1

XML Timestamping Profile of the OASIS Digital Signature Services

2nd Committee Draft, 12 September 2006 (WD11)

5 **Document identifier:**
6 oasis-dss-1.0-profiles-timestamping-spec-cd-r2

7 **Location:**
8 http://docs.oasis-open.org/dss/v1.0/

9 **Editor:**
10 Trevor Perrin, *individual*
11 Juan Carlos Cruellas, *individual* <cruellas@ac.upc.edu>

12 **Contributors:**
13 Dimitri Andivahis, Surety
14 Frederick Hirsch, Nokia
15 Pieter Kasselman, Betrusted
16 Andreas Kuehne, *individual*
17 Paul Madsen, Entrust
18 John Messing, American Bar Association
19 Tim Moses, Entrust
20 Nick Pope, *individual*
21 Rich Salz, DataPower
22 Ed Shallow, Universal Postal Union

23 **Abstract:**
24 This document profiles the OASIS DSS core protocols for the purpose of creating and
25 verifying XML-encoded time-stamps.

26 **Status:**
27 This is a **Public Review Draft** produced by the OASIS Digital Signature Service
28 Technical Committee. Comments may be submitted to the TC by any person by clicking
29 on "Send A Comment" on the TC home page at:
30 http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dss.

31 For information on whether any patents have been disclosed that may be essential to
32 implementing this specification, and any offers of patent licensing terms, please refer to
33 the Intellectual Property Rights section of the Digital Signature Service TC web page at
34 http://www.oasis-open.org/committees/dss/ipr.php.

Table of Contents

1 Introduction

The DSS signing and verifying protocols are defined in **[DSSCore]**. As defined in that document, these protocols have a fair degree of flexibility and extensibility. This document profiles these protocols to limit their flexibility and extend them in concrete ways. The resulting profile is suitable for implementation and interoperability.

The following sections describe how to understand the rest of this document.

1.1 Notation

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this specification are to be interpreted as described in IETF RFC 2119 **[RFC 2119]**. 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.

This specification uses the following typographical conventions in text: `<ns:Element>`, `Attribute`, **Datatype**, `OtherCode`.

1.2 Namespaces

The structures described in this specification are contained in the schema file [TST-XSD]. All schema listings in the current document are excerpts from the schema file. In the case of a disagreement between the schema file and this document, the schema file takes precedence.

This schema is associated with the following XML namespace:

`urn:oasis:names:tc:dss:1.0:profiles:TimeStamp:schema#`

Conventional XML namespace prefixes are used in this document:

- The prefix `dss:` stands for the DSS core namespace **[Core-XSD]**.

Applications MAY use different namespace prefixes, and MAY use whatever namespace defaulting/scoping conventions they desire, as long as they are compliant with the Namespaces in XML specification **[XML-ns]**.

2 Profile Features

2.1 Identifier

urn:oasis:names:tc:dss:1.0:profiles:timestamping

2.2 Scope

This document profiles the DSS signing and verifying protocols defined in **[DSSCore]**.

2.3 Relationship To Other Profiles

This profile is based directly on the **[DSSCore]**.

2.4 Signature Object

This profile supports the creation and verification of isolated <dss:Timestamp> elements as defined in **[DSSCore]**. These elements can wrap different types of time-stamp tokens; this profile does not specify or constrain the internal structure of the <dss:Timestamp>, unless the <dss:SignatureType> optional input is used (see section 3.1.1).

2.5 Transport Binding

This profile is transported using the HTTP POST Transport Binding defined in **[DSSCore]**.

2.6 Security Binding

This profile is secured using the TLS X.509 Server Authentication Binding defined in **[DSSCore]**.

3 Profile of Signing Protocol

3.1 Element <SignRequest>

3.1.1 Element <OptionalInputs>

The <dss:SignatureType> optional input from **[DSSCore]** is supported and may be sent by the client. The timestamping specific optional input <RenewTimestamp> may also be supported and may be sent by the client. No other optional inputs are supported.

3.1.1.1 Element <SignatureType>

The <dss:SignatureType> optional input may be one of these values, from section 7. of **[DSSCore]**:

urn:oasis:names:tc:dss:1.0:core:schema:XMLTimeStampToken

urn:ietf:rfc:3161

Servers may support other values. However, servers are under no obligation to support *any* particular values. Thus, clients using the <dss:SignatureType> optional input may not interoperate with certain servers.

