



---

# Signature Archive Profile of the OASIS Digital Signature Service (DSS)

## Working Draft 01, 27 December 2005

**Document identifier:**

**Location:**

<http://www.oasis-open.org/committees/dss>

**Editor:**

Carlos González-Cadenas, netfocus

**Contributors:**

Marta Cruellas, CATCert

Francesc Oliveras, CATCert

Ignacio Alamillo, CATCert

**Abstract:**

This document defines XML request/response protocols for requesting server-based archive operations over electronic signatures.

**Status:**

This is a **Working Draft** produced by the OASIS Digital Signature Service Technical Committee. Committee members should send comments on this draft to [dss@lists.oasis-open.org](mailto:dss@lists.oasis-open.org).

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

## Table of Contents

29	1	Introduction .....	4
30	1.1	Notation .....	4
31	1.2	Schema Organization and Namespaces.....	4
32	2	Profile Features.....	5
33	2.1	Identifier.....	5
34	2.2	Scope .....	5
35	2.3	Relationship to Other Profiles .....	5
36	2.4	Signature Object.....	5
37	2.5	Transport Binding.....	5
38	2.6	Security Binding .....	5
39	3	Common Protocol Structures.....	6
40	3.1	ArchiveIdentifier-based Requests .....	6
41	3.2	Result Codes.....	6
42	4	Archive Protocol.....	7
43	4.1	The Archive Submit Protocol.....	7
44	4.1.1	Element <ArchiveSubmitRequest>.....	7
45	4.1.2	Element <ArchiveSubmitResponse> .....	7
46	4.1.3	Optional Inputs and Outputs .....	8
47	4.2	The Archive Retrieval Protocol.....	9
48	4.2.1	Element <ArchiveRetrievalRequest>.....	9
49	4.2.2	Element <ArchiveRetrievalResponse> .....	10
50	4.2.3	Optional Inputs and Outputs .....	10
51	4.3	The Archive Delete Protocol .....	10
52	4.3.1	Element <ArchiveDeleteRequest>.....	10
53	4.3.2	Element <ArchiveDeleteResponse>.....	10
54	4.3.3	Optional Inputs and Outputs .....	11
55	4.4	The Archive Modify Protocol .....	11
56	4.4.1	Element <ArchiveModifyRequest> .....	11
57	4.4.2	Element <ArchiveModifyResponse>.....	11
58	4.4.3	Optional Inputs and Outputs .....	11
59	4.5	The Archive Verify Protocol.....	12
60	4.5.1	Element <ArchiveVerifyRequest>.....	12
61	4.5.2	Element <ArchiveVerifyResponse> .....	12
62	4.5.3	Optional Inputs and Outputs .....	12
63	4.6	Result Codes.....	12

64	5	Identifiers.....	14
65	5.1	Archive Modes.....	14
66	6	References.....	15
67	6.1	Normative .....	15
68		Appendix A. Revision History .....	16
69		Appendix B. Notices .....	17
70			

---

# 71 1 Introduction

## 72 1.1 Notation

73 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",  
74 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be  
75 interpreted as described in IETF RFC 2119 [RFC 2119]. These keywords are capitalized when  
76 used to unambiguously specify requirements over protocol features and behavior that affect the  
77 interoperability and security of implementations. When these words are not capitalized, they are  
78 meant in their natural-language sense.

79 This specification uses the following typographical conventions in text:  
80 <ArchiveProfileElement>, <ns:ForeignElement>, Attribute, **Datatype**, OtherCode.

81 `Listings of Archive schemas appear like this.`

## 82 1.2 Schema Organization and Namespaces

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

86 This schema is associated with the following XML namespace:

87 `urn:oasis:names:tc:dss:1.0:profiles:archive`

88 If a future version of this specification is needed, it will use a different namespace.

89 Conventional XML namespace prefixes are used in the schema:

- 90
- 91 • The prefix `dss`: stands for the DSS core namespace [Core-XSD].
  - 92 • The prefix `ds`: stands for the W3C XML Signature namespace [XMLSig].
  - 93 • The prefix `xs`: stands for the W3C XML Schema namespace [Schema1].
  - The prefix `archp`: or no prefix defaults to the namespace of the present document.

