema files 57

A.1 JSON Schema file location for “$schema” "http://uri.etsi.org/19342/v1.1.1/json#" 57

A.2 XML Schema file location for namespace http://uri.etsi.org/19342/v1.1.1# 57

History 58

# Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server ([https://ipr.etsi.org](https://ipr.etsi.org/)).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

# Foreword

This draft Technical Specification (TS) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

# Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](http://portal.etsi.org/Help/editHelp%21/Howtostart/ETSIDraftingRules.aspx) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

# Introduction

Standards for digital signatures have generally been developed for a long time considering solutions tailored to the characteristics of devices such as desktop computers and laptops where all signature processing was done in one system locally to the user. These traditional signature solutions assume that the signer uses smart cards or tokens to create any required digital signatures. Given developments in distributed systems, cloud computing, mobile equipment and related technologies, solutions have been emerging in the last few years where the process of digital signature creation and construction of AdES format is done in a distributed way with different steps of the process carried out by different systems/services that may be controlled by different actors.

The present document specifies protocols and interfaces for components providing specific functionalities as part of a process for remote digital signatures creation and construction of AdES formats. The present document aims at supporting electronic signatures and electronic seals, including qualified electronic signatures and qualified electronic seals according to the Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC (the eIDAS Regulation).

# 1 Scope

The present document specifies protocols and interfaces applicable when the process of creating AdES digital signatures as defined by ETSI EN 319 102-1 [4] and/or Digital Signature Values, as result of Data To Be Signed Representations signatures, is carried out by a distributed solution comprised of two or more systems/services/components.

The present document is limited to remote server signing, i.e. the signing key is held in a remote shared service.

NOTE 1: Remote signature creation with local signing, i.e. the signing key is held with the signer’s personal device but other steps in the signature creation are carried out by means of networked services, is a possible solution but protocols for such architecture are not covered in the present document.

Finally, the present document specifies two bindings, each one in a different syntax (XML and JSON), for each of the aforementioned protocols.

As far as it has been possible and suitable, the protocols have re-used constructs of CSC JSON and OASIS DSS-X XML specifications. When this has not been possible the present document specifies new components semantically and also syntactically in the two formats: XML and JSON.

The authorized signer’s use of its key for signing requires users to provide multiple proofs of their claimed identity before being granted access to the needed set of resources. The way in which the user identity verification process is carried out by the service provider or any suggestion concerning the usage of multi-factor authentication mechanisms is out of the scope of the present document.

# 2 References

## 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or nonspecific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <http://docbox.etsi.org/Reference>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] Cloud Signature Consortium Standard: "Architectures and protocols for remote signature applications. Version 1.0.3.0".

[2] OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 – Commitee Specification Draft 01.

[3] OASIS Standard: Advanced Electronic Signature Profiles of the OASIS Digital Signature Service Version 2.0 – Working Draft 02.

[4] ETSI EN 319 102-1: "Electronic signatures and infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".

[5] OASIS Standard: Asynchronous Processing Abstract Profile of the OASIS Digital Signature Services Version 1.0.

[6] CEN EN 419 241-1: "Trustworthy Systems supporting Server Signing; Part 1: General System Security Requirements".

[7] IETF RFC 5646: "Tags for Identifying Languages".

## 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or nonspecific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC.

[i.2] ETSI SR 019 020: "The framework for standardization of signatures; Standards for AdES digital signatures in mobile and distributed environment".

[i.3] ETSI TS 119 441: "Electronic signatures and infrastructures (ESI); Policy and security requirements for trust service providers providing AdES digital signature validation services".

[i.4] ETSI TS 119 442: "Electronic signatures and infrastructures (ESI); Protocol profiles for trust service providers providing AdES digital signature validation services".

[i.5] ETSI EN 319 122-1: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures".

[i.6] ETSI EN 319 122-2: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 2: Extended CAdES signatures".

[i.7] ETSI EN 319 132-1: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 1: Building blocks and XAdES baseline signatures".

[i.8] ETSI EN 319 132-2: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 2: Extended XAdES signatures".

[i.9] ETSI EN 319 142-1: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 1: Building blocks and PAdES baseline signatures".

[i.10] ETSI EN 319 142-2: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 2: Additional PAdES signatures profiles".

[i.11] ETSI TR 119 001: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures; Definitions and abbreviations".

[i.12] ETSI TS 119 312: "Electronic Signatures and Infrastructures (ESI); Cryptographic Suites".

[i.13] IETF RFC 6749: "The OAuth 2.0 Authorization Framework”.

[i.14] IETF RFC 7519: "JSON Web Token (JWT)”.

[i.15] IETF RFC 3061: "A URN Namespace of Object Identifiers”.

[i.16] IETF RFC 8017: "PKCS #1: RSA Cryptography Specifications Version 2.2”.

[i.17] ETSI TS 103 173: " Electronic Signatures and Infrastructures (ESI); CAdES Baseline Profile".

[i.18] ETSI TS 103 171: "Electronic Signatures and Infrastructures (ESI); XAdES Baseline Profile".

[i.19] ETSI TS 103 172: " Electronic Signatures and Infrastructures (ESI); PAdES Baseline Profile".

# 3 Definitions and abbreviations

## 3.1 Definitions

For the purposes of the present document, the terms and definitions given in ETSI TR 119 001 [i.11] and the following apply:

**AdES (digital) signature:** digital signature that is either a CAdES signature, or a PAdES signature or a XAdES signature

**client application:** an application running in a signer’s environment that accesses the services made available by the SCASC and/or the SSASC

**digital signature:** data appended to, or a cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

**digital signature value:** result of thecryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

**remote signature creation device**: signature creation device used remotely from signer perspective and providing control of signing operation on the signer’s behalf

**server signing application**: application using a remote signature creation device to create a digital signature value on behalf of a signer

**server signing application service component**: TSP service component employing a server signing application

**server signing application service provider**: TSP operating a server signing application service component

**signature creation application**: application within the signature creation system that creates the AdES digital signature and relies on the SCDev to create a digital signature value

NOTE: The SCDev can be managed by the SSASC.

**signature creation application service component**: TSP service component employing a signature creation application

**signature creation application service provider**: TSP operating a signature creation application service component

**signature creation device:** configured software or hardware used to implement the signature creation data and to create a digital signature value

**signature creation policy**: set of criteria used when creating a digital signature processed or to be processed by the SCASC or the SSASC

**signature creation service**: TSP service implementing a signature creation application and / or a server signing application

**signature creation service provider**: a service provider offering a signature creation service

## 3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CSC Cloud Signature Consortium

DTBS Data To Be Signed

DTBSF Data To Be Signed (Formatted)

DTBSR Data To Be Signed Representation

DSV Digital Signature Value

SAM Signature Activation Module

SCA Signature Creation Application

SCASC Signature Creation Application Service Component

SCASP Signature Creation Application Service Provider

SCDev Signature Creation Device

SCS Signature Creation Service

SCSP Signature Creation Service Provider

SD Signer’s Document

SDO Signed Data Object

SDOC Signed Data Object Composer

SDR Signer’s Document Representation

SSA Server Signing Application

SSASC Server Signing Application Service Component

SSASP Server Signing Application Service Provider

# 4 Signature creation process, service decomposition

## 4.1 Signature creation process steps and data elements

Figure 1 below (derived from ETSI EN 319 102-1 [4] section 4.2.1) shows the various steps and the related data elements for a signature creation process. For remote signature creation, different steps of this process are carried out according to a decomposition into several components, which shall have access to or make available the corresponding data elements. The process illustrated in the figure below is limited to the buildings blocks and information needed for creating a signature without taking in consideration issues such as signer authentication, authorization to the signing key usage, signing certificate availability. The signature activation module in the tamper protected area is needed only when the Signature Creation Service (SCS) complies to the sole control assurance level 2 (SCAL2) signature activation mechanism.



Figure 1: Process Steps and Data Elements in Signature Creation

## 4.2 Service main components and interfaces

The above process points out scenarios where the AdES and/or Digital Signature Value (DSV) are created using a signing key held within a cryptographic security module named Signature Creation Device (SCDev) operated by a Signature Creation Service Provider (SCSP).

Based on the different types of data managed in requests and responses, two main components can be identified in the above schema providing different interfaces for signing management: the Server Signing Application Service Component (SSASC) and the Signature Creation Application Service Component (SCASC) defined below.

The SSASC is the component supporting digital signature values creation. The SSASC is able to interact with the SCDev holding the signer’s private key. When the SSASC uses the SCDev, the authorized signer is able to control the signing key with a certain level of confidence.

The SSASC interface has the Data To Be Signed Representation (DTBSR) and other parameters as main input and the digital signature value as main output.

The SCASC is the component supporting AdES digital signature creation and carrying out several specific parts of the signature creation process. The SCASC is able to interact with the SSASC for requesting digital signature values creation.

The SCASC interface has the document(s) to be signed (SD) or its (their) representation (SDR) and other parameters as main input and the signed document(s) or the digital signature(s) as main output.

By SCS we denote a TSP service implementing a signature creation application (SCA) and / or a server signing application (SSA).

Some variants of these interfaces are possible depending on the functional split between the SCS and the signer’s local system.

The following clauses specify main information objects and processes in SCASC and SSASC.

## 4.3 Signature Creation Application

### 4.3.1 Signer’s document and hashing

The signature creation process starts with the signer’s document (SD), which is to be signed. The SD is represented (SDR) by a hash value in the Data To Be Signed (DTBS). The following observations are made:

* The creation of the SDR (the hashing) can be done where the SD is stored or by the SCASC. In the former case, the SDR shall be transferred to the SCASC while in the latter case, the SD shall be transferred to the SCASC.
* The SD is part of the final Signed Data Object (SDO). Part of the Signed Data Object Composer (SDOC) function (building of the final AdES format) is to relate the digital signature value to the SD.

An important design decision for remote signature creation services is where the SD, and thus its content, needs to be available. Making available only the SDR limits threats to confidentiality but may result in limitations in the functionality of the remote signature creation solution (i.e. when enveloping or enveloped signatures need to be created, or when visual representation of the signature needs to be included).

### 4.3.2 DTBS composition and formatting

In the two processes of DTBS composition and formatting, which in the context of this specification are seen together, the SDR (hash of the document to be signed) and hashes of all signed attributes are assembled into the Data To Be Signed Formatted (DTBSF). In addition to a certificate identifier (hash of signing certificate, possibly also of further certificates in a certificate chain) as indicated in the figure, further signed attributes are required or allowed by the ETSI standard signature formats (C/X/PAdES). For example all baseline CAdES and XAdES variants require the presence of the signed attributes “document type” (of SD) and “claimed signing time”.

The signed attributes, whose presence is needed in the DTBS, or their hash values are available to the SCASC when the DTBSF is created by the SCASC.

### 4.3.3 DTBS preparation

This step consists of creating the DTBSR from DTBSF. The SCASC prepares the entire DTBSF, calculates the hash, and sends the hash value (DTBSR) as input to a SSASC.

