XML Profile for Signature Validation Tokens

Version 1.0 - 2020-09-24 - *Draft version*

Registration number: **2020-62**

Table of Contents

1. Introduction

The "Signature Validation Token" specification [SVT] defines the basic token to support signature validation in a way that can significantly extend the lifetime of a signature.

This specification defines a profile for implementing SVT with a signed XML document, and defines the following aspects of SVT usage:

- How to include reference data related to XML signatures and XML documents in an SVT.
- How to add an SVT token to a XML signature.

XML documents can have any number of signature elements, signing an arbitrary number of fragments of XML documents. The actual signature element may be included in the signed XML document (enveloped), include the signed data (enveloping) or may be separate from the signed content (detached).

To provide a generic solution for any type of XML signature an SVT is added to each XML signature element within the XML signature `<ds:Object>` element.

1.1. Requirements Notation

The key words **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**, **SHOULD NOT**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

These keywords are capitalized when used to unambiguously specify requirements over protocol features and behavior that affect the interoperability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

1.2. Definitions

The definitions in [SVT] apply also to this document.

1.2. Notation

1.2.1 References to XML Elements from XML Schemas

When referring to elements from the W3C XML Signature namespace (`http://www.w3.org/2000/09/xmldsig\#`) the following syntax is used:

- `<ds:Signature>`

When referring to elements from the ETSI XAdES XML Signature namespace (`http://uri.etsi.org/01903/v1.3.2#`) the following syntax is used:

- `<xades:CertDigest>`

When referring to elements defined in this specification (`http://id.swedenconnect.se/svt/1.0/sig-prop/ns`) the following syntax is used:

- `<svt:Element>`

2. SVT in XML Documents

When SVT is provided for XML signatures then one SVT SHALL be provided for each XML signature.

An SVT embedded within the XML signature element SHALL be placed in a `<svt:SignatureValidationToken>` element as defined in section 2.1.1.

2.1.1. SignatureValidationToken Signature Property

The `<svt:SignatureValidationToken>` element SHALL be placed in a `<ds:SignatureProperty>` element in accordance with [XMLDsig]. The `<ds:SignatureProperty>` element SHALL be placed inside a `<ds:SignatureProperties>` element inside a `<ds:Object>` element inside a `<ds:Signature>` element.

Note: [XMLDsig] requires the `Target` attribute to be present in `<ds:SignatureProperty>`, referencing the signature targeted by this signature property. If an SVT is added to a signature that do not have an `Id` attribute, implementations SHOULD add an `Id` attribute to the `<ds:Signature>` element and reference that `Id` in the `Target` attribute. This `Id` attribute and `Target` attribute value matching is required by the [XMLDsig] standard, but it is redundant in the context of SVT validation as the SVT already contains information that uniquely identifies the target signature. Validation applications SHOULD not reject an SVT token because of `Id` and `Target` attribute mismatch, and MUST rely on matching against signature using signed information in the SVT itself.

The `<svt:SignatureValidationToken>` element is defined by the following XML Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
    targetNamespace="http://id.swedenconnect.se/svt/1.0/sig-prop/ns"
    xmlns:svt="http://id.swedenconnect.se/svt/1.0/sig-prop/ns">

  <xs:element name="SignatureValidationToken" type="svt:SignatureValidationTokenType" />

  <xs:complexType name="SignatureValidationTokenType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

</xs:schema>
```

The SVT token SHALL be included as a string representation of the SVT JWT. Note that this is the string representation of the JWT without further encoding. The SVT MUST NOT be represented by the Base64 encoded bytes of the JWT string.

Example:

```
<ds:Signature Id="MySignatureId">
  ...
  <ds:Object>
    <ds:SignatureProperties>
      <ds:SignatureProperty Target="#MySignatureId">
        <svt:SignatureValidationToken>
              eyJ0eXAiOiJKV1QiLCJhb...2aNZ
        </svt:SignatureValidationToken>
      </ds:SignatureProperty>
    </ds:SignatureProperties>
  </ds:Object>
</ds:Signature>
```

2.1.2 Multiple SVT in a signature

If a new SVT is stored in a signature which already contains a previously issued SVT, implementations can choose to either replace the existing SVT or to store the new SVT in addition to the existing SVT.

If the new SVT is stored in addition to the old SVT, it SHOULD be stored in a new `<ds:SignatureProperty>` element inside the existing `<ds:SignatureProperties>` element where the old SVT is located.

For interoperability robustness, signature validation applications MUST be able to handle signatures where the new SVT is located in a new `<ds:Object>` element.

3. SVT Claims

3.1. Signature Reference Data

The SVT SHALL contain a **SigReference** claims object that SHALL contain the following data:

Claim	Value
`id`	The Id-attribute of the XML signature, if present.
`sig_hash`	The hash over the signature value bytes.
`sb_hash`	The hash over the canonicalized `<ds:SignedInfo>` element (the bytes the XML signature algorithm has signed to generated the signature value).

3.2. Signed Data Reference Data

An SVT according to this profile SHALL contain one instance of the **SignedData** claims object for each `<ds:Reference>` element in the `<ds:SignedInfo>` element. The **SignedData** claims object shall contain the following data:

Claim	Value
`ref`	The value of the URI attribute of the corresponding `<ds:Reference>` element.
`hash`	The hash of all bytes identified corresponding `<ds:Reference>` element after applying all identified canonicalization and transformation algorithms. These are the same bytes that is hashed by the hash value in the `<ds:DigestValue>` element inside the `<ds:Reference>` element.

3.3. Signer Certificate References

The SVT SHALL contain a **CertReference** claims object. The `type` claim of the **CertReference** claims object SHALL be either `cert`, `chain`, `cert_hash` or `cert_and_chain_hash`.

- The `cert` type SHALL be used when signature validation was performed using a single certificate not present in the target XML signature.
- The `chain` type SHALL be used when signature validation was performed using a certificate chain where some or all of the certificates in the chain are not present in the target signature.
- The `cert_hash` type SHALL be used when signature validation was performed using a single certificate that is present in the target XML signature.
- The `cert_and_chain_hash` type SHALL be used when signature validation was performed using a certificate chain where all of the certificates in the chain are present in the target signature.

Note: The `cert` type MUST NOT be used with a XAdES signatures where the signing certificate in the target signature is bound to the signature through a `<xades:CertDigest>` element.

4. JOSE Header

4.1. SVT Signing Key Reference

The SVT JOSE header MUST contain one of the following header parameters in accordance with [RFC7515], for storing a reference to the public key used to verify the signature on the SVT:

Header Parameter	Value
`x5c`	Holds an X.509 certificate [RFC5280] or a chain of certificates. The certificate holding the public key that verifies the signature on the SVT MUST be the first certificate in the chain.
`kid`	A key identifier holding the Base64 encoded hash value of the certificate that can verify the signature on the SVT. The hash algorithm MUST be the same hash algorithm used when signing the SVT as specified by the `alg` header parameter.

5. Normative References

[RFC2119]

> Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, March 1997.

[RFC5280]

> D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, May 2008.

[RFC8174]

> Leiba, B., Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, May 2017.

[XMLDsig]

> XML Signature Syntax and Processing Version 1.1 - W3C Recommendation 11 April 2013.

[EidRegistry]

> Swedish eID Framework - Registry for identifiers.

[SVT]

> Signature Validation Token.