
Stapelsignaturen

Konzept

Dipl.-Ing. Martin Centner, Dipl.-Ing. Thomas Rössler

Zusammenfassung

Mit der Novelle der Signaturverordnung wurde für die Möglichkeit von Stapelsignaturen mit der Bürgerkarte bei nur einmaliger PIN-Eingabe vorgesorgt.

In diesem Konzept-Papier wird beschrieben, wie der Mechanismus zur Erstellung von Stapelsignaturen vor dem rechtlichen Hintergrund technisch unter Verwendung der Bürgerkarte umgesetzt werden kann.

Dabei werden vordergründig die notwendigen Änderungen auf Ebene der Bürgerkartenschnittstelle „Security Layer“ spezifiziert. Zusätzlich werden die Implikationen auf die Benutzerschnittstelle der Bürgerkartenumgebung (Bürgerkartensoftware) skizziert, um geeignete und rechtskonforme Wege der Darstellung von zu signierenden Dokumenten im Zuge der Stapelsignatur aufzuzeigen.

Inhaltsverzeichnis

1	Stapelsignaturen	2
2	Stapelsignaturen mit der Bürgerkartensoftware	2
3	Stapelsignaturen mit MOA-SS / MOA-SP	7
A	Entwurf Schema-Erweiterung Security-Layer für Bulk-Anfragen und Bulk-Antworten	9

1 Stapelsignaturen

Die Signaturverordnung (SigV) in ihrer geltenden Fassung¹ erlaubt das Auslösen des Signaturvorgangs für mehrere sichere elektronische Signaturen durch die einmalige Eingabe des Autorisierungscode.

SigV §4 - Technische Sicherheitserfordernisse für die Systemumgebung der Signaturerstellungseinheit bei sicheren Signaturen:

...

(2) Die Signaturfunktion in der Signaturerstellungseinheit des Signators darf nur nach Verwendung von Autorisierungscode (zB PIN-Eingabe, Fingerabdruck) auslösbar sein. Die Anzahl der Signaturen, die mit einer Autorisierung des Signators gegenüber seiner Signaturerstellungseinheit ausgelöst wird, muss dem Signator im Zeitpunkt des Auslösens des Signaturvorgangs bekannt sein. Derselbe Autorisierungscode darf nicht für unterschiedliche Anwendungen (zB Signatur- und Bankomatfunktion) verwendbar sein. Die eingegebenen Autorisierungscode dürfen von den verwendeten Systemelementen nicht über den Signaturvorgang hinaus im Speicher verbleiben. Eingabeerleichterungen bei wiederholter Eingabe von Autorisierungscode müssen ausgeschlossen sein. Das unbefugte Erfahren der Autorisierungscode muss durch dessen Gestaltung und durch Sperrmechanismen wirksam ausgeschlossen sein.

Dies erlaubt, den Signaturvorgang für einen Stapel von einzelnen und unabhängigen sicheren elektronischen Signaturen, mit dem gleichen Schlüsselpaar, in der Bürgerkartensoftware durch die einmalige Eingabe des PIN-Code auszulösen. Dabei muss jedoch die Möglichkeit zur Darstellung der zu signierenden Daten gewahrt bleiben. Alle Daten, die mit der Eingabe des PIN-Code signiert werden, müssen daher zum Zeitpunkt der Eingabe bereits vorliegen. Das „Freischalten“ der Bürgerkartensoftware für eine Anzahl von Signaturen oder für ein bestimmtes Zeitfenster, bevor die zu signierenden Daten überhaupt vorliegen, ist damit nicht zulässig.

Als Stapelsignaturen werden in Folge einzelne von einander unabhängige Signaturen bezeichnet, die auf dem gleichen Schlüsselpaar beruhen, und deren Erzeugung durch die einmalige Eingabe des Autorisierungscode (PIN-Code) ausgelöst werden.