### 4.3.4 SDO composer

As the final step, the SDO (the AdES format) is constructed. This consists of combining the digital signature value with other parameters into the requested format. Depending on the format, the digital signature made available for the SD is named:

* Enveloped: The signature is added to the SD (e.g. PAdES signature),
* Enveloping: The signature wraps the SD (e.g. certain CAdES formats),
* Detached: The signature is a separate object linked to the SD..

The SDO composing is done by a separate service instance or integrated with other functions in the SCASC.

## 4.4 Server Signing Application

### 4.4.1 Signature creation

The purpose of the signature creation process is to take DTBSR and create a digital signature value under the control of the signer. In the context of this specification, the creation of the digital signature value is managed by a SSASC that uses a signing key, held within a cryptographic security module (SCDev), that the signatory can activate by means of a secure authorization and activation process.

#### 4.4.1.1 Signature activation

The SSASC uses a remote SCDev in order to generate, maintain and use the signing keys under the control of their authorized signers. The authorized signer remotely controls the signing key with a certain level of confidence eventually by means of the Signature Activation Module (SAM) that is a software component using the Signature Activation Data (SAD) to authenticate the signer and gain its authorisation to activate its signing key for the purpose of signing the DTBSR. This process ensures confidence that the signing keys are under the control of the signer.

Two different levels of confidence of the control of the signing key, as defined in CEN EN 419 241-1 [6], are considered in the present document:

* Sole control assurance level 1 (SCAL1):
	+ The signing keys are used, with a low level of confidence, under the sole control of the signer.
	+ The authorised signer’s use of its key for signing is enforced by the SSASC which authenticates the signer. The activation of the signing key can remain for a given period and/or for a given number of signatures.

It is not expected that such implementations would meet the requirements of sole control as it would be expected for a stand-alone QSCD as defined in the eIDAS [i.1] Regulation.

* Sole control assurance level 2 (SCAL2):
	+ The signing keys are used, with a high level of confidence, under the sole control of the signer.
	+ The authorised signer’s use of its key for signing is enforced by the Signature Activation Module by means of Signature Activation Data provided, by the signer, using a Signature Activation Protocol, in order to enable the use of the corresponding signing key to sign specific documents.

#### 4.4.1.2 Signature creation by SCDev

Signature creation process is performed by SCDev. In the context of this specification, only architectures where the signature creation process is carried out by a remote SCDev are considered. According to the above sole control assurance levels the signing key performs the digital signature value creation after a successful signer authentication by the SSASC (SCAL1) or after a successful SAD verification by the SAM.

# 5 Architectures for server signing

## 5.1 Overview

This clause describes the architectures of systems supporting remote server signing pointing out the fundamental interactions of SCASC and SSASC with the other parties involved in remote signature processes and taking in consideration the level of confidence of the control of the signing keys.

A typical schema for representing systems supporting remote server signing includes a SCASC that is connected to a SSASC hosting the remote Signature Creation Device (providing QES and/or AdES). Services such as CA/RA, OCSP and CRLs, timestamping and authentication and/or authorization servers are considered external to the schema.

Two main scenarios are considered:

* the signature client application sends to the Signature Creation Service Provider requests to generate one or more AdESs;
* the signature client application sends to the Signature Creation Service Provider requests to generate one or more Digital Signature Values and then completes the creation of the AdES structure.

The defined protocols allow both SCASC and SSASC to implement batch signing for documents, hashes of documents and DTBSRs. In such processes of batch signatures of documents the signer is not requested to explicitly approve each document signature.

## 5.2 Introduction to architectures

Two different architectures will be presented in the following clauses in which SCASC and SSASC implement different authentication and authorization mechanisms according to the level of confidence of the control of the signing keys, taking in consideration that the TSP managing SCASC and/or SSASC can delegate the authentication and authorization processes to an external party (e.g. to an identity and/or an authentication provider).

The architectures include two main environments: the signer’s environment and the TSP protected environment.

NOTE: in case of a SCS complying to SCAL2, the TSP protected environment includes a tamper protected device (e.g. cryptographic module conforming to CEN EN 419 221 series).

The signer’s environment is local to the signer and its protection is under responsibility of the signer. The TSP protected environment is operated according to the security policy chosen by the TSP for securing the operations of the SCS and can store in a protected form, signing key(s) and link(s) between key(s) and signer(s).

In the following models the dotted lines are used to represent data streams that are not part of the protocols defined in the document and are present only for the purpose of specifying any possible features of the illustrated services.

## 5.3 Remote signing services with SCAL1

In this model the signing key confidentiality and integrity are ensured by the SCDev that can be activated by the SSASC. Such activation can remain for a given period and/or for a given number of signatures.

The signer can be authenticated by a SCASC or a SSASC depending on whether the SCSP is hosting a SCASC and/or a SSASC. If the SCSP is hosting only the SSASC then the SCASC can be provided i.e. directly in the signer environment or by a different SCSP. SCASC and SSASC can delegate signer authentication to an external party. When the signer authentication succeeds, the corresponding signing key may be used for signature operations on behalf of the signer within a certain time frame and/or a certain amount of signature operations thus allowing the management of bulk/batch signature operations.



## 5.4 Remote signing services with SCAL2

In this model a third main environment is defined in addition to signer’s and TSP protected environments: the tamper protected environment. It is operated within the TSP protected environment, protects the use of signing keys and enforces signature activation to be under the signer control with a greater degree of confidence than in the previous model.

In this model the signing key confidentiality and integrity are ensured by the SAM that can be activated by the SSASC. The SAM verifies the SAD in order to be able to authorize the requested signature operation. The SAM can delegate signer authentication to an external party. When the SAD validation succeeds, the corresponding signing key may be used for signature operations on behalf of the signer.



## 5.5 Security, integrity and confidentiality

ETSI TS 119 431-2 provides requirements for TSPs operating a SCASC supporting AdES digital signature creation.

ETSI TS 119 431-1 provides requirements for TSPs operating a SSASC supporting digital signature value creation.

# 6 Protocol profiles specification

## 6.1 Introduction

The present document specifies the semantics of a protocol for requesting the creation of digital signatures to a remote server and for receiving the related response.

For the aforementioned semantics the present document specifies two bindings, each one in a different format (XML and JSON).

As far as it has been possible and suitable, the profiles specified in the present document have taken as starting point a number of OASIS DSS and DSS-X Technical Committees’ specifications, namely "OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 – Commitee Specification Draft 01" [2], "OASIS Standard: Advanced Electronic Signature Profiles of the OASIS Digital Signature Service Version 2.0 – Working Draft 02" [3] and the Cloud Signature Consortium specification "Architectures, and protocols for remote signature applications Version 1.0.3.0" [1].

The rest of the document is organized as follows:

* Sub-clauses 6.2 and 6.3 provide general remarks on the XML and JSON protocols relying on OASIS DSS and DSS-X Technical Committees’ and CSC’s protocols.
* Clause 7 specifies the components of the protocols for remote digital signature creation (XML and JSON) used by SSASC and SCASC.
* Clause 8 specifies the profiles implemented by SSASC and SCASC by using the components defined in clause 7.

For each component of the aforementioned protocols, the present document:

* Defines requirements for the semantics of the component (i.e. its mandatory contents, its optional contents, etc). These requirements are defined in clauses "Component semantics".
* Defines requirements for the XML component of the XML protocol relying on OASIS DSS and DSS-X Technical Committees’ protocols, which is able to fulfil the semantic requirements already defined. These requirements are defined in clauses named "XML related component".
* Defines requirements for the JSON component of the JSON protocol relying on CSC’s protocols, which is able to fulfil the semantic requirements already defined. These requirements are defined in clauses named "JSON related component".

## 6.2 OASIS DSSX TC XML related protocol

The structures described in this specification are contained in the schema files [DSS\_Core\_XSD], [AdES\_XSD], [ASYN\_XSD], [SIG\_POL\_XSD], and the xml schema file [ETSI\_SIG\_CORE\_XSD]. The new elements and types defined in that schema are defined within the XML namespace whose URI value is: [http://uri.etsi.org/19432/v1.1.1#](http://uri.etsi.org/19432/v1.1.1)

EDITORIAL NOTE: the schema files above are not yet available

Table 1 shows the URI values of other XML namespaces and their corresponding prefixes used in the aforementioned schema file and within the present document.

|  |  |
| --- | --- |
| **URI value of the XML Namespace**  | **Prefix**  |
| http://uri.etsi.org/19432/v1.1.1# | etsisig |
| urn:oasis:names:tc:dss:1.0:core:schema | dss |
| urn:oasis:names:tc:dss:1.0:profiles:AdES:schema# | dssades |
| urn:oasis:names:tc:dss:1.0:profiles:asynchronousprocessing:1.0 | dssasyn |
| urn:oasis:names:tc:dss:1.0:profiles: VisibleSignatures:schema# | dssvs |
| http://www.w3.org/2000/09/xmldsig# | ds  |
| <http://uri.etsi.org/01903/v1.3.2> | xades |
| <http://uri.etsi.org/01903/v1.4.1> | xadesv141 |
| urn:oasis:names:tc:SAML:2.0:assertion | saml2 |
| http://docs.oasis-open.org/dss-x/ns/base | dsb |
| http://docs.oasis-open.org/dss-x/ns/core | dss2 |

The present document will reference components in the aforementioned documents and further profiles some of them.

In the absence of any further requirement defined in the present document, the requirements defined in the aforementioned documents for each element present in this profile, shall apply.

In addition the present document specifies elements that are not specified in the aforementioned documents. For these elements, the present document also defines the processing model that the server shall implement. This processing model is specified below the indication *Processing model* within each clause that specifies one of these elements.

## 6.3 CSC JSON related protocol

The structures described in this specification are contained in the schema files [ETSI\_SIG\_CORE\_JSCHEMA].

EDITORIAL NOTE: the schema files above are not yet available

In addition the present document specifies elements that are not specified in the CSC specifications document. For these elements the present document also defines the processing model that the server shall implement. This processing model is specified below the indication *Processing model* within each clause that specifies one of these elements.

# 7 Protocol components definitions

## 7.1 Introduction

This clause defines the components of the protocol for remote digital signature creation. The components represent the data that can be passed to or returned by the SCS in order to request and execute the SCS functionalities. Clause 8 defines the profiles implemented by the SCS that make use of the components defined in this clause.

## 7.2 Component for asynchronous/synchronous operation mode selection

### 7.2.1 Component semantics

With the term synchronous operation mode it is intended that the client application after sending any request to the SCS will wait for it to finish before moving on to another task. In such operation mode the SCS will perform the requested operation and return the corresponding outcomes to the client application. With the term asynchronous operation mode it is intended that the client application, after sending any request to the SCS, can move on to another task before the requested operation finishes. In such operation mode the SCS will accept the request and return a notification informing the client application about the acceptance of its request. The client will be able to request the corresponding outcomes to the SCS later on.