---

94 **2 Profile Features**

95 **2.1 Identifier**

96 urn:oasis:names:tc:dss:1.0:profiles:archive

97 **2.2 Scope**

98 This document adds an archive protocol to the other ones described in **[DSSCore]**.

99 This profile is concrete, can be directly implemented, and MAY be further profiled.

100 **2.3 Relationship to Other Profiles**

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

102 **2.4 Signature Object**

103 The signature object can only include signatures, therefore excluding other objects like  
104 timestamps and certificates.

105 **2.5 Transport Binding**

106 This profile does not constrain any transport binding defined in **[DSSCore]**.

107 **2.6 Security Binding**

108 This profile does not constrain any security binding defined in **[DSSCore]**.

---

## 109 3 Common Protocol Structures

### 110 3.1 ArchivelIdentifier-based Requests

111 Several requests of the protocols described below need to include the identifier of the archived  
112 signature in order to point the signature object that is subject to operation.

113 The requests of these protocols can use the **ArchivelIdentifierRequest** complex type, defined as  
114 follows.

115 `<dss:OptionalInputs>` [Optional]

116 The optional inputs that MAY customize the archive process.

117 `<ArchivelIdentifier>` [Required]

118 The identifier of the archived signature subject to the operation.

119 The elements `RequestID` and `Profile` MUST be interpreted as stated in **[DSSCore]**.

```
120 <xs:complexType name="ArchivelIdentifierRequest">
121   <xs:sequence>
122     <xs:element ref="dss:OptionalInputs" minOccurs="0"/>
123     <xs:element name="ArchivelIdentifier" type="ArchivelIdentifier"/>
124   </xs:sequence>
125   <xs:attribute name="RequestID" type="xs:string" use="optional"/>
126   <xs:attribute name="Profile" type="xs:anyURI" use="optional"/>
127 </xs:complexType>
```

128

129 When the identifier for the object cannot be found by the server, it MUST refuse the request using  
130 the minor code `ArchivelIdentifierNotFound`.

### 131 3.2 Result Codes

132 This section includes common result codes used by the different protocols defined within this  
133 document.

134 The URN used for the `<dss:ResultMajor>` elements is described in **[DSSCore]**. The URN  
135 used for the `<dss:ResultMinor>` elements MUST be  
136 `urn:oasis:names:tc:dss:1.0:profiles:archive:resultminor:` followed by the  
137 codes described below.

<code>&lt;dss:ResultMajor&gt;</code>	<code>&lt;dss:ResultMinor&gt;</code>	Description
<code>RequesterError</code>	<code>ArchivelIdentifierNotFound</code>	There's no signature archived with the identifier included in the request.

138

## 4 Archive Protocol

139 The following protocol supports server-side (long-term) archival of electronic signatures, allowing  
140 the clients to interact with the server to

- 141 • archive signatures (the **Archive Submit Protocol**),
  - 142 • retrieve archived signatures (the **Archive Retrieval Protocol**)
  - 143 • delete archived signatures (the **Archive Delete Protocol**),
  - 144 • modify the archival options of an archived signature (the **Archive Modify Protocol**)
  - 145 • verify the integrity of the stored signatures (the **Archive Verify Protocol**)
- 146

### 147 4.1 The Archive Submit Protocol

148 This protocol allows the client to submit signatures to the service for their archival.

#### 149 4.1.1 Element <ArchiveSubmitRequest>

150 The element <ArchiveSubmitRequest> can be used to request the archival of a signature or  
151 timestamp.

152 <dss:OptionalInputs> [Optional]

153 The optional inputs that customize the archive submit process.

154 <dss:SignatureObject> [Required]

