Spezifikation MOA ID 2004-03-15	Konvention MOA ID 1.2 Öffentlicher Entwurf

Bezeichnung	Spezifikation Module für Online Applikationen - ID
Kurzbezeichnung	MOA ID 1.2
Version	1.2
Datum	2004-03-15
Dokumentenklasse	Konvention
Dokumentenstadium	Entwurf öffentlich
Kurzbeschreibung	Dieses Dokument spezifiziert ein Servermodul für die weitere Umsetzung der E-Government-Strategien. Das Basismodul ID dient zur sicheren Identifikation und Authentifikation von Benutzern mittels Bürgerkarte. Dabei wird die digitale Signatur zum sicheren Beweis der Authentizität verwendet. Das Design sieht vor, dass Online-Applikationen die Identifikation und Authentifikation an MOA ID auslagern können. Mit dem vorliegenden Dokument wird eine weitere Vereinheitlichung des E-Governments und sowie eine Vereinfachung der Umsetzung von modernen Online-ApplikationenAnwendungen angestrebt.
Autoren	ARGE Spezifikation MOA; Ansprechpartner: Stabsstelle IKT-Strategie des Bundes, Bundeskanzleramt. *Email:* *Rudolf.Schamberger@cio.gv.at* *Gregor.Karlinger@cio.gv.at* Bundesministerium für Finanzen *Email:* *Ludwig.Moser@bmf.gv.at*
Arbeitsgruppe	ARGE Spezifikation MOA

Inhaltsverzeichnis

Dokumenten Information

Versionen

Version	Datum	Beschreibung	Autor
1.0	8.10.2002	Version 1.0	ARGE
1.1	30.06.2003	Version 1.1 • Funktionserweiterung MOA-ID Ergänzung um BKU Auswahl. • Editoriale Ergänzungen	ARGE
1.2	15.03.2004	Version 1.2 • Funktionserweiterungen MOA-ID • Umstellung auf Stammzahl und bPK	ARGE

Bundeskanzleramt Republik Österreich

Bundesministerium für Finanzen

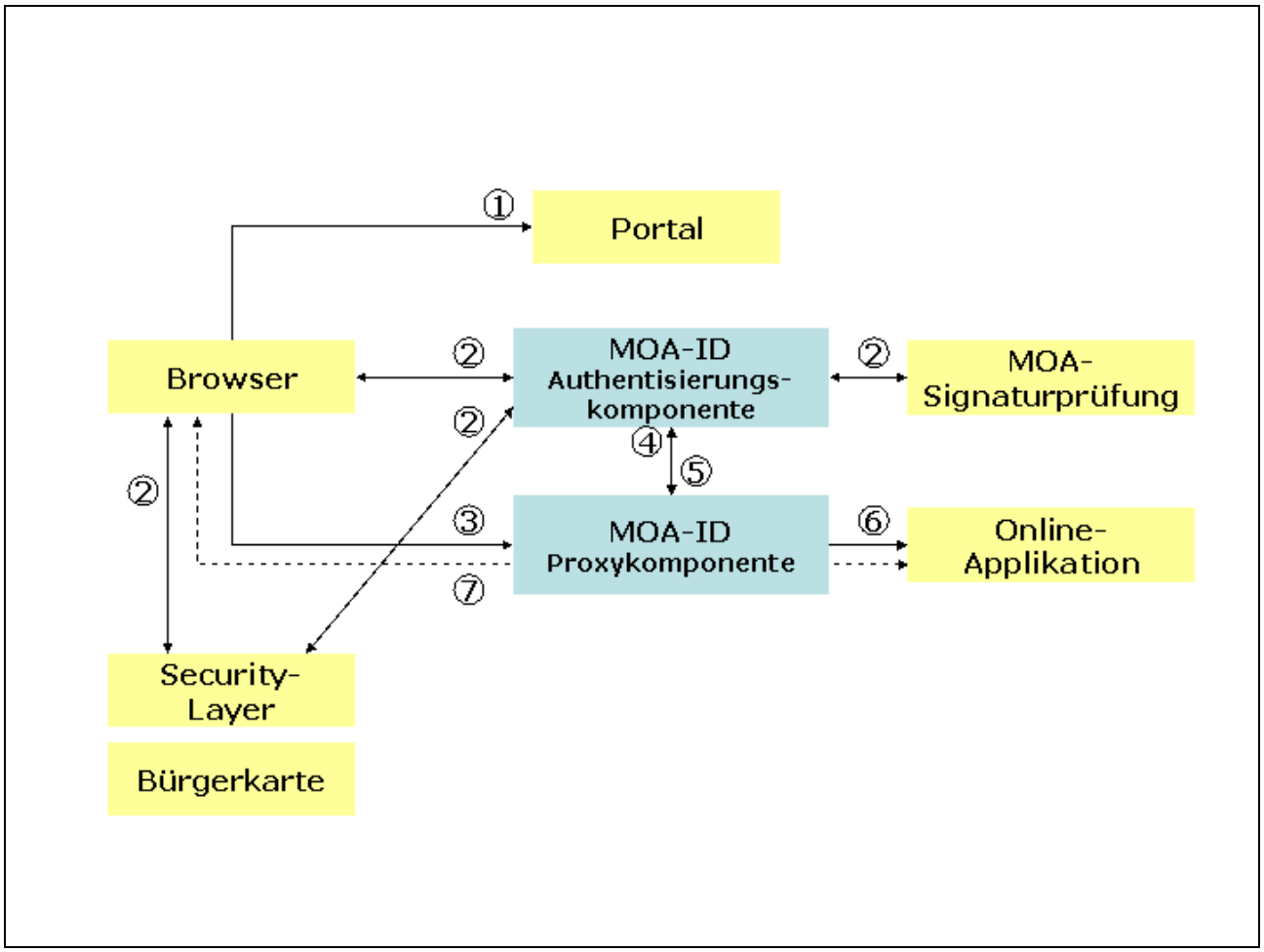
1 Anwendungsfälle

Die primäre Funktion von MOA-ID ist die sichere Identifikation und Authentifikation von Benutzern unter Anwendung des Konzepts Bürgerkarte. Dabei werden die Personenbindung und die Signaturfunktion der Bürgerkarte verwendet um den Benutzer sicher zu authentifizieren. MOA-ID überprüft die benutzerbezogenen Daten die aus der s.g. Personenbindung der Bürgerkarte stammen und kann diese bzw. Teile daraus in Folge beliebigen Onlineanwendungen zur Verfügung stellen. Als weitere wichteige Funktion berechnet MOA-ID seit der Version 1.2 die sogennante bereichssspezifische Personenkennung (bPK).

Die Anwendungsfälle, die MOA abdecken muss sind in der folgenden Liste zusammengefasst:

- Auswahl einer Bürgerkartenumgebung (BKU-Auswahl)
- Kommunikation mit dem Browser des Benutzers und der Bürgerkartenumgebung
- Authentisierung eines Benutzers (Bürger oder Behördenmitarbeiter) mittels Bürgerkarte und Security-Layer
- Berechnung der Bereichsspezifischen Personenkennung (bPK).
- Fehlerhandling
- Konfiguration von MOA-ID
- Weitergabe der Anmeldedaten an nachfolgende Online-Anwendungen.

2 Überblick



MOA-ID besteht aus zwei Komponenten, der Authentisierungskomponente und der Proxykomponente, die mit dem Browser des Benutzers und Online-Applikationen Schnittstellen aufweisen. Mit dem Security-Layer des Benutzers wird sowohl über den Browser als auch direkt kommuniziert.