2 Stapelsignaturen mit der Bürgerkartensoftware

Um mehrere Signaturerstellungsanfragen zu einem Stapel zusammenzufassen zu können, die alle mit dem gleichen Schlüsselpaar nach einmaliger Eingabe des PIN-Code abgearbeitet und signiert

¹ Änderung der Signaturverordnung (BGBl. II Nr. 527/2004) – http://ris1.bka.gv.at/authentic/findbgbl.aspx?name=entwurf&format=html&docid=COO_2026_100_2_116664

werden sollen, müssen in der Applikationsschnittstelle Security-Layer² entsprechende Mechanismen vorgesehen werden. Der nachfolgende Abschnitt spezifiziert die dazu notwendigen Erweiterungen der Security-Layer Schnittstelle.

2.1 Bulk-Anfrage

Bei einer Bulk-Anfrage werden mehrere einzelne Anfragen vom gleichen Typ zu einer Anfrage zusammengefasst. Bulk-Anfragen sind nur dann sinnvoll, wenn die Zusammenfassung mehrere Einzelanfragen eine Auswirkung auf die Interaktion mit dem Benutzer hat. In allen anderen Fällen können mehrere sequentielle Einzelanfragen verwendet werden.

Die Bulk-Anfrage enthält pro Einzelanfrage ein optionales Attribut, mit dem ein Name zur Anzeige durch die Bürgerkartensoftware übergeben werden kann.

Bulk-Anfragen werden durch entsprechende Bulk-Antworten beantwortet, wobei für jede in der Bulk-Anfrage enthaltene Einzelanfrage eine entsprechende Einzelantwort in der Bulk-Antwort vorhanden sein muss. Die Zuordnung von Einzelanfragen zur Einzelantworten basiert auf der Reihenfolge in der Bulk-Anfrage bzw. Bulk-Antwort. Um die Zuordnung von Einzelanfragen zu Einzelantworten zu erleichtern und die Selektion von einzelnen Anfragen oder Antworten in einer Bulk-Anfrage bzw. einer Bulk-Antwort über XML-Mechanismen zu vereinfachen, kann für jede Einzelanfrage ein Id-Attribut (vom Typ `xsd:ID`) gesetzt werden. Die Einzelantwort einer Bulk-Antwort enthält dann die Id der entsprechenden Einzelanfrage. Die Reihenfolge der Einzelantworten muss jedoch auch bei gesetztem Id-Attribut der Reihenfolge der Einzelanfragen entsprechen.

Bei der Fehlerbehandlung von Bulk-Anfragen ist zwischen Fehlern die Bulk-Anfrage selbst oder Einzelanfragen aus der Bulk-Anfrage betreffen zu unterscheiden. Tritt im Zuge der Abarbeitung einer Bulk-Anfrage ein Fehler bei der Bearbeitung einer Einzelanfrage auf, dann wird eine Fehlerantwort an die entsprechende Position in die Bulk-Antwort aufgenommen. Kann die gesamte Bulk-Anfrage nicht korrekt verarbeitet werden, dann wird anstelle einer Bulk-Antwort eine Fehlerantwort mit einem entsprechenden Fehlercode 3xxx zurückgegeben.

2.1.1 Signatur-Bulk-Anfrage

Signatur-Bulk-Anfragen bestehen aus einzelnen Signatur-Anfragen, wobei CMS- und XML-Signatur-Anfragen gemeinsam verwendet werden können. Die einzelnen Signatur-Anfragen müssen sich jedoch auf denselben Signaturschlüssel beziehen. Daher muss bei Bulk-Anfragen der Bezeichner des für die Signaturerstellung zu verwendenden Schlüssels (`s1:KeyboxIdentifier`) bei allen Einzelanfragen gleich sein. Stimmt der Bezeichner nicht bei allen Einzelanfragen überein, wird anstelle der Bulk-Antwort eine Fehlerantwort mit dem Fehlercode 3003 (XML-Struktur der Befehlsanfrage enthält eine unerlaubte Kombination aus optionalen Elementen oder Attributen) zurückgegeben.