This component shall be used by a client application to request a synchronous or asynchronous operation mode from the SCS.

NOTE 1: when this component is not defined by the client application, the SCS may take one of the following behaviours to perform the requested operation, according to the selected service policy:

* process the request only in synchronous mode
* process the request only in asynchronous mode
* process the request in synchronous mode and decide under particular conditions to process the request in asynchronous mode.

NOTE: when the asynchronous processing is initiated by the SCS it returns a response including:

* a result code informing the client application about the requested operation processing in asynchronous mode (see clause 7.24); and
* a unique response identifier that the client application will use to obtain the outcomes of the requested operation from the SCS (see clause 7.26).

Details on the asynchronous processing mode are described in clause 8.6.

### 7.2.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

* operationMode string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| operationMode | Required | String | The type of operation mode requested to the SCS. It shall take one of the following values:“A”: an asynchronous operation mode is requested“S”: a synchronous operation mode is requested. |

### 7.2.3 XML related component

The element for asynchronous/synchronous operation mode selection shall be etsisig:OperationMode, child element of the dss2:OptionalInputs element.

The OperationMode element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="OperationMode" type="etsisig:OperationModeType" />

<xs:simpleType name="OperationModeType">
 <xs:restriction base="xs:string">
 <xs:enumeration value="Synchronous" />
 <xs:enumeration value="Asynchronous" />
 </xs:restriction>
</xs:simpleType>

## 7.3 Component for identification of the request

### 7.3.1 Component semantics

This component contains a string value included by the client application and is used in order to correlate requests with subsequent corresponding responses or in order to poll asynchronous requests outcomes.

### 7.3.2 JSON related component

The component for the identification of the request shall be represented by the following

* requestID string type parameter.

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| requestID | Required | String | Data from the client application generally used to handle a signature transaction identifier. |

### 7.3.3 XML related component

The element for correlating requests with subsequent responses shall be the RequestID attribute of the dss2:SignRequest root element and the RequestId attribute of the dss2:SignResponse root element.

## 7.4 Component for credential authorization

### 7.4.1 Component semantics

This component shall contain the information needed to authorize the use of the signing key.

When the sole control assurance level 1 (SCAL1) is implemented this component contains the data used to control, with a low level of confidence, that a given signature operation is performed on behalf of the signer under sole control of the signer.

When the sole control assurance level 2 (SCAL2) is implemented this component contains the data used to control, with a high level of confidence, that a given signature operation is performed on behalf of the signer under sole control of the signer.

### 7.4.2 JSON related component

The component for submitting the credential authorization shall be represented by the following

* SAD string type parameter;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| SAD | Required | String | Authentication data used to authorize the use of the signing key |

The “SAD” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.4.3 XML related component

The element for credential authorization shall be etsisig:SignatureActivationData, child element of the dss2:OptionalInputs element. If the value to be submitted is not of string type, it shall be encoded into a string value using Base64 encoding.

The SignatureActivationData element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="SignatureActivationData" type="etsisig:SignatureActivationDataType" />

<xs:complexType name="SignatureActivationDataType">
 <xs:simpleContent>
 <xs:extension base="xs:string">
 <xs:attribute name="type" type="xs:string" use="optional" />
 </xs:extension>
 </xs:simpleContent>
</xs:complexType>

## 7.5 Component for defining optional data to be returned

### 7.5.1 Component semantics

This component shall be used by the client to define optional data to be returned from the SCS. This element shall contain a list of element names of the optional data requested. If the SCS doesn’t recognize or can’t handle any optional data to be returned, it shall reject the request and return an error.

### 7.5.2 JSON related component

The component for requesting the optional data to be returned shall be represented by the following

* optionalData array of string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| optionalData | Optional | Array of String | List of element names identifying the data that are requested to be returned. |

### 7.5.3 XML related component

The element for requesting optional data to be returned shall be the optional element etsisig:ReturnOptionalData, contained in the dss2:OptionalInputs element.

The ReturnOptionalData element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="ReturnOptionalData" type="xs:string" />

## 7.6 Component for defining the validity period for asynchronous requests

### 7.6.1 Component semantics

This component shall be used to specify a maximum period of time within which the asynchronous request outcome(s) retrieval shall be completed. The validity period shall be calculated starting from the moment of the request acceptance. The SCS can destroy any outcome(s) after the maximum period of time expiration. Therefore the SCS will not be able to return any outcome(s) after the maximum period of time expiration.

### 7.6.2 JSON related component

The component for requesting the maximum period of time setting shall be represented by the following

* validity\_period number type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| validity\_period | Required | Integer | Maximum period of time, starting from the asynchronous request acceptance and expressed in milliseconds, within which the asynchronous request outcome(s) retrieval shall be completed. |

### 7.6.3 XML related component

The element for defining the validity period for asynchronous requests shall be etsisig:ValidityPeriod, child element of the dss2:OptionalInputs element. The value specifies the maximum period of time, starting from the asynchronous request acceptance and expressed in milliseconds, within which the asynchronous request outcome(s) retrieval shall be completed.

The ValidityPeriod element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="ValidityPeriod" type="xs:int" />

### 7.6.4 Processing model

If the component is present, the SCS shall perform the requested operations and keep available the corresponding outcomes within the period of time specified in this component. If the requested operations are not completed or the client application does not request the available outcomes in the specified time the processing of the request can be terminated.

## 7.7 Component for service authentication

### 7.7.1 Component semantics

This component shall contain information to authenticate the client application to access to the SCASC or the SSASC.

NOTE: the way a client application authenticates to the SCASC or SSASC is out of scope of this document.

### 7.7.2 JSON related component

The authorization component shall be included into the Authorization HTTP header of every call and shall be in “Bearer” format.

### 7.7.3 XML related component

The element for service authentication shall be dss2:ClaimedIdentity, child element of the dss2:OptionalInputs element.

## 7.8 Component for identifying signature credentials

### 7.8.1 Component semantics

This component is used to uniquely identify the signer’s private key and corresponding certificate to be used for signature creation.

### 7.8.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

* credentialID string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| credentialID | Required | String | The identifier associated to the private key and corresponding certificate. |

The “credentialID” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.8.3 XML related component

The element for identifying signature credentials shall be the dss2:KeySelector, child element of dss2:OptionalInputs element.

## 7.9 Component for language and region selection

### 7.9.1 Component semantics

This component shall be used to request a preferred language of the response and shall be specified according to RFC5646 [7].

The service should provide language-specific responses using the requested language. In the case the requested language is not supported then no error shall be raised and the responses shall be produced in a default language that can be specified in the language output parameter.

### 7.9.2 JSON related component

The component for selecting language and region settings shall be represented by the following

* lang string type parameter.

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| Lang | Required | String | The preferred language of the SCS responses, specified according to RFC5646 [7]. The SCS should provide language-specific responses using the specified language. If the specified language is not supported then SCS shall provide responses in its own default language. |

The “Lang” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.9.3 XML related component

The element for language and culture selection shall be dsb:Language, child element of the dss2:OptionalInputs element.

## 7.10 Component for specifying the contents from certificate info to be returned

### 7.10.1 Component semantics

This component is used to specify which contents of the signing certificate chain shall be returned. If this component isn’t defined only the end-entity certificate will be returned.

### 7.10.2 JSON related component

The component for specifying the contents from certificate chain to be returned shall be represented by the following

* returnCertificates string type parameter
* authInfo boolean type parameter
* certInfo boolean type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| returnCertificates | Optional | String | Specifies which certificates from the certificates chain shall be returned in the SCS response.* “none”: no certificate is returned.
* “single”: only the end entity certificate is returned.
* “chain”: the full certificate chain is returned.

The default value is “single”. |
| certInfo | Optional conditional | Boolean | Specifies if the information on the end entity certificate shall be returned as printable strings. The default value is “false”, so if the parameter is omitted then the information will not be returned. This element can carry a value only if the parameter returnCertificates contains a value different from “none”. |
| authInfo | Optional conditional | Boolean | Specifies if the information on the authorization mechanisms supported by this credential shall be returned. The default value is “false”, so if the parameter is omitted then the information will not be returned. This element can carry a value only if the parameter returnCertificates contains a value different from “none”. |

The “returnCertificates” element is not defined in Cloud Signature Consortium Standard [1]. The “certInfo” and “authInfo” elements are defined in Cloud Signature Consortium Standard [1] and are copied here for information.

### 7.10.3 XML related component

The element for list the certificate chain shall be etsisig:ReturnSigningCertificate, child element of the etsisig:OptionalInputs element.

The ReturnSigningCertificate element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="ReturnSigningCertificate" type="etsisig:ReturnSigningCertificateType" />

<xs:complexType name="ReturnSigningCertificateType">
 <xs:attribute name="ReturnCertificates" type="etsisig:ReturnCertificatesType" use="optional" />

 <xs:attribute name="CertificateInfo" type="xs:boolean" use="optional" />

 <xs:attribute name="AuthorizationInfo" type="xs:boolean" use="optional" />

</xs:complexType>

<xs:simpleType name="ReturnCertificatesType">
 <xs:restriction base="xs:string">
 <xs:enumeration value="None" />
 <xs:enumeration value="Single" />
 <xs:enumeration value="Chain" />
 </xs:restriction>
</xs:simpleType>

### 7.10.4 Processing model

If the component is present, the SCS shall check the values passed in the component parameters.

If the parameter indicating which certificates shall be returned is present the SCS will return in its response no certificate, only the signing certificate or the full signing certificate chain if the parameter value is respectively “none”, “single” or “chain”.

If the parameter indicating if signing certificate information shall be returned in the form of strings is valued “true” the SCS will return in its response the signing certificate information as specified in clause 7.22.

If the parameter indicating if signing key and certificate authorization information shall be returned is valued “true” the SCS will return in its response the signing key and certificate authorization information as specified in clause 7.28.

## 7.11 Component for managing digital signatures transactions

### 7.11.1 Component semantics

This component may be used to initiate and manage a sequence of an agreed number of signatures to be regarded as a single unit of work (named transaction in the present clause) between the client and the SCS.

The transaction shall be identified by the component defined in clause 7.3 that will handle a transaction identifier unique within the SCS.

EXAMPLE 1. A working transaction in which a signer needs to put in a document a certain number of PAdES signatures in different parts in order to acknowledge different contents in the document. In such case a new DTBS is computed before every new signature. By using a working transaction to complete all the signatures allows the signer to have a better control of the whole signature process.

EXAMPLE 2. A signer is requesting to sign a very large number of documents, where each document is being included in the request in its entirety and each document is fairly big. This component can be used to prevent the request from being very big. Instead of sending all the documents to the SCASC at the same time, the documents can be sent one at a time, and they would be signed with the same signatory action. The session key would be used to be sure that subsequent documents belong to the same transaction.

NOTE 3: the definition of the authorization protocol is out of scope of the present document.