155 The signature to archive.

156 The elements RequestID and Profile MUST be interpreted as stated in [DSSCore].

```
157 <xs:element name="ArchiveSubmitRequest">
158   <xs:complexType>
159     <xs:sequence>
160       <xs:element ref="dss:OptionalInputs" minOccurs="0"/>
161       <xs:element ref="dss:SignatureObject"/>
162     </xs:sequence>
163     <xs:attribute name="RequestID" type="xs:string" use="optional"/>
164     <xs:attribute name="Profile" type="xs:anyURI" use="optional"/>
165   </xs:complexType>
166 </xs:element>
```

#### 167 4.1.2 Element <ArchiveSubmitResponse>

168 The element <ArchiveSubmitResponse> MUST be produced by the server as a response to  
169 an <ArchiveSubmitRequest>.

170 <ArchiveIdentifier> [Optional]

171 The unique identifier assigned by the server to the archived signature object.

172

```
173 <xs:element name="ArchiveSubmitResponse">
174   <xs:complexType>
175     <xs:complexContent>
176       <xs:extension base="dss:ResponseBaseType">
177         <xs:sequence>
178           <xs:element name="ArchiveIdentifier" type="ArchiveIdentifier" minOccurs="0"/>
179         </xs:sequence>
180       </xs:extension>
181     </xs:complexContent>
182   </xs:complexType>
183 </xs:element>
```

184 If the submit operation is successfully processed, the server MUST include an  
185 <ArchiveIdentifier> element, including the unique identifier of the archived object, that  
186 SHOULD be retained by the client in order to be able to request subsequent operations with the  
187 archived object.

188 When there's any archive option that is not supported by the server, the server MUST refuse the  
189 request using the minor code `UnsupportedArchiveOption`.

## 190 4.1.3 Optional Inputs and Outputs

### 191 4.1.3.1 Optional Input <ArchivePolicy>

192 This optional input instructs the server to use a specific archive policy when archiving this  
193 signature. The archive policy SHOULD include relevant information for the archival process, like  
194 the retention period, the algorithms and key lengths, and the techniques (i.e. timestamping,  
195 timemarking) used to guarantee the integrity of the archived object over time.

196 When the server is unable to find an appropriate archive policy using the identifier requested by  
197 the client, the request MUST be rejected using minor code `ArchivePolicyNotFound`.

```
198 <xs:element name="ArchivePolicy" type="xades:ObjectIdentifierType"/>
```

### 199 4.1.3.2 Optional Input <RetentionPeriod>

200 This optional input instructs the server to archive the signature contained in the request,  
201 guaranteeing its integrity, for at least the duration included in the optional input starting at the  
202 moment of request processing.

203 The server MAY refuse the request when this optional input is used with the <ArchivePolicy>  
204 element, using minor code `IncompatibleRetentionPeriodInformation`.

```
205 <xs:element name="RetentionPeriod" type="xs:duration"/>
```

### 206 4.1.3.3 Optional Input <UpdateSignature>

207 This optional input instructs the server to update the signature prior to its archival, including  
208 important evidences that can be essential to survive repudiation / arbitration processes (by  
209 proving that the signature was valid at the time it was created (the signer's certificate was valid at  
210 the signature creation time)).

211

```
212 <xs:element name="UpdateSignature">  
213   <xs:complexType>  
214     <xs:attribute name="Type" type="xs:anyURI" use="required"/>  
215   </xs:complexType>  
216 </xs:element>
```

### 217 4.1.3.4 Optional Input <ArchiveMode>

218 This optional input instructs the server how to handle the archival of the signature. Two options  
219 are defined in the present document

- 220 • Opaque Mode (MANDATORY): The server handles the signature as an opaque binary object,  
221 applying the integrity protection measures over the whole object as a binary stream. If there's  
222 a need for signature updating prior to archival, the client MAY use the verifying protocol to  
223 update the signature.
- 224 • ES-A Mode (OPTIONAL): The server handles the signature as an archive electronic  
225 signature as defined in [XAdES] and [CAAdES], applying the integrity protection measures  
226 over the signature using the rules defined in the former specifications. The signature MUST  
227 be updated to an ES-A prior to archival.