² Applikationsschnittstelle Security-Layer – <http://www.buergerkarte.at/konzept/securitylayer/spezifikation/aktuell/core/Core.html>

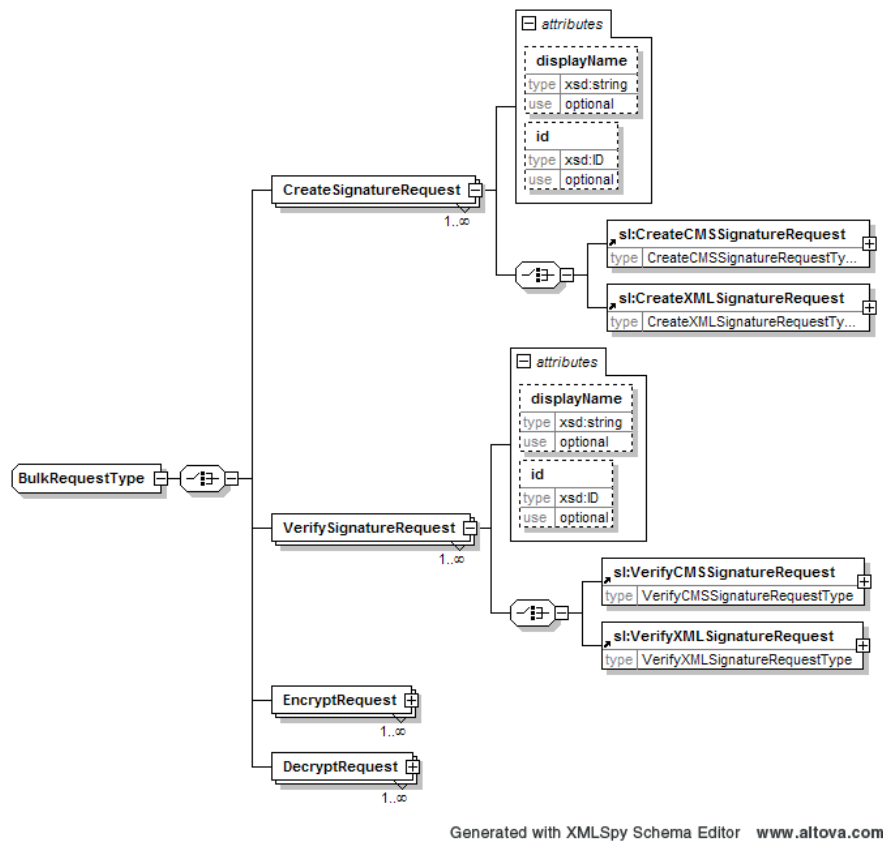


Abb. 1: Schema BulkRequest

Beispiel für eine Signatur-Bulk-Anfrage:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <BulkRequest xmlns="http://www.egiz.gv.at/namespaces/securitylayer-bulk/0.1-
  draft#" xmlns:sl="http://www.buergerkarte.at/namespaces/securitylayer/1.2#"
  xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
3   <CreateSignatureRequest displayName="Urlaubsantrag &quot;RÄ¶ssler&quot;">
4     <sl:CreateXMLSignatureRequest xmlns:sl="http://www.buergerkarte.at/
  namespaces/securitylayer/1.2#">
5       <sl:KeyboxIdentifier>SecureSignatureKeypair</sl:KeyboxIdentifier>
6       <sl:DataObjectInfo Structure="enveloping">
7         <sl:DataObject>
8           <sl:XMLContent>Ich bin ein einfacher Text.</sl:XMLContent>
9         </sl:DataObject>
10        <sl:TransformsInfo>
11          <sl:FinalDataMetaInfo>
12            <sl:MimeType>text/plain</sl:MimeType>
13          </sl:FinalDataMetaInfo>
14        </sl:TransformsInfo>
15      </sl:DataObjectInfo>
16    </sl:CreateXMLSignatureRequest>

```