### 7.11.2 JSON related component

The component for initiating and managing a transaction shall be represented by the following

1. numSignatures number type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| numSignatures | Required | Integer | The number of signatures to be performed in the context of the transaction. The SCS will check this value in the context of the transaction. |

The “numSignatures” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.11.3 XML related component

The element for managing digital signature transaction shall be the etsisig:NumberOfSignatures element, child element of dss2:OptionalInputs.

Transaction is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="NumberOfSignatures" type="xs:int"/>

## 7.12 Component for service policy selection

### 7.12.1 Component semantics

This component shall contain a non-ambiguous identifier of the service policy under which the server shall perform the requested operation.

### 7.12.2 JSON related component

The component for specifying the identifier of the service policy under which the server shall perform the requested operation shall be represented by the following

* policy string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| policy | Required | String | The element that identifies a particular service policy associated with the SCS. The policy element may be used to select a specific service policy if a SCS supports multiple policies, or as a sanity-check to make sure the SCS implements the service policy the client expects. |

### 7.12.3 XML related component

The element for service policy selection shall be dsb:ServicePolicy, child element of the dss2:OptionalInputs element.

## 7.13 Component for signature creation policy selection

### 7.13.1 Component semantics

This component shall contain information that the service requires to generate signature(s). The information that shall be supplied is the identification of the signature creation policy that shall be used while signing the DTBSR(s).

### 7.13.2 JSON related component

The component for specifying the signature creation policy identification to be used by the SCS shall be represented by the following

* signaturePolicyID string type parameter

or, alternatively, the following two parameters:

* signAlgo string type parameter
* signAlgoParams string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| signaturePolicyID | Required Conditional | String | The element that identifies a particular signature creation policy associated with the SCS. It shall have the value of a unique identifier of the signature creation policy as an URI. If the identifier of the signature creation policy is an OID, then the value of this element shall be an URN indicating the value of the aforementioned OID as specified in RFC 3061 [i.15]. This element can carry a value only if the signAlgo parameter is not specified. |
| signAlgo | Required Conditional | String | The element specifies the algorithm OID used for signing. This element can carry a value only if the signaturePolicyID parameter is not specified. |
| signAlgoParams | Optional Conditional | String | The element specifies the Base64-encoded of DER-encoded ASN.1 signature parameters. This element can carry a value only if required and/or allowed by the signature algorithm like, for example, the RSA-PSS cryptographic signature scheme (RFC 8017 [i.16]). |

The “signaturePolicyID” element is not defined in Cloud Signature Consortium Standard [1]. The “signAlgo” and “signAlgoParams” elements are defined in Cloud Signature Consortium Standard [1] and are copied here for information.

### 7.13.3 XML related component

The element for signature algorithm selection shall be dss2:SignatureAlgorithm, child element of the dss2:OptionalInputs element.

EDITORIAL NOTE: DSS2 does not currently support including parameters to the signature algorithm (e.g. RSA-PSS). We have asked OASIS how this can be achieved.

## 7.14 Component for optional signature attributes/properties selection

### 7.14.1 Component semantics

The request can include this component if there are certain signed or unsigned attributes/properties that the SCASC is requested to include in the signature. The client application can pass to the SCASC a particular value to be used for each attribute/property or leave the value up to the SCASC to be determined. If no value is passed and the SCASC cannot calculate it the corresponding attribute/property shall not be included in the signature. The SCASC can include additional attributes/properties in the signature, even if these ones aren't explicitly requested by the client application (i.e. because such attributes/properties are mandated by the signature profiles).

### 7.14.2 JSON related component

The component for the selection of the attributes/properties to be included in the signature shall be represented by the following two arrays:

1. signed\_props containing the list of attributes to be added to the signatures’s signed attributes
2. unsigned\_props containing the list of attributes to be added to the signatures’s unsigned attributes

specified according to the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| signed\_props | Required conditional | Array of Attribute Object | List of signed attributes. The attributes that can be included depend from the signature format. |
| unsigned\_props | Required conditional | Array of Attribute Object | List of unsigned attributes. The attributes that can be included depend from the signature format. |

The ‘Attribute Object’ is a JSON Object composed by the following attributes:

1. attribute\_name string type parameter
2. attribute\_value string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| attribute\_name | Required | Array of String | Names or OIDs of the attributes/properties to be included in the signature. The attributes and/or properties names defined in the table in clause 6.3 of ETSI EN 319 122-1/132-1/142-1 [i.5], [i.7], [i.9] documents should be used. |
| attribute\_value | Conditional | Array of String | Depending on the attributes/properties specified in the attribute\_name parameter, this parameter contains values to be used for the corresponding attributes/properties to be included in the signature. When some element of this array is not defined the SCASC shall calculate it, if needed.+ |

### 7.14.3 XML related component

The element for optional signature attributes/properties shall be dss2:Properties, child element to the dss2:OptionalInputs element.

## 7.15 Component for protocol identifier

### 7.15.1 Component semantics

This component can be used by the client application to communicate to the server which protocol is being used to communicate to the server itself. The value of this component shall be an identifier notifying that the request has been built using the profile defined by the present document.

The identifier for the profile defined by the present document shall be:

http://uri.etsi.org/19432/v1.1.1/creationprofile#

The request may contain additional components whose values are identifiers of other profiles that have also been used for building the request.

This component shall be used to notify the server that the client expects processing of the request according to the profile defined by the present document.

### 7.15.2 JSON related component

The component for notifying to SCS the information of the protocol that is being used by the client shall be represented by the following

* profile string type parameter;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| profile | Required | String | String that identifies the protocol being used by the client to communicate to SCS. The value should be the one specified in clause 7.15.1. |

### 7.15.3 XML related component

The element used to notify the server of the profile shall be the dsb:Profile element of the dss2:SignRequest. Exactly one such element shall hold the value of the profile identifier as specified in 7.15.1.

## 7.16 Component for requesting specific signature formats

### 7.16.1 Component semantics

This component is used to request a specific signature format. The signature format and conformance level shall be the same for each document that will be signed within the identified signature request.

The conformance levels of the "baseline profiles" standards, defined in [i.5], [i.7], [i.9], should be used.

|  |  |  |
| --- | --- | --- |
| **Signature format** | **Conformance level** | **URI** |
| CAdES/PAdES/XAdES | AdES-B-B | http://www.etsi.org/ades/191x2/level/baseline/B-B# |
| CAdES/PAdES/XAdES | AdES-B-T | http://www.etsi.org/ades/191x2/level/baseline/B-T# |
| CAdES/PAdES/XAdES | AdES-B-LT | http://www.etsi.org/ades/191x2/level/baseline/B-LT# |
| CAdES/PAdES/XAdES | AdES-B-LTA | http://www.etsi.org/ades/191x2/level/baseline/B-LTA#  |
| CAdES/PAdES/XAdES | AdES-B | http://www.etsi.org/ades/etsits/level/baseline/B-B# |
| CAdES/PAdES/XAdES | AdES-T | http://www.etsi.org/ades/etsits/level/baseline/B-T# |
| CAdES/PAdES/XAdES | AdES-LT | http://www.etsi.org/ades/etsits/level/baseline/B-LT# |
| CAdES/PAdES/XAdES | AdES-LTA | http://www.etsi.org/ades/etsits/level/baseline/B-LTA# |

According to the type of signature selected a client could also specify the following signature propreties

|  |  |
| --- | --- |
| Signature format | Signed envelope property |
| CAdES | DetachedAttachedParallel |
| PAdES | CertificationRevision |
| XAdES | EnvelopedEnvelopingDetached |

### 7.16.2 JSON related component

The component for requesting a specific signature format shall be represented by the following parameters

1. signature\_format string type parameter;
2. conformance\_level string type parameter;
3. signed\_envelope\_property string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| signature\_format | Required | String | The required signature format:* “C” shall be used to request the creation of a CAdES signature;
* “X” shall be used to request the creation of a XAdES signature;
* “P” shall be used to request the creation of a PAdES signature.
 |
| conformance\_level | Optional | String | * “AdES-B-B” shall be used to request the creation of a baseline 191x2 level B signature
* “AdES-B-T” shall be used to request the creation of a baseline 191x2 level T signature
* “AdES-B-LT” shall be used to request the creation of a baseline 191x2 level LT signature;
* “AdES-B-LTA” shall be used to request the creation of a baseline etsits level LTA signature.
* “AdES-B” shall be used to request the creation of a baseline etsits level B signature
* “AdES-T” shall be used to request the creation of a baseline etsits level T signature
* “AdES-LT” shall be used to request the creation of a baseline etsits level LT signature;
* “AdES-LTA” shall be used to request the creation of a baseline etsits level LTA signature.

The parameter is optional. The default baseline level is AdES-B-B in case it is omitted.If a timestamp is needed its request and inclusion is managed by the SCS according to SCS configuration and policies.The required property concerning the signed envelope. |
| signed\_envelope\_property | Optional Conditional | String |  |

### 7.16.3 XML related component

The element for requesting a specific signature type shall be the dss2:SignatureType, child element of the dss2:OptionalInputs element, with values as specified in OASIS Standard “Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 – Working Draft 05” [2].

The element for requesting a specific signature conformance level shall be the etsisig:CreateAugmentedSignature, child element of the dss2:OptionalInputs element, with an URI reference identifying the pre-defined level to which the server is requested to augment the signature.

<xs:element name="CreateAugmentedSignature" type="dss2:AugmentSignatureInstructionType"/>

This table lists URIs for the levels specified for AdES signatures in ETSI EN 319 122 [i.5] and [i.6], ETSI EN 319 132 [i.7] and [i.8], ETSI EN 319 142 [i.9] and [i.10], ETSI TS 103 171 [i.18], ETSI TS 103 172 [i.19], and ETSI TS 103 173 [i.17].

|  |  |
| --- | --- |
| **Conformance level** | **URI** |
| AdES-B-B | http://www.etsi.org/ades/191x2/level/baseline/B-B# |
| AdES-B-T | http://www.etsi.org/ades/191x2/level/baseline/B-T# |
| AdES-B-LT | http://www.etsi.org/ades/191x2/level/baseline/B-LT# |
| AdES-B-LTA | http://www.etsi.org/ades/191x2/level/baseline/B-LTA#  |
| AdES-B | http://www.etsi.org/ades/etsits/level/baseline/B-B# |
| AdES-T | http://www.etsi.org/ades/etsits/level/baseline/B-T# |
| AdES-LT | http://www.etsi.org/ades/etsits/level/baseline/B-LT# |
| AdES-LTA | http://www.etsi.org/ades/etsits/level/baseline/B-LTA# |

## 7.17 Component for signer identification

### 7.17.1 Component semantics

This component is used to uniquely identify the signer within the SCS. This component represents the signer identifier associated to the signer identity within the SCS.

A request shall contain acomponent for signer identification

NOTE: this component can be accepted or rejected by the server based on the current service authentication (i. e. in order to avoid to obtain the list of credentials associated to a signer not binded to the current service authentication).