228

229 When the server doesn't support the archive mode, it MUST refuse the request using the minor  
230 code `UnsupportedArchiveMode`.

231 Any other modes MAY be defined in further profiles.

232

```
233 <xs:element name="ArchiveMode" type="xs:anyURI"/>
```

234

## 235 4.2 The Archive Retrieval Protocol

236 This protocol allows the client to request the retrieval of the archived object in an immutable way  
237 (the archived object is not removed from the archive and its archive options remain unchanged).

### 238 4.2.1 Element <ArchiveRetrievalRequest>

239 The element <ArchiveRetrievalRequest> can be used to retrieve archived signatures  
240 without altering in any way the status of the archived object.

```
241 <xs:element name="ArchiveRetrievalRequest" type="ArchiveIdentifierRequest"/>
```

## 242 **4.2.2 Element <ArchiveRetrievalResponse>**

243

244 The element <ArchiveRetrievalResponse> MUST be produced by the server as a  
245 response to an <ArchiveRetrievalRequest>.

246 <dss:SignatureObject> [Optional]

247 The requested signature to retrieve (that is archived by the server).

```
248 <xs:element name="ArchiveRetrievalResponse">  
249   <xs:complexType>  
250     <xs:complexContent>  
251       <xs:extension base="dss:ResponseBaseType">  
252         <xs:sequence>  
253           <xs:element ref="dss:SignatureObject" minOccurs="0"/>  
254         </xs:sequence>  
255       </xs:extension>  
256     </xs:complexContent>  
257   </xs:complexType>  
258 </xs:element>
```

259 If the retrieval operation is successfully processed, the server MUST include an  
260 <dss:SignatureObject> element, including the archived signature being requested.

## 261 **4.2.3 Optional Inputs and Outputs**

262 No optional inputs/outputs are defined for this protocol.

## 263 **4.3 The Archive Delete Protocol**

264 This protocol allows the client to request the removal of the archived signature from the archive.

### 265 **4.3.1 Element <ArchiveDeleteRequest>**

266 The element <ArchiveDeleteRequest> can be used to request the removal of a signature  
267 from the archive.

```
268 <xs:element name="ArchiveDeleteRequest" type="ArchiveIdentifierRequest"/>
```

### 269 **4.3.2 Element <ArchiveDeleteResponse>**

270 The element <ArchiveDeleteResponse> MUST be produced by the server as a response to  
271 an <ArchiveDeleteRequest>.

272 <dss:SignatureObject> [Optional]

273 The requested signature to delete (that is archived by the server).

274

```
275 <xs:element name="ArchiveDeleteResponse">
```

276  
277  
278  
279  
280  
281  
282  
283  
284

```
<xs:complexType>  
  <xs:complexContent>  
    <xs:extension base="dss:ResponseBaseType">  
      <xs:sequence>  
        <xs:element ref="dss:SignatureObject" minOccurs="0"/>  
      </xs:sequence>  
    </xs:extension>  
  </xs:complexContent>  
</xs:complexType>
```

285

```
</xs:element>
```

286 If the delete operation is successfully processed, the server MUST include an  
287 <dss:SignatureObject> element containing the signature being removed from the archive.

### 288 4.3.3 Optional Inputs and Outputs

289 No optional inputs/outputs are defined for this protocol.

## 290 4.4 The Archive Modify Protocol

291 This protocol allows the client to modify the archival options of an archived signature.

### 292 4.4.1 Element <ArchiveModifyRequest>

293 The element <ArchiveModifyRequest> can be used to request the modification of the  
294 archival options of an archived signature.

295

```
<xs:element name="ArchiveModifyRequest" type="ArchiveIdentifierRequest"/>
```

### 296 4.4.2 Element <ArchiveModifyResponse>

297 The element <ArchiveModifyResponse> MUST be produced by the server as a response to  
298 an <ArchiveModifyRequest>.

299

```
<xs:element name="ArchiveModifyResponse" type="dss:ResponseBaseType"/>
```

300 When the server is not able to modify the archive options associated to an archived signature, it  
301 MUST refuse the request using the minor code `UnsupportedModification`.