3.1.1.2 Element <RenewTimestamp>

The <RenewTimestamp> optional input element indicates to the server that the current sign request is a request for the renewal of an existing timestamp on data that were timestamped in the past, so that the validity period of the existing timestamp is effectively extended.

```
<xs:element name="RenewTimestamp">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="PreviousTimestamp">
    <xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PreviousTimestamp">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="dss:Timestamp">
    <xs:sequence>
  </xs:complexType>
</xs:element>
```

If the <RenewTimestamp> optional input is present in the sign request submitted by the client to the server, and it is supported by the server, the <PreviousTimestamp> element contained in this optional input must also be present as an element of the resulting timestamp generated by

153 the server and returned to the client. For XML timestamps of type <ds:signature>, processing
154 rules are described in Section 3.2.3.

155 Before submitting the sign request, the client must verify that the <PreviousTimestamp>
156 element corresponds to the document(s) being re-timestamped, and the client should verify the
157 <PreviousTimestamp> element.

158 Note: Legitimate reasons to renew a timestamp include (a) the public key certificate used to verify
159 the digital signature in the timestamp is nearing its expiration date, or (b) the client needs to
160 replace the hash value used for the timestamped data in the existing timestamp with a hash value
161 using a stronger hash algorithm.

162 3.1.2 Element <InputDocuments>

163 The client MAY send any component of <dss:InputDocument> element as input document. The
164 extraction and processing of these elements MUST be carried out as indicated in the core
165 document, with the changes mentioned in the present document.
166 If the client is not sending the <dss:SignatureType> optional input, then the client SHOULD only
167 send a single input document, since some types of time-stamps (e.g. RFC 3161) can only cover
168 one document per time-stamp.

169 If the client is sending the <dss:SignatureType> optional input, then the client MAY send multiple
170 input documents, if the client knows that the specified time-stamp type can handle them.

171 3.2 Element <SignResponse>

172 3.2.1 Element <Result>

173 This profile defines no additional <ResultMinor> codes.

174 3.2.2 Element <OptionalOutputs>

175 The server MUST NOT return any optional outputs.

176 3.2.3 Element <SignatureObject>

177 The server MUST return a <dss:Timestamp> signature object.

178 If the <RenewTimestamp> optional input is present in the sign request submitted by the client to
179 the server, and it is supported by the server, the <PreviousTimestamp> element contained in
180 this optional input must also be present as an element of the resulting timestamp generated by
181 the server and returned to the client. Specifically, for XML processing rules for XML timestamps
182 of type <ds:signature>, the server must include the <PreviousTimestamp> element
183 contained in the optional input as a child of an additional <ds:Signature>/<ds:Object> in
184 the newly generated timestamp (i.e. in addition to the <ds:object> containing the
185 <TstInfo>). An additional <ds:SignedInfo>/<ds:Reference> referencing the
186 <ds:Object>/<dss:PreviousTimestamp> must be included in the signature of the new
187 timestamp signature.

188 The server generating the new timestamp in response to a request carrying the
189 <RenewTimestamp> optional input need make no assertions about the validity of the
190 <PreviousTimestamp> element submitted to it within this optional input.

191 A server that does not support the <RenewTimestamp> optional input must reject the sign
192 request with a <ResultMajor> code of RequesterError and a <ResultMinor> code
193 urn:oasis:names:tc:dss:1.0:resultminor:NotSupported.

4 Profile of Verifying Protocol

4.1 Element <VerifyRequest>

4.1.1 Element <OptionalInputs>

The client may submit the <UseVerificationTime> optional input to instruct the server to determine the timestamp's validity at the specified time, instead of the current time. No other optional inputs are supported.

4.1.2 Element <SignatureObject>

The client MUST send a <dss:Timestamp> signature object.

Note: A timestamp T_2 that was generated by a server in response to a renewal request for timestamp T_1, that is, in response to a sign request on the same data as for timestamp T_1 and carrying timestamp T_1 within the <PreviousTimestamp> element of the <RenewTimestamp> optional input, may be used to assert current time validity for timestamp T_1. This situation applies when timestamp T_1's current time validity can no longer be asserted independently, for example, because the cryptographic primitives in timestamp T_1 are considered compromised. Specifically, the client may:

- submit a verify request for timestamp T_2,
- submit a verify request for timestamp T1 and include the optional input <UseVerificationTime> with a value set to the issue time of timestamp T2 (i.e. using element <SpecificTime>).