```
17 </CreateSignatureRequest>
18 <CreateSignatureRequest displayName="Krankmeldung &quot;Meyer&quot;">
19   <sl:CreateXMLSignatureRequest xmlns:sl="http://www.buergerkarte.at/
   namespaces/securitylayer/1.2#">
20     <sl:KeyboxIdentifier>SecureSignatureKeypair</sl:KeyboxIdentifier>
21     <sl:DataObjectInfo Structure="enveloping">
22       <sl:DataObject>
23         <sl:XMLContent>Ich bin ein einfacher Text.</sl:XMLContent>
24       </sl:DataObject>
25       <sl:TransformsInfo>
26         <sl:FinalDataMetaInfo>
27           <sl:MimeType>text/plain</sl:MimeType>
28         </sl:FinalDataMetaInfo>
29       </sl:TransformsInfo>
30     </sl:DataObjectInfo>
31   </sl:CreateXMLSignatureRequest>
32 </CreateSignatureRequest>
33 </BulkRequest>
```

Bei einer Signatur-Bulk-Anfrage wird durch die Bürgerkartensoftware eine Liste der zu signierenden Daten angezeigt. Die Bürgerkartensoftware bietet die Möglichkeit, einzelne zu signierende Daten wie bei einer einzelnen Signatur-Anfrage darzustellen. Mit der Eingabe des PIN-Codes wird die Signatur für alle zu signierenden Daten ausgelöst. Die Bürgerkartensoftware kann dem Signator zusätzlich die Möglichkeit bieten, einzelne Daten von der Signatur auszunehmen. Anstelle der Signatur-Antwort wird in der Bulk-Antwort dann eine Fehler-Antwort mit dem Fehlercode 6001 (Abbruch durch den Bürger über die Benutzerschnittstelle) an entsprechender Stelle übermittelt.

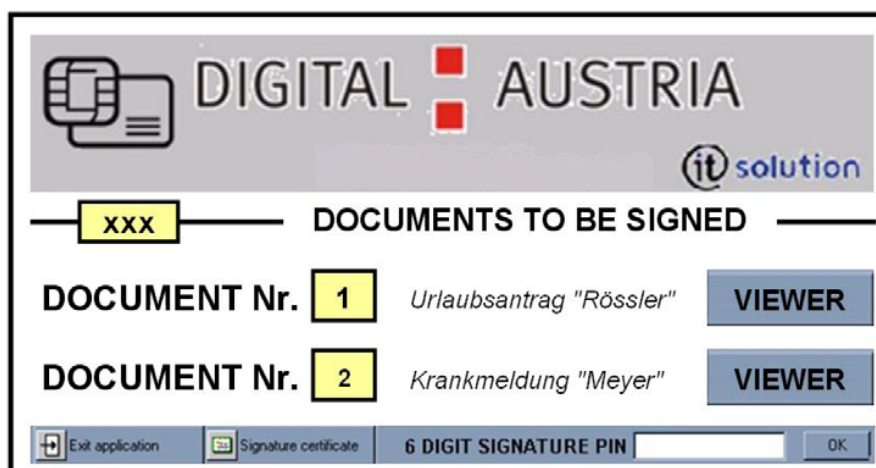


Abb. 2: Anzeige der zu signierenden Daten in der Bürgerkartensoftware

2.1.2 Signatur-Bulk-Antwort

Als Antwort auf eine Signatur-Bulk-Anfrage wird eine Signatur-Bulk-Antwort zurückgegeben. Die Reihenfolge der Einzelantworten in der Signatur-Bulk-Antwort entspricht der Reihenfolge der Einzelanfragen in der Signatur-Bulk-Anfrage.

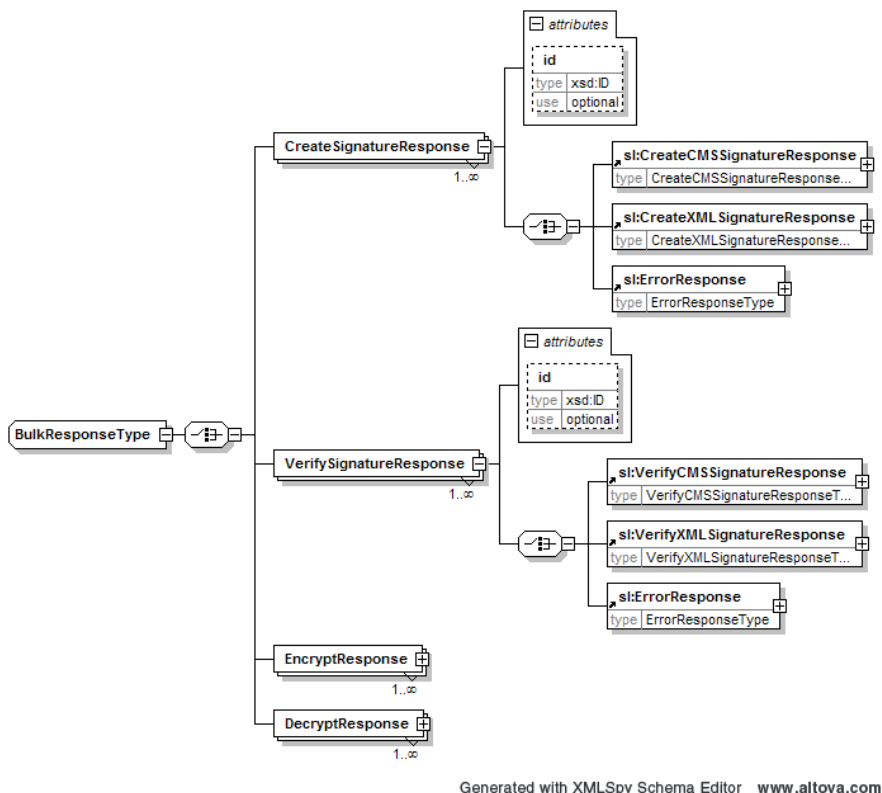


Abb. 3: Schema BulkResponse

Beispiel für eine Signatur-Bulk-Antwort:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <BulkResponse xmlns="http://www.egiz.gv.at/namespaces/securitylayer-bulk/0.1-
   draft#" xmlns:sl="http://www.buergerkarte.at/namespaces/securitylayer/1.2#"
   xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
3   <CreateSignatureResponse>
4     <sl:CreateXMLSignatureResponse>
5       <dsig:Signature>
6         ...
7       </dsig:Signature>
8     </sl:CreateXMLSignatureResponse>
9   </CreateSignatureResponse>
10  <CreateSignatureResponse>
11    <sl:CreateXMLSignatureResponse>
12      <dsig:Signature>

```

```
13     ...
14     </dsig:Signature>
15     </sl:CreateXMLSignatureResponse>
16     </CreateSignatureResponse>
17 </BulkResponse>
```

2.1.3 Signaturprüfungs-Bulk-Anfrage

Eine Signaturprüfungs-Bulk-Anfrage fasst mehrere CMS- und XML-Signaturprüfungsanfragen zu einer Anfrage zusammen. Bei einer Bulk-Anfrage wird das Ergebnis der Signaturprüfung für jede einzelne Signatur in einer Liste angezeigt. Der Benutzer erhält so die Möglichkeit, das Ergebnis und die signierten Daten jeder einzelnen Signatur zu überprüfen.

2.1.4 Signaturprüfungs-Bulk-Antwort

Das Ergebnis einer Signaturprüfungs-Bulk-Anfrage ist wiederum eine Signaturprüfungs-Bulk-Antwort in der die Reihenfolge der Einzelantworten der Reihenfolge der Einzelanfragen in der Bulk-Anfrage entspricht.

3 Stapelsignaturen mit MOA-SS / MOA-SP³

Eine Realisierung von Bulk-Anfragen für MOA-SS und MOA-SP scheint zunächst nahe liegend und wäre mit den oben vorgestellten Bulk-Anfragen und Bulk-Antworten analog möglich. Allerdings, würde sich die Behandlung von Bulk-Anfragen bei MOA-SS und MOA-SP in der derzeitigen Form nicht von der Behandlung von Einzelanfragen unterscheiden, da hier keine Interaktion mit dem Benutzer vorgesehen ist. Daher würde sich auch durch die Implementierung von Bulk-Anfragen kein Mehrwert ergeben.

Da der Kommunikationsaufwand von Einzelanfragen im Vergleich zu den in der Signaturerstellung und -prüfung notwendigen Schritten (Parsen, kryptographische Bearbeitung) geringer ist, erscheint die Bulk- Abfrage aus Prozess-Sicht auch ineffizienter, da ein paralleles Abarbeiten der Resultate nicht möglich ist, sondern die Antwort des Blocks abgearbeitet werden muss.

Berücksichtigt man zusätzlich, den Ressourcenbedarf für das Zusammenfassen von Einzelanfragen zu einer Bulk-Anfrage und für das Aufteilen der Bulk-Anfrage in Einzelanfragen zur Abarbeitung, dann scheint eine Realisierung von Bulk-Anfragen in MOA-SS und MOA-SP nicht zweckmäßig.

Beim Zusammenfassen sehr vieler Einzelanfragen zu einer Bulk-Anfrage ergeben sich allerdings sehr große XML-Datenstrukturen, bei deren Bearbeitung speziell Rücksicht auf den damit verbundenen Ressourcenbedarf zu legen ist. Zwar lässt sich das Verarbeiten der Bulk-Anfragen über

³ MOA-SS (Serversignatur) / MOA-SP (Signaturprüfung) – <http://www.cio.gv.at/online services/basicmodules/moa-spss/>

geeignete Mechanismen (z.B. Stream-Parsing der Bulk-Anfrage und Aufteilen in Einzelanfragen) relativ effizient implementieren, allerdings stoßen auch diese Mechanismen zum Aufteilen bzw. die Transport-Mechanismen zur Übermittlung der Anfragen an die Signatursoftware bei sehr großen Datenstrukturen an ihre Grenzen. Daher wird für die Verarbeitung von sehr großen Stapeln besonders im Serverbereich ein Mechanismus sinnvoll sein, bei dem nicht eine große Bulk-Anfrage, sondern die Einzelanfragen an die Signatursoftware übermittelt und von dieser zu einem Stapel zusammengefasst werden. Ein solcher Mechanismus benötigt dann auch geeignete Mechanismen um das Abarbeiten eines Stapels auszulösen und um die Ergebnisse an die Applikation zurück zu geben.

Anders ist die Situation bei eindeutiger Bindung der MOA-SS Signatur an den Benutzer⁴, wo Interaktionsmechanismen im Auslösevorgang absehbar sind. Die Umsetzung eines Stapelsignatur-Demonstrators wird daher im Umfeld des Teilprojekts *Fortgeschrittene Signaturen* analysiert.

⁴ Dies wird im Teilprojekt *Fortgeschrittene Signatur* behandelt.

A Entwurf Schema-Erweiterung Security-Layer für Bulk-Anfragen und Bulk-Antworten

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- DRAFT: Securitylayer, Bulk-Schnittstellenspezifikation -->
3 <!-- 15. 03. 2006, EGIZ E-Government Innovationszentrum -->
4 <xsd:schema xmlns:sl="http://www.buergerkarte.at/namespaces/securitylayer/1.2#"
  xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" xmlns:xsd="http://www.w3.
  org/2001/XMLSchema" xmlns="http://www.egiz.gv.at/namespaces/securitylayer-
  bulk/0.1-draft#" targetNamespace="http://www.egiz.gv.at/namespaces/
  securitylayer-bulk/0.1-draft#" elementFormDefault="qualified"
  attributeFormDefault="unqualified" version="1.2.1">
5 <xsd:import namespace="http://www.buergerkarte.at/namespaces/securitylayer
  /1.2#" schemaLocation="http://www.buergerkarte.at/konzept/securitylayer/
  spezifikation/aktuell/core/Core-1.2.xsd"/>
6 <!--#####
  -->
7 <!--# Bulk Request # -->
8 <!--#####
  -->
9 <xsd:element name="BulkRequest" type="BulkRequestType"/>
10 <xsd:complexType name="BulkRequestType">
11 <xsd:choice>
12 <xsd:element name="CreateSignatureRequest" maxOccurs="unbounded">
13 <xsd:complexType>
14 <xsd:choice>
15 <xsd:element ref="sl:CreateCMSSignatureRequest"/>
16 <xsd:element ref="sl:CreateXMLSignatureRequest"/>
17 </xsd:choice>
18 <xsd:attribute name="displayName" type="xsd:string" use="optional"/>
19 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
20 </xsd:complexType>
21 </xsd:element>
22 <xsd:element name="VerifySignatureRequest" maxOccurs="unbounded">
23 <xsd:complexType>
24 <xsd:choice>
25 <xsd:element ref="sl:VerifyCMSSignatureRequest"/>
26 <xsd:element ref="sl:VerifyXMLSignatureRequest"/>
27 </xsd:choice>
28 <xsd:attribute name="displayName" type="xsd:string" use="optional"/>
29 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
30 </xsd:complexType>
31 </xsd:element>
32 <xsd:element name="EncryptRequest" maxOccurs="unbounded">
33 <xsd:complexType>
34 <xsd:choice>
35 <xsd:element ref="sl:EncryptCMSRequest"/>
36 <xsd:element ref="sl:EncryptXMLRequest"/>
37 </xsd:choice>
38 <xsd:attribute name="displayName" type="xsd:string" use="optional"/>
39 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
40 </xsd:complexType>
```