### 302 4.4.3 Optional Inputs and Outputs

#### 303 4.4.3.1 Optional Input <ArchivePolicy>

304 As described in section 3.1.3.1.

#### 305 4.4.3.2 Optional Input <RetentionPeriod>

306 As described in section 3.1.3.2.

307 **4.4.3.3 Optional Input <UpdateSignature>**

308 As described in section 3.1.3.3.

309 **4.5 The Archive Verify Protocol**

310 This protocol allows the client to verify the integrity of an archived object.

311 **4.5.1 Element <ArchiveVerifyRequest>**

312 The element <ArchiveVerifyRequest> can be used to request the verification of the integrity  
313 of an archived object.

314 

```
<xs:element name="ArchiveVerifyRequest" type="ArchiveIdentifierRequest"/>
```

315 **4.5.2 Element <ArchiveVerifyResponse>**

316 The element <ArchiveVerifyResponse> MUST be produced by the server as a response to  
317 an <ArchiveVerifyRequest>.

318 

```
<xs:element name="ArchiveVerifyResponse" type="dss:ResponseBaseType"/>
```

319 When the server cannot determine the integrity status of the object (i.e. due to missing  
320 information), this request MUST be refused with the minor code  
321 IntegrityCheckUnavailable.

322 When the archived object is determined to have suffered an integrity violation, the server MUST  
323 this situation including a minor code IntegrityViolation in the response.

324 When the integrity is correctly verified, the server MUST inform the client using the minor code  
325 ValidIntegrityCheck.

326 **4.5.3 Optional Inputs and Outputs**

327 No optional inputs/outputs are defined for this protocol.

328 **4.6 Result Codes**

329 The URN used for the <dss:ResultMajor> elements is described in [DSSCore]. The URN  
330 used for the <dss:ResultMinor> elements MUST be  
331 urn:oasis:names:tc:dss:1.0:profiles:archive:resultminor: followed by the  
332 codes described below.

<dss:ResultMajor>	<dss:ResultMinor>	Description
RequesterError	UnsupportedArchiveOption	Any of the options selected by the client regarding the archival

		of the signature is not supported by the server.
RequesterError	ArchivePolicyNotFound	The policy selected by the client cannot be found by the server.
RequesterError	IncompatibleRetentionPeriodInformation	The client is using both an archival policy and a manually-selected retention period.
ResponderError	IntegrityCheckUnavailable	The server cannot determine whether the integrity of the archived object has been preserved or not.
ResponderError	IntegrityViolation	The archived signature has been tampered.
Success	ValidIntegrityCheck	The server has verified the integrity of the signature successfully.
ResponderError	UnsupportedModification	The server is not able to modify the archive options in the way requested by the client.

333

334

---

335 **5 Identifiers**

336 **5.1 Archive Modes**

337 The archive modes defined in this specification are:

Opaque/Raw Mode	urn:oasis:names:tc:dss:1.0:profiles:archive:modes:opaque
ES-A Mode	urn:oasis:names:tc:dss:1.0:profiles:archive:modes:ES-A

---

338 **6 References**

339 **6.1 Normative**

340 [TO BE DONE]

341

342

343

344

345

---

## Appendix A. Revision History

Rev	Date	By Whom	What
wd01	26/12/2005	Carlos González-Cadenas	Initial Version

347

---

## Appendix B. Notices

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

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

360 Copyright © OASIS Open 2003. *All Rights Reserved.*

361 This document and translations of it may be copied and furnished to others, and derivative works  
362 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
363 published and distributed, in whole or in part, without restriction of any kind, provided that the  
364 above copyright notice and this paragraph are included on all such copies and derivative works.  
365 However, this document itself does not be modified in any way, such as by removing the  
366 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS  
367 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
368 Property Rights document must be followed, or as required to translate it into languages other  
369 than English.

370 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
371 successors or assigns.

372 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
373 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO  
374 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY  
375 RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A  
376 PARTICULAR PURPOSE.

377