If the result codes in the server verify responses indicate that both timestamps are valid as requested, the client may assert that timestamp T_1 is currently valid, as supported by the fact that timestamp T_1 is considered valid at the issue time of timestamp T_2, and timestamp T_2 is considered valid currently. This process may be generalized to timestamps that were generated after multiple renewal requests on the same data, that is, timestamp T_1, renewed by timestamp T_2, renewed by timestamp T_3, and so on.

4.1.3 Element <InputDocuments>

The client MAY send any component of <dss:InputDocuments> element as input documents. The extraction and processing of these elements MUST be carried out as indicated in the core document, with the changes mentioned in the present document.

4.2 Element <VerifyResponse>

4.2.1 Element <Result>

This profile defines no additional <dss:ResultMinor> codes.

4.2.2 Element <OptionalOutputs>

The server MUST return the `<dss:SigningTimeInfo>` optional output.

4.2.2.1 Element <SigningTimeInfo>

The server MUST return this optional output profiled as detailed below:

1. Its `<dss:SigningTime>` child will contain the time indicated in the timestamp token (the value in `<dss:CreationTime>` element of DSS XML timestamps or the `genTime` field in RFC 3161 timestamp tokens).
2. If the timestamp token verified includes an indication of the deviation around the time present in the timestamp token (like the `accuracy` field in RFC 3161 timestamps or the `<dss:ErrorBound>` element in DSS XML timestamps), its `<dss:SigningTimeBoundaries>` child MUST be present and it MUST contain the lower and the upper boundaries suitably computed within its children.

The server MUST NOT return any other optional outputs.

5 Editorial Issues

1) What type of signature object should be supported? An <XMLTimeStampToken> (like now) or a more generic <Timestamp>?

This profile supports a generic Timestamp; a profile of this profile could make it more specific.

2) What bindings should be used? A SOAP binding (like now) or a simple HTTP POST binding?

We're referencing an HTTP POST binding, for now.

3) Are the clients required to verify received timestamps? Does this eliminate the need for an authenticated binding in the signing profile?

Right now it says no.

6 References

6.1 Normative

[Core-XSD] T. Perrin et al. *DSS Schema*. OASIS, **(MONTH/YEAR TBD)**

[DSSCore] T. Perrin et al. *Digital Signature Service Core Protocols and Elements*. OASIS, **(MONTH/YEAR TBD)**

[TST-XSD] T. Perrin et al. Timestamping Profile Schema, OASIS, **(MONTH/YEAR TBD)**

[RFC 2119] S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels.* IETF RFC 2396, August 1998. http://www.ietf.org/rfc/rfc2396.txt.

[XML-ns] T. Bray, D. Hollander, A. Layman. *Namespaces in XML*. W3C Recommendation, January 1999. http://www.w3.org/TR/1999/REC-xml-names-19990114

[XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing.* W3C Recommendation, February 2002. http://www.w3.org/TR/1999/REC-xml-names-19990114

271

Appendix A. Revision History

Rev	Date	By Whom	What
wd-01	2004-01-06	Trevor Perrin	Initial version
wd-02	2004-01-20	Trevor Perrin	Added “Type of Signature Object” section, and editorial issues 1-3; organized references
wd-03	2004-02-03	Trevor Perrin	Reorganized; based around <dss:Timestamp> instead of XMLTimeStampToken.
Wd-04	2004-02-29	Trevor Perrin	Changed Verify Response to use <SigningTime> optional output.
Wd-06	2004-06-28	Trevor Perrin	Mentioned as committee draft
Wd-07	2006-04-06	Trevor Perrin	Added optional input RenewTimestamp
Wd-08	2006-08	Juan Carlos Cruellas	Alignment with version 46 of the core
Wd-09	2006-09	Juan Carlos Cruellas	Addition of material for dealing with RenewTimestamp.
Wd-10	2006-09	Juan Carlos Cruellas	Small modification of XML tag for alignment with core.

Appendix B. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.