```
41     </xsd:element>
42     <xsd:element name="DecryptRequest" maxOccurs="unbounded">
43         <xsd:complexType>
44             <xsd:choice>
45                 <xsd:element ref="sl:DecryptCMSRequest"/>
46                 <xsd:element ref="sl:DecryptXMLRequest"/>
47             </xsd:choice>
48             <xsd:attribute name="displayName" type="xsd:string" use="optional"/>
49             <xsd:attribute name="id" type="xsd:ID" use="optional"/>
50         </xsd:complexType>
51     </xsd:element>
52 </xsd:choice>
53 </xsd:complexType>
54 <!--#####
55     <!--# Bulk Response # -->
56 <!--#####
57 <xsd:element name="BulkResponse" type="BulkResponseType"/>
58 <xsd:complexType name="BulkResponseType">
59     <xsd:choice>
60         <xsd:element name="CreateSignatureResponse" maxOccurs="unbounded">
61             <xsd:complexType>
62                 <xsd:choice>
63                     <xsd:element ref="sl:CreateCMSSignatureResponse"/>
64                     <xsd:element ref="sl:CreateXMLSignatureResponse"/>
65                     <xsd:element ref="sl:ErrorResponse"/>
66                 </xsd:choice>
67                 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
68             </xsd:complexType>
69         </xsd:element>
70         <xsd:element name="VerifySignatureResponse" maxOccurs="unbounded">
71             <xsd:complexType>
72                 <xsd:choice>
73                     <xsd:element ref="sl:VerifyCMSSignatureResponse"/>
74                     <xsd:element ref="sl:VerifyXMLSignatureResponse"/>
75                     <xsd:element ref="sl:ErrorResponse"/>
76                 </xsd:choice>
77                 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
78             </xsd:complexType>
79         </xsd:element>
80         <xsd:element name="EncryptResponse" maxOccurs="unbounded">
81             <xsd:complexType>
82                 <xsd:choice>
83                     <xsd:element ref="sl:EncryptCMSResponse"/>
84                     <xsd:element ref="sl:EncryptXMLResponse"/>
85                     <xsd:element ref="sl:ErrorResponse"/>
86                 </xsd:choice>
87                 <xsd:attribute name="id" type="xsd:ID" use="optional"/>
88             </xsd:complexType>
89         </xsd:element>
90         <xsd:element name="DecryptResponse" maxOccurs="unbounded">
91             <xsd:complexType>
```

```
92     <xsd:choice>
93         <xsd:element ref="sl:DecryptCMSResponse"/>
94         <xsd:element ref="sl:DecryptXMLResponse"/>
95         <xsd:element ref="sl:ErrorResponse"/>
96     </xsd:choice>
97     <xsd:attribute name="id" type="xsd:ID" use="optional"/>
98 </xsd:complexType>
99 </xsd:element>
100 </xsd:choice>
101 </xsd:complexType>
102 </xsd:schema>
```