### 7.17.2 JSON related component

The component for specifying the signer identification shall be represented by the following

* SignerIdentity string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| SignerIdentity | Required | String | The identifier associated to the signer identity. |

### 7.17.3 XML related component

The element for signer identification shall be the etsisig:SignerIdentity element, child element of etsisig:OptionalInputs.

The etsisig:SignerIdentity element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="SignerIdentity" type="dss2:ClaimedIdentityType"/>

## 7.18 Component for specifying response URL

### 7.18.1 Component semantics

With this component a client application can communicate to the SCS a destination URL where the client expects to receive a notification when the SCS has completed the requested operation.

This component shall accept absolute URL.

### 7.18.2 JSON related component

The component for specifying the response URL (s) where the notification of the requested operation completion is to be received by the client application shall be represented by the following

* response\_uri string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| response\_uri | Required | String | The element shall have the value of one location where the SCS will notify the signature creation operation completion, as an URI value. |

### 7.18.3 XML related component

The element for specifying response URL shall be ResponseURL, child element to the dss2:OptionalInputs element.

The ResponseURL element shall be defined as in XML schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2 and is copied below for information.

<xs:element name="ResponseURL" type="xs:anyURI"/>

### 7.18.4 Processing model

If this component is present and the SCS performs the requested operation in asynchronous operation mode, after the completion of such operation the SCS will invoke the location specified by this component providing as input parameter the component specified in clause 7.26 that the client application will include in the subsequent request polling the pending signature results.

## 7.19 Component for submitting document(s) or hash(es) to be signed

### 7.19.1 Component semantics

This component shall be used to pass to the SCS the list of document(s) or the list of hash(es) for which the signature(s) generation is requested.

Document(s) and hash(es) shall be defined using two different containers that are mutually exclusive in this component. Therefore an instance of this component shall not contain both a list of documents and a list of hashes,

The two containers shall include the following information:

1. The base-64 encoded content(s) of the document(s) to be signed.
2. The base-64 encoded hash(es) of the document(s) to be signed and the digest algorithm used to calculate such hash(es).

A list of hashing algorithms recommended in ETSI TS 119 312 [i.12] is listed in the following table:

|  |  |  |
| --- | --- | --- |
| Hash algorithm OID | Hash algorithm name | Hash algorithm URI |
| 2.16.840.1.101.3.4.2.1 | SHA-256 | http://www.w3.org/2001/04/xmlenc#sha256 |
| 2.16.840.1.101.3.4.2.2 | SHA-384 | http://www.w3.org/2001/04/xmldsig-more#sha384 |
| 2.16.840.1.101.3.4.2.3  | SHA-512 | http://www.w3.org/2001/04/xmlenc#sha512 |
| 2.16.840.1.101.3.4.2.8 | SHA3-256 | http://www.w3.org/2007/05/xmldsig-more#sha3-256 |
| 2.16.840.1.101.3.4.2.9 | SHA3-384 | http://www.w3.org/2007/05/xmldsig-more#sha3-384 |
| 2.16.840.1.101.3.4.2.10 | SHA3-512 | http://www.w3.org/2007/05/xmldsig-more#sha3-512 |

### 7.19.2 JSON related component

The component for submitting document(s) or hash(es) to be signed shall be represented by the following (mutually exclusive) parameters

* documents: array of string type parameter;
* documentDigests: JSON objects containing hash(es) to be signed and the digest algorithm OID used to calculate the hash(es) specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| documents | Required Conditional | Array of String | Base64-encoded document(s) content(s), to be signed. This parameter shall not be used if the parameter documentDigests is passed. |
| documentDigests | Required Conditional | JSONObject | JSON Object containing hash(es) to be signed and digest algorithm OID used to calculate di hash(es). This parameter shall not be used if the parameter documents is passed. |

The documentDigest element shall be a JSONObject containing the following parameters:

* hashes array of string type parameter
* hashAlgorithmOID string type parameter

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| hashes | Required | Array of String | Base64-encoded document(s) hash(es), to be signed. |
| hashAlgorithmOID | Required | String | Hashing algorithm OID used to calculate document hash(es). |

### 7.19.3 XML related component

The element for submitting document(s) to be signed shall be dss2:Document, child element of dss2:InputDocuments element. The MimeType attribute of dss2:Base64Data, child element of dss2:Document, shall be set.

The element for submitting hash(es) shall be dss2:DocumentHash, child element of dss2:InputDocuments element. The client shall incorporate the base64 encoding of this digest value into ds:DigestInfos and the hash algorithm as an URI into ds:DigestMethod, child element of the dss2:DocumentHash element.

## 7.20 Component for returning service information

### 7.20.1 Component semantic

This element shall contain:

1. a general description of the service;
2. a general name identifying the service;
3. a pointer to the logo of the service;
4. the country where the service is operating;
5. information concerning the protocol supported, i.e. the URI http://uri.etsi.org/19432/v1.1.1/creationprofile# specifies the support of the profile specified in the present document;
6. a list of the versions of the protocol supported;
7. a list of supported languages;
8. a list of signature policies implemented by the SCS;
9. a list of service policies implemented by the SCS;
10. a list of accepted operation modes;
11. a list of supported authentication modes;
12. a list of signature formats, conformance levels and signed envelope properties;
13. a list of names of all the API methods implemented and supported by the SCS.

Some of the above items may be empty or absent indicating a feature not supported by the SCS.

### 7.20.2 JSON related component

The component for returning service information shall be represented by the following

1. description string type result;
2. name string type result;
3. logo string type result;
4. region string type result;
5. protocol string type result;
6. versions array of string type result;
7. lang array of string type result;
8. signaturePolicies an array of strings result;
9. servicePolicies an array of strings result;
10. operationModes an array of strings result;
11. authType an array of strings result;
12. signatureFormats an array of signature format, conformance level and signed envelope property values defining which values can be passed in the component defined in clause 7.16
13. methods an array of strings result.

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Value | Description |
| description | Required | String | A short description of the service. |
| name | Required | String | The name of the service. |
| logo | Required | String | The URI of the image file containing the logo of the Service. The image shall be in either JPEG or PNG format and not larger than 256x256 pixels. |
| region | Required | String | The ISO 3166-1 alpha-2 code where the service is operating. |
| protocol | Required | String | The name of the protocol supported by the SCS. |
| versions | Required | Array of String | The versions of the protocol specifications supported by the SCS. The format of the strings is Major.Minor.x. |
| lang | Required | Array of String | List of the supported languages in the service responses, specified according to RFC5646. |
| signaturePolicies | Required | Array of String | List of the supported signature policies names. |
| servicePolicies | Required | Array of String | List of the supported service policies names. |
| operationModes | Required | Array of String | List of the supported operation modes. |
| authType | Required | Array of String | List of the authentication mechanisms supported by the service for API methods access. |
| signatureFormats | Required | Array of JSONObject | List of JSONObject specifying the signature formats supported by the SCS and containing the following parameters:* signature\_format string type parameter;
* conformance\_level string type parameter;
* signed\_envelope\_property string type parameter;

as specified in clause 7.16.2. |
| methods | Required | Array of String | List of the supported API methods names. |

The “signatureFormats” element is not defined in Cloud Signature Consortium Standard [1]. The other elements are defined in Cloud Signature Consortium Standard [1] and are copied here for information.

### 7.20.3 XML related component

The element for returning service information shall be etsisig:ServiceInformation child element of the etsisig:OptionalOutputs.

The ServiceInformation element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="ServiceInformation" type="ServiceInformationType"/>

<xs:complexType name="ServiceInformationType">
 <xs:sequence>
 <xs:element name="Description" type="xs:string"/>
 <xs:element name="Version" type="xs:string"/>
 <xs:element name="Logo" type="xs:anyURI"/>
 <xs:element name="Region" type="xs:string"/>
 <xs:element name="SupportedProtocol" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedLanguage" type="xs:language" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedSignaturePolicy" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedServicePolicy" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedOperationMode" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedAuthMode" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="SupportedMethod" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
 </xs:complexType>

## 7.21 Component for returning signed documents or signatures

### 7.21.1 Component semantics

This component shall be used to return the requested signatures. The protocol shall allow returning the signatures in two different containers according to the following rules:

1. If the signature is enveloped within the signed document, it shall be included in a specific container identified as the container for the signed document.
2. If the signature is not enveloped then it shall be included in a specific container identified as the container that encloses the signature.

### 7.21.2 JSON related component

The component for returning the requested signed documents or signatures shall be represented by the following

1. DocumentWithSignature: JSON array of string type result;
2. SignatureObject: JSON array of string type result;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| DocumentWithSignature | Required Conditional | Array of String | Base64-encoded signatures enveloped within the documents. This element can carry a value only if the client application requested the creation of signature(s) enveloped within the signed document(s). |
| SignatureObject | Required Conditional | Array of String | Base64-encoded signatures detached from the the documents. This element can carry a value only if the client application requested the creation of not enveloped signature(s). |

### 7.21.3 XML related component

The element for returning signatures not enveloped within the documents shall be dsb:Base64Signature, child element of the dss2:SignatureObject.

In the case when returning signatures enveloped within the documents the following two elements shall be used:

1. dss2:DocumentWithSignature, child element of the dss2:OptionalOutputs, containing the document that envelopes the signature
2. dss2:SignaturePtr, child element of the dss2:SignatureObject, that shall point to the former dss2:DocumentWithSignature.

## 7.22 Component for returning signing certificate information

### 7.22.1 Component semantics

This component is used for returning signing chain/certificate/credential information.

This component shall contain information about the signing credential and the signing certificate/chain used or to be used in the operation of DSV(s) creation.

This component shall contain the following data:

1. the signing X.509 certificate/chain;
2. information about the signing key:
	1. status
	2. algo
	3. len
	4. curve
3. the signer certificate attribute details:
	1. status
	2. validFrom
	3. validTo
	4. issuerDN
	5. serialNumber
	6. subjectDN
4. credential supporting multiple signatures creation with a single authorization request specification

The inclusion of this component in the response is required by the client application using the “Component for defining optional data to be returned” defined in clause 7.5. The information about signing X.509 certificate/chain to be included in this component are specified by the client application using the “Component for specifying the contents from certificate info to be returned” defined in clause 7.10.

### 7.22.2 JSON related component

The component for returning signing certificate/credential information shall be represented by the following

1. cert/certificates: array of string type result;
2. key/status: string type result;
3. key/algo: array of string type result;
4. key/len: integer type result;
5. key/curve: string type result;
6. cert/status: string type result;
7. cert/validFrom: string type result;
8. cert/validTo: string type result;
9. cert/IssuerDN: string type result;
10. cert/serialNumber: string type result;
11. cert/subjectDN: string type result;
12. multisign: boolean type result;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| cert/certificates | Required Conditional | Array of String | Contains one or more Base64-encoded X.509v3 certificates from the certificate chain. If the certificates parameter defined in 7.10.1.2 clause is “chain”, the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the certificates parameter is “single”, only the end entity certificate shall be returned. If the certificates parameter is “none”, this parameter shall not be returned. |
| key/status | Required | String | enabled | disabledThe status of enablement of the signing key of the credential:* “enabled”: the signing key is enabled and can be used for signing.
* “disabled”: the signing key is disabled and cannot be used for signing. This may occur when the owner has disabled it or when the SCS has detected that the associated certificate is expired or revoked.
 |
| key/algo | Required | String | The list of OIDs of the supported key algorithms. For example: 1.2.840.113549.1.1.1 = RSA encryption, 1.2.840.10045.4.3.2 = ECDSA with SHA256  |
| key/len | Required | Number | The length of the cryptographic key in bits.  |
| key/curve | Required Conditional | String | The OID of the ECDSA curve. The value shall only be returned if keyAlgo is based on ECDSA.  |
| cert/status | Optional | String | valid | expired | revoked | suspendedThe status of validity of the end entity certificate. The value is optional. The SCS shall only return a value that is accurate and consistent with the actual validity status of the certificate at the time the response is generated. |
| cert/validFrom | Required | String | The validity start date from the X.509v3 signing certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. “YYYYMMDDHHMMSSZ”). This parameter shall be returned when certInfo defined in clause 7.10 is “true”. |
| cert/validTo | Required | String | The validity end date from the X.509v3 signing certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. “YYYYMMDDHHMMSSZ”). This parameter shall be returned when certInfo defined in clause 7.10 is “true”. |
| cert/IssuerDN | Required Conditional | String | The Issuer Subject Distinguished Name from the X.509v3 end entity certificate in printable string format, UTF-8-encoded according to RFC 2253. This parameter shall be returned when certInfo defined in clause 7.10 is “true”.  |
| cert/serialNumber  | Required Conditional | String | The Serial Number from the X.509v3 certificate in hex encoded format. This parameter shall be returned when certInfo defined in clause 7.10 is “true”  |
| cert/subjectDN | Required Conditional | String | The Distinguished Name from the X.509v3 certificate in printable string format, UTF-8-encoded according to RFC 2253. This parameter shall be returned when certInfo defined in clause 7.10 is “true”  |
| multisign | Required Conditional | Boolean | Specifies if the credential supports multiple signatures to be created with a single authorization request  |

The above elements are defined in Cloud Signature Consortium Standard [1] and are copied here for information.

### 7.22.3 XML related component

The element for returning signing certificate/credential/chain shall be ds:KeyInfo, child element of the dss2:OptionalOutputs root element.

The element for returning signing certificate/credential/chain information shall be ds:X509Data, child element of the ds:KeyInfo element. To include additional attributes as listed in point 3 in the semantics section above, the X509Details element shall be used in the any element of ds:KeyInfo.

The X509Details element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="X509Details" type="X509DetailsType"/>

<xs:element name="Status" type="etsisig:CertificateStatusType" />
<xs:element name="NotBefore" type="xs:dateTime" />
<xs:element name="NotAfter" type="xs:dateTime" />

<xs:complexType name="X509DetailsType">
 <xs:sequence>
 <xs:element ref="etsisig:Status" />
 <xs:element ref="etsisig:NotBefore />
 <xs:element ref="etsisig:NotAfter" />
 </xs:sequence>
 </xs:complexType>

<xs:simpleType name="CertificateStatusType">
 <xs:restriction base="xs:string">
 <xs:enumeration value="Valid"/>
 <xs:enumeration value="Expired"/>
 <xs:enumeration value="Revoked"/>
 <xs:enumeration value="Suspended"/>
 </xs:restriction>
 </xs:simpleType>

## 7.23 Component for returning the list of the signing certificate(s)

### 7.23.1 Component semantics

This component shall be used for returning available signing certificate(s) of the signer. Each signing certificate may also include its chain.

### 7.23.2 JSON related component

The component for returning available signing certificate(s) of the signer shall be represented by the following

* credentialIDs array of string type parameters;
* certificates array of string type parameters;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| credentialIDs | Required | Array of String | One or more credentialID associated with the provided or implicit signer identification. |
| certificates | Conditional | Array of String | Contains one or more Base64-encoded X.509v3 certificates from the signing certificate chain. If the returnCertificates parameter defined in 7.10.1.2 clause is “chain”, the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the certificates parameter is “single”, only the end entity certificate shall be returned. If the returnCertificates parameter is “none”, this parameter shall not be returned. |

### 7.23.3 XML related component

The element for returning the list of the signing certificate(s) shall be ds:KeyInfo, child element of the etsisig:OptionalOutputs root element.

## 7.24 Component for notifying operation result(s)

### 7.24.1 Component semantics

This component shall contain information representing the outcome of the request.

NOTE: When the requesting mode is synchronous this component will return the outcome of the processing. When the requesting mode is asynchronous this component will return the confirmation or not of the acceptance of the request.

### 7.24.2 JSON related component

When the HTTP status of the request is different from 200 OK, the outcome of the requested operation shall be returned in the body of the HTTP response using the “application/json” media type. The json structure includes the following

* error string type retrun;
* error\_description string type retrun

 specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| error | Required | String | An error code string. |
| error\_description | Optional | String | A human readable description, written in a language that considers the lang parameter specification, providing additional error information to assist the client application in understanding the error that occurred. |

The status codes and error messages defined in clause 10 of CSC [1] shall be used if applicable.

### 7.24.3 XML related component

The element for notifying the result shall be the dsb:Result with values of child elements dss2:ResultMajor and dss2:ResultMinor as specified in [2].

## 7.25 Component for service policy identification

### 7.25.1 Component semantics

This component shall be used to return the name of the service policy used by the server to perform the requested operation.

The inclusion of this component in the response is required by the client using the “Component for defining optional data to be returned” defined in clause 7.5.

### 7.25.2 JSON related component

The component for returning the service policy identification shall be represented by the following

* policy string type result

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| policy | Required | String | The element that identifies a particular policy associated with the SCS. |

### 7.25.3 XML related component

The element for service policy identification shall be dsb:AppliedPolicy, child element of the dss2:OptionalOutputs element.

## 7.26 Component for identification of the response

### 7.26.1 Component semantics

This component contains a string value generated by the SCS uniquely identifying the response originated from the SCS itself. This component is mainly used in asynchronous operation mode where the client application shall provide the responseID value received with the initial response included in the requestID component of any subsequent request polling the pending signature results.

### 7.26.2 JSON related component

The component for identifying the signature request whose results are requested shall be represented by the following

* responseID string type parameter;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| responseID | Required | String | Arbitrary string value generated by the SCS uniquely identifying the response originated from the SCS itself. |

### 7.26.3 XML related component

The element for identifying the signature request whose results are requested shall be the requestID attribute of the element dss2:PendingRequest root element.

## 7.27 Component for signature creation policy identification

### 7.27.1 Component semantics

This component shall contain information used to generate signature(s) and/or DSV(s).

The information shall contain:

1. The identification of the signature creation policy used while creating the DSV(s)
2. Other optional parameters containing the locations of the signature creation policy document

The inclusion of this component in the response is required by the client using the “Component for defining optional data to be returned” defined in clause 7.5.

NOTE: The signature creation policy may be a part of the service policy. The signature creation policy may therefore be implicitly identified by the applied service policy.

### 7.27.2 JSON related component

The component for returning the signature creation policy identification shall be represented by the following

* signaturePolicyID string type parameter;
* signaturePolicyLocations array of string type parameter.

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| signaturePolicyID | Required | String | The element that identifies a particular policy associated with the SCS. It shall have the value of a unique identifier of the signature creation policy as an URI. If the identifier of the signature creation policy is an OID, then the value of this element shall be an URN indicating the value of the aforementioned OID as specified in RFC 3061. |
| signaturePolicyLocations | Optional | Array of String | Every string element shall have the value of one location where the signature creation policy document can be accessed, as an URI value. |

### 7.27.3 XML related component

The element for signature creation policy identification shall be dsb:AppliedPolicy, child element of the dss2:OptionalOutputs element.

## 7.28 Component for returning credential authorization mode

### 7.28.1 Component semantics

This component specifies the authorization mode required by the identified signature credential.

The SCSP may specify the supported authorization modes, for example in the service policy or in the terms and conditions of the service.

Some possible returned values are:

1. “implicit”: the authorization process is managed by the SCS autonomously
2. “explicit”: the client shall collect needed factors of security elements
3. “authorizationCode”: the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on authorization code (RFC 6749)
4. “identificationToken”: the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on implicit grant (RFC 6749)

Other different values can be returned according to the authorization modes supported and specified by the SCSP.

### 7.28.2 JSON related component

The component for returning the authorization mode of the identified signature credential shall be represented by the following

* authMode string type result;

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| authMode | Required | String | Specifies one of the authorization modes:“implicit”: the authorization process is managed by the SCS autonomously.“explicit”: the client application shall collect the needed factors of security elements.“authorizationCode”: the authorization process is managed by the SCS using a mechanism based on authorization code (for example an OAuth 2.0 mechanism as described in RFC6749 [i.13]).“identificationToken”: the authorization process is managed by the SCS using a mechanism based on security tokens containing user profile information (like the user's name, email, and so forth), represented in the form of claims (for example a JWT as described in RFC7519 [i.14] or a SAML assertion). |

The “authMode” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.28.3 XML related component

The element for returning credential authorization mode shall be etsisig:AuthorizationMode, child element of the etsisig:OptionalOutputs element.

The etsisig:AuthorizationMode element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="AuthorizationMode" type="AuthorizationModeType"/>
<xs:simpleType name="AuthorizationModeType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Implicit"/>
        <xs:enumeration value="Explicit"/>
       <xs:enumeration value="AuthorizationCode"/>
        <xs:enumeration value="IdentificationToken"/>
    </xs:restriction>
</xs:simpleType>

## 7.29 Component for returning digital signature value(s)

### 7.29.1 Component semantics

This component shall contain a list of base64 encoded signature values corresponding to the DTBSR(s) passed in the documentDigests parameter specified in clause 7.19.

The digital signature value(s) position into the list shall be the same of the hashes included in DTBSR(s) component.

This component can be specified according to possible alternative behaviours of the SCS:

1. When one or more of the requested signatures fail, this component is not returned and an error code is returned as signature creation result outcome.
2. When one or more of the requested signatures fail, the corresponding DSV(s) are returned as empty values.

### 7.29.2 JSON related component

The component for returning the DSV(s) shall be represented by the following

* signatures array of string type result

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| signatures | Required | Array of String | One or more base64-encoded signature value(s). In case of multiple signatures, the signatures values shall be returned in the same order of the corresponding hashes provided as an input parameter. |

### 7.29.3 XML related component

The element for returning digital signature value(s) shall be dss2:SignatureObject, child element of the dss2:SignResponse root element. There shall be one dss2:SignatureObject for each DSV, according to the ETSI remote signing profile. The optional attribute WhichDocument shall contain one instance of a unique identifier reference, used to correlate the signature object with its input document or hash.

## 7.30 Component for returning sole control assurance level required

### 7.30.1 Component semantics

This component specifies the sole control assurance level required by the identified signature credential, as defined in [6]. Only two values of sole control assurance level shall be supported: “SCAL1” and “SCAL2”. The “SCAL1” value indicates that at least a basic authorization is required, the “SCAL2” value indicates the need to use a SAD value in order to ensure the authorized signer’s use of its signing key.

### 7.30.2 JSON related component

The component for returning the sole control assurance level required by the identified signature credential shall be represented by the following

1. SCAL string type result

specified according to the following table

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Presence | Type | Description |
| SCAL | Required | String | Specifies the Sole Control Assurance Level required by the credential, as defined in CEN EN 419 241-1 [6]:• “1”: at least a basic authorization is required (SCAL1). This level does not require to pass a Signature Activation Data to the signHash profile.• “2”: it is required that the authorized signer remotely controls the signing key with a high level of confidence (SCAL2). This level requires to pass the Signature Activation Data to the signHash profile. |

The “SCAL” element is defined in Cloud Signature Consortium Standard [1] and is copied here for information.

### 7.30.3 XML related component

The element for returning SCAL level required shall be etsisig:SoleControlAssuranceLevel, child element of the etsisig:OptionalOutputs element.

The SoleControlAssuranceLevel element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

<xs:element name="SoleControlAssuranceLevel" type="SoleControlAssuranceLevelType"/>

<xs:simpleType name="SoleControlAssuranceLevelType">
 <xs:restriction base="xs:string">
 <xs:enumeration value="SCAL1"/>
 <xs:enumeration value="SCAL2"/>
 </xs:restriction>
 </xs:simpleType>

# 8 Profiles for remote signature creation

## 8.1 Introduction

In this document the profiles are intended to represent the interfaces by which any client can interact with any SCS conforming to this specification. Technically speaking, profiles represent the way for accessing a web-based service. A profile can be considered a software-to-software programming interface enabling applications and services to talk to each other. The following clauses define the profiles to be provided by SCSs. For any profile it is specified which of the components defined in the previous clause can be used, by means of a table where the following information are provided:

* the reference to the clause where the component is specified;
* a brief description of the component;
* the presence of the component.

The value included in the column “Presence” of the table has the following meaning:

|  |  |
| --- | --- |
| Value | Description |
| **M** | The component shall be included in the request to or response from the SCS. |
| **O** | The component may be included in the request to or response from the SCS. |
| **C** | The component shall be included in the request or response based on the occurrence of certain conditions. |

## 8.2 Profile for signature request (A)

### 8.2.1 Profile semantics

The mandatory input parameter for this profile is either the signer’s document or the SDR (signer’s document representation, for example the hash value of the document). In practice, other parameters are usually also needed, such as:

other signature attributes, signed or unsigned (e.g. document type),

signature creation policy identification or signature policy parameters such as resulting signature format (AdES type),

signer identification, usually also authentication of signer,

identification of the signing key to be used, when more than one alternative exists for the user.

For user authentication, several alternatives exist, including use of identity assertions (SAML, Oauth2, OpenID Connect etc.) issued by trusted authentication and identity provision services.

The message for requesting the creation of AdES signature(s) to an SCASC shall contain the following components:

1) One component for submitting the document(s) or document representation(s) to be signed. Clause 7.19 specifies semantic requirements for this component.

NOTE 1: The message may contain one or more documents and/or one or more document representations for signature creation. An AdES signature will be created for each of these input components. Other components are used to select the type of signature that will be created for each document or document hash.

2) One component for notifying the server that the protocol profile defined by the present document is to be used. Clause 7.15 specifies semantic requirements for this component.

NOTE 2: The profile defined by the present document may be combined with other profiles to request additional features or functionality provided by the SCASC as long as these profiles do not conflict with the requirements specified in this document.

3) One component for identifying the signing key to be used by the server for computing the requested signature operations. Clause 7.8 specifies semantic requirements for this component.

The message for requesting the creation of AdES signature(s) to the SCASC may contain other components for requesting additional features. Clause 7.5 lists these optional components and contain references to clauses that specify semantic requirements for each component.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref.** | **Component for** | **Presence** |
| **7.2** | asynchronous/synchronous operation mode selection | **O** |
| **7.3** | identification of the request | **O** |
| **7.4** | credential authorization | **O** |
| **7.5** | defining optional data to be returned | **O** |
| **7.6** | defining the validity period for asynchronous requests | **O** |
| **7.7** | service authentication | **O** |
| **7.8** | identifying signature credentials | **M** |
| **7.9** | language and region selection | **O** |
| **7.11** | managing digital signatures transactions | **O** |
| **7.12** | service policy selection | **O** |
| **7.13** | signature creation policy selection | **O** |
| **7.14** | optional signature attributes/properties selection | **O** |
| **7.15** | protocol identifier | **M** |
| **7.16** | requesting specific signature formats | **O** |
| **7.18** | specifying response URL | **O** |
| **7.19** | submitting document(s) or hash(es) to be signed | **M** |

### 8.2.2 JSON related component

The element to request the signature of a document or a document representation shall be the set of required and optional parameters to invoke the document signature by means of the signatures/signDoc method.

**Processing model.**

The server shall process the components received with the signatures/signDoc in order to calculate the remote digital signature of one or multiple document(s) or SDR(s) as indicated in the clause 8.2.1 of the present document. This method can receive credential authorization in the form of Signature Activation Data (SAD).

### 8.2.3 XML related component

The element that shall be the main component for requesting the creation of AdES signature(s) shall be the root element of the message dss:SignRequest as specified in [2].

**Processing model.**
The server shall process the components in the dss:SignRequest as indicated in the corresponding clauses of [2].

The server shall process each child of the dss:OptionalInputs and dss:InputDocuments components as indicated in the corresponding clause of the present document if the child is not specified in any of the referenced OASIS documents. Otherwise, the server shall follow the processing model defined in the corresponding OASIS document.

## 8.3 Profile for signature response (B)

### 8.3.1 Profile semantics

The mandatory response returned by this profile is the result of the signature operation requested by the client. In practice, other responses are usually also returned, such as:

signed documents or signatures,

an identifier for correlating response to corresponding request, quite important if the asynchronous operating mode has been set in the corresponding request.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.21** | returning signed documents or signatures | **O** |
| **7.22** | returning signing certificate information | **O** |
| **7.24** | notifying operation result(s) | **M** |
| **7.25** | service policy identification | **O** |
| **7.26** | response identification | **M** |
| **7.27** | signature creation policy identification | **O** |

### 8.3.2 JSON related component

The element to respond to the signature of the document(s) request shall be the set of components as indicated in the clause 8.3.1 of the present document.

### 8.3.3 XML related component

The element that shall be the main component for responding to the creation of AdES signature(s) request shall be the root element of the message dss2:SignResponse as specified in [2].

## 8.4 Profile for DSVs creation request (C)

### 8.4.1 Profile semantics

In this scenario the SCASC or an application in the signer’s environment prepares the DTBS(s) that are sent to the SSASC along with other information required to compute the signature.

Therefore the mandatory input parameter for this profile is the DTBS(s). In practice, as in the case of profile defined in clause 8.2, other parameters are usually also needed, such as:

signature creation policy identification,

signer identification, usually also authentication of signer,

identification of the signing key to be used, when more than one alternative exists for the user.

The SSASC creates the digital signature value(s) using the signer’s private key held on the SCDev and returns the outcome of the signature operation and information to retrieve DSVs.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref.** | **Component for** | **Presence** |
| **7.2** | asynchronous/synchronous operation mode selection | **O** |
| **7.3** | identification of the request | **O** |
| **7.4** | credential authorization | **O** |
| **7.5** | defining optional data to be returned | **O** |
| **7.6** | defining the validity period for asynchronous requests | **O** |
| **7.7** | service authentication | **O** |
| **7.8** | identifying signature credentials | **M** |
| **7.9** | language and region selection | **O** |
| **7.11** | managing digital signatures transactions | **O** |
| **7.12** | service policy selection | **O** |
| **7.13** | signature creation policy selection | **O** |
| **7.15** | protocol identifier | **M** |
| **7.18** | specifying response URL | **O** |
| **7.19** | submitting document(s) or hash(es) to be signed | **M** |

### 8.4.2 JSON related component

The element to request the signature on a DTBSR shall be the set of required parameters to invoke the hash signature by means of the signatures/signHash method as specified in CSC [1].

**Processing model**

The server shall process the components as indicated in the signatures/signHash description of the CSC [1].

The server shall process each component as indicated in the clause 8.4.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

### 8.4.3 XML related component

The element for requesting the hash signature(s) shall be the root element of the message dss2:SignRequest as specified in the present clause.

**Processing model**

The server shall process the components inherited from dsb:RequestBaseType as indicated in the clause (Processing for XML Signature) in particular in the variant for <DocumentHash> of OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 - Working Draft 05 [2].

## 8.5 Profile for digital signature value response (D)

### 8.5.1 Profile semantics

This profile returns the results of the signatures generated by the SSASC using the information contained in the components received with the request message.

The mandatory response returned by this profile is the result of the signature operation requested by the client. In practice, other responses are usually also returned, such as:

DSV(s) generated by the SSASC,

an identifier for correlating response to corresponding request, quite important if the asynchronous operating mode has been set in the corresponding request.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.22** | returning signing certificate information | **O** |
| **7.24** | notifying operation result(s) | **M** |
| **7.25** | service policy identification | **O** |
| **7.26** | response identification | **M** |
| **7.27** | signature creation policy identification | **O** |
| **7.29** | returning DSV | **O** |

### 8.5.2 JSON related component

The element to respond to the signature of the hash(es) request shall be the set of parameters returned in the response of the signatures/signHash method as specified in CSC [1].

### 8.5.3 XML related component

The element to respond to the DTBS(s) signature request shall be the dss2:SignResponse root element as indicated in the clause 8.5.1 and in [2] specifications.

## 8.6 Profile for asynchronous processing (E)

### 8.6.1 Profile semantics

This profile shall manage the messages for requesting to the SCS to return the responses corresponding to previously sent (initial) requests processed in asynchronous mode by the SCS itself. Requests of this type are named pending-requests hereinafter.

In asynchronous processing one client usually sends an initial request to the server. The initial request shall contain, among other things, a request identifier generated by the client, as specified in clause 7.3.

The server can return a response indicating that the signature creation request has been accepted but it has not yet been completed. Within this initial response, the server shall convey a response identifier, as specified in clause 7.26. Both client and server can correlate the response identifier to the request identifier.

Under this processing model the client, after a certain time from the initial request, can send a pending-request to the server. This pending-request shall include the response identifier previously returned by the server. This response identifier allows the server to correlate this pending-request to the initial request and can return the signature creation results or return again an indication of “not yet finished”.

If this is the case, the client can send subsequent requests until the server returns a response with the signature creation result. Each subsequent request shall include the request identifier included in the initial request and returned by the server in the response to the initial request.

For managing asynchronous processing, a component for identifying the request as a pending-request associated to an initial request is required. The component specified in clause 7.3 shall be used to include the response identifier identifying the pending-request.

Other optional components may also be required by the SSASC. These optional components are defined in the clauses 7.7 and 7.9.

The main output is the DSV(s) or the signed document(s) or signature(s). This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.3** | identification of the request/polling signature results | **M** |
| **7.7** | service authentication | **O** |
| **7.9** | language and region selection | **O** |
| **7.15** | protocol identifier | **M** |

### 8.6.2 JSON related component

The element that shall indicate to the SCS that the client is requesting the response corresponding to a previously sent (initial) request (as part of an asynchronous protocol) shall be the set of required and optional parameters defined in clause 8.6.1 to invoke the signatures/signPolling method.

**Processing model.**

The server shall process each component as indicated in the clause 8.6.1 of the present document.

The server shall check the completion of the operations related to the request identified by the attribute requestID and respond accordingly returning the generated DSV(s) or signed document(s) or signature(s) or the indication that the operations have not yet finished or an error code (for example because the request cannot be completed or because the time allowed for requesting the results has expired).

### 8.6.3 XML related component

The element that shall indicate to the SCS that the client is requesting the response corresponding to a previously sent (initial) request (as part of an asynchronous protocol) shall be the dss2:PendingRequest element as specified in [2].

## 8.7 Profile for signing certificates list request (F)

### 8.7.1 Profile semantics

This profile shall be used to request the credentials list of a user identified with the component for “signer identification”. A user may have one or multiple credentials associated within a single user identifier.

The mandatory input parameter for this profile is the signer identification.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.3** | identification of the request | **O** |
| **7.7** | service authentication | **O** |
| **7.9** | language and region selection | **O** |
| **7.10** | contents from certificate chain to be returned | **O** |
| **7.15** | protocol identifier | **M** |
| **7.17** | signer identification | **M** |

### 8.7.2 JSON related component

The element to retrieve credentials shall be the set of parameters required by the credentials/list method as specified in CSC [1].

**Processing model**

The server shall process the components as indicated in the credentials/list description of the CSC [1].

The server shall process each component as indicated in the clause 8.7.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

### 8.7.3 XML related component

The element that shall be the main component for requesting a signing certificates list shall be the root element of the message etsisig:InformationRequest.

The etsisig:InformationRequest element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The etsisig:SignerIdentity, child element of the etsisig:OptionalInputs element shall be set in order to identify the signer for which the list of the signing credentials is being requested.

**Processing model**

The server shall process the components inherited from the etsisig:SignerIdentity element in order to retrieve and return the list of credentials associated with the user identifier.

## 8.8 Profile for signing certificates list response (G)

### 8.8.1 Profile semantics

This profile shall be used to return the credentials list of a user.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.23** | returning the list of the signing certificate(s) | **O** |
| **7.24** | notifying operation result(s) | **M** |

### 8.8.2 JSON related component

The element to respond to retrieve user’s credentials list shall be the set of results returned in the response of the credentials/list method as specified in CSC [1].

### 8.8.3 XML related component

The element that shall be the main component for responding with signing certificates list shall be the root element of the message etsisig:InformationResponse.

The etsisig:InformationResponse element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The ds:KeyInfo,child element of the etsisig:OptionalOutputs element will return the requested signing certificates list.

## 8.9 Profile for certificate information retrieval request (H)

### 8.9.1 Profile semantics

This profile shall be used to request some information about a signing certificate identified with the component for “identifying signing credential”.

The mandatory input parameter for this profile is the signing credential identification.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.3** | identification of the request | **O** |
| **7.7** | service authentication | **O** |
| **7.8** | identifying signature credentials | **M** |
| **7.9** | language and region selection | **O** |
| **7.10** | contents from certificate chain to be returned | **O** |
| **7.15** | protocol identifier | **M** |

### 8.9.2 JSON related component

The element to request credentials information shall be the set of parameters required by the credentials/info method as specified in CSC [1].

**Processing model**

The server shall process the components as indicated in the credentials/info description of the CSC [1].

The server shall process each component as indicated in the clause 8.9.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

### 8.9.3 XML related component

The element that shall be the main component for requesting certificate information retrieval shall be the root element of the message etsisig:InformationRequest.

The etsisig:InformationRequest element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The dss2:KeySelector,child element of the etsisig:OptionalInputs element shall be set in order to identify the signing credential whose information are needed to be returned as main output of the profile.

## 8.10 Profile for certificate information retrieval response (I)

### 8.10.1 Profile semantics

This profile shall be used to return some information about a signing certificate.

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.22** | returning signing certificate information | **O** |
| **7.24** | notifying operation result(s) | **M** |
| **7.28** | returning credential authorization mode | **O** |
| **7.30** | returning SCAL level required | **O** |

### 8.10.2 JSON related component

The element to respond to retrieve user’s credential list shall be the set of parameters returned in the response of the credentials/info method as specified in CSC [1].

### 8.10.3 XML related component

The element that shall be the main component for responding with certificate information retrieval shall be the root element of the message etsisig:InformationResponse.

The etsisig:InformationResponse element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The ds:KeyInfo,child element of the etsisig:OptionalOutputs element will return the requested signing certificate and key information.

## 8.11 Profile for service information request (J)

### 8.11.1 Profile semantics

This component may returns several information about the SCS and the list of the functionalities implemented and supported by it.

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.9** | language and culture selection | **O** |

### 8.11.2 JSON related component

The component to request service information shall be the set of required parameters required by the info method as specified in CSC [1].

**Processing model**

The server shall process the components as indicated in the info description of the CSC [1].

The server shall process each component as indicated in the clause 8.11.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

### 8.11.3 XML related component

The element that shall be the main component for requesting service information shall be the root element of the message etsisig:InformationRequest.

The etsisig:InformationRequest element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The dsb:Language, child element of the etsisig:OptionalInputs element can be set for language and culture selection.

## 8.12 Profile for service information response (K)

### 8.12.1 Profile semantics

This profile includes the following components:

|  |  |  |
| --- | --- | --- |
| **Ref** | **Component for** | **Presence** |
| **7.20** | returning service information | **M** |

### 8.12.2 JSON related component

The component to respond to service information request list shall be the set of values returned in the response of the info method as specified in CSC [1].

### 8.12.3 XML related component

The element that shall be the main component for responding with service information shall be the root element of the message etsisig:InformationResponse.

The etsisig:InformationResponse element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The etsisig:ServiceInformation,child element of the etsisig:OptionalOutputs element will return the requested service information.

## 8.13 Component use summary

A = Profile for signature request B = Profile for signature response
C = Profile for digital signature value request D = Profile for digital signature value response
E = Profile for asynchronous processing F = Profile for signing certificates list request
G = Profile for signing certificates list response H = Profile for certificate information retrieval request
I = Profile for certificate information retrieval response J = Profile for service information request
K = Profile for service information response

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ref.** | **Component for:** | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** |
| **7.2** | asynchronous/synchronous operation mode selection | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.3** | identification of the request | **O** |   | **O** |  | **M** | **O** |  | **O** |  |  |  |
| **7.4** | credential authorization | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.5** | defining optional data to be returned | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.6** | defining the validity period for asynchronous requests | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.7** | service authentication | **O** |   | **O** |  | **O** | **O** |  | **O** |  |  |  |
| **7.8** | identifying signature credentials | **M** |   | **M** |  |  |  |  | **O** |  |  |  |
| **7.9** | language and region selection | **O** |   | **O** |  | **O** | **O** |  | **O** |  | **O** |  |
| **7.10** | list the certificate chain |   |   |  |  |  | **O** |  | **O** |  |  |  |
| **7.11** | managing digital signatures transactions | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.12** | service policy selection | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.13** | signature creation policy selection | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.14** | optional signature attributes/properties selection | **O** |  |  |  |  |  |  |  |  |  |  |
| **7.15** | protocol identifier | **M** |   | **M** |  | **M** | **M** |  | **M** |  |  |  |
| **7.16** | requesting specific signature formats | **O** |   |  |  |  |  |  |  |  |  |  |
| **7.17** | signer identification |   |   |  |  |  | **M** |  |  |  |  |  |
| **7.18** | specifying response URL | **O** |   | **O** |  |  |  |  |  |  |  |  |
| **7.19** | submitting document(s) or hash(es) to be signed | **M** |   | **M** |  |  |  |  |  |  |  |  |
| **7.20** | returning service information |   |   |  |  |  |  |  |  |  |  | **M** |
| **7.21** | returning signed documents or signatures |   | **O** |  |  |  |  |  |  |  |  |  |
| **7.22** | returning signing certificate information |  | **O** |  | **O** |  |  |  |  | **O** |  |  |
| **7.23** | returning the list of the signing certificate(s) |   |   |  |  |  |  | **O** |  |  |  |  |
| **7.24** | notifying operation result(s) |  | **M** |  | **M** |  |  | **M** |  | **M** |  |  |
| **7.25** | service policy identification |   | **O** |  | **O** |  |  |  |  |  |  |  |
| **7.26** | identification of the response |   |  **M** |  | **M** | **M** |  |  |  |  |  |  |
| **7.27** | signature creation policy identification |   | **O** |  | **O** |  |  |  |  |  |  |  |
| **7.28** | returning credential authorization mode |   |   |  |  |  |  |  |  | **O** |  |  |
| **7.29** | returning DSV |   |   |  | **O** |  |  |  |  |  |  |  |
| **7.30** | returning SCAL level required |   |   |  |  |  |  |  |  | **O** |  |  |

\*\*\*

Annex A (normative):
XML and JSON Schema files

# A.1 JSON Schema file location for “$schema” "http://uri.etsi.org/19432/v1.1.1/json#"

The file at [JSONSIGCREATIONPROT\_URL] ([JSONSIGCREATIONPROT]) contains the definitions of elements and types defined within the JSON schema whose "$schema" value is "http://uri.etsi.org/19432/v1.1.1/json#"

# A.2 XML Schema file location for namespace http://uri.etsi.org/19432/v1.1.1#

The file at [XSDSIGCREATIONPROT\_URL] ([XSDSIGCREATIONPROT]) contains the definitions of elements and types defined within the namespace whose URI value is http://uri.etsi.org/19432/v1.1.1/xml#.

# History

|  |
| --- |
| **Document history** |
| v0.0.0.a | December 2017 | For discussions between STF539 members. |
| v0.0.1 | February 2018 | First draft |
| v0.0.2 | June 2018 | New draft to be discussed at ESI#63 meeting |
| v0.0.3 | June 2018 | New draft to be discussed at ESI#63 meeting, added some previously missing subclauses |
| V0.0.4 | June 2018 | New draft after comments received at ESI#63 meeting |
| V0.0.5 | July 2018 | Stable draft for public review |
| V0.0.6 | September 2018 | New draft, not yet complete, after public review comments to be discussed at ESI#64 meeting  |
| V0.0.7 | October 2018 | New draft, after public review comments, for ESI approval  |
| V0.0.8 | December 2018 | New draft for ESI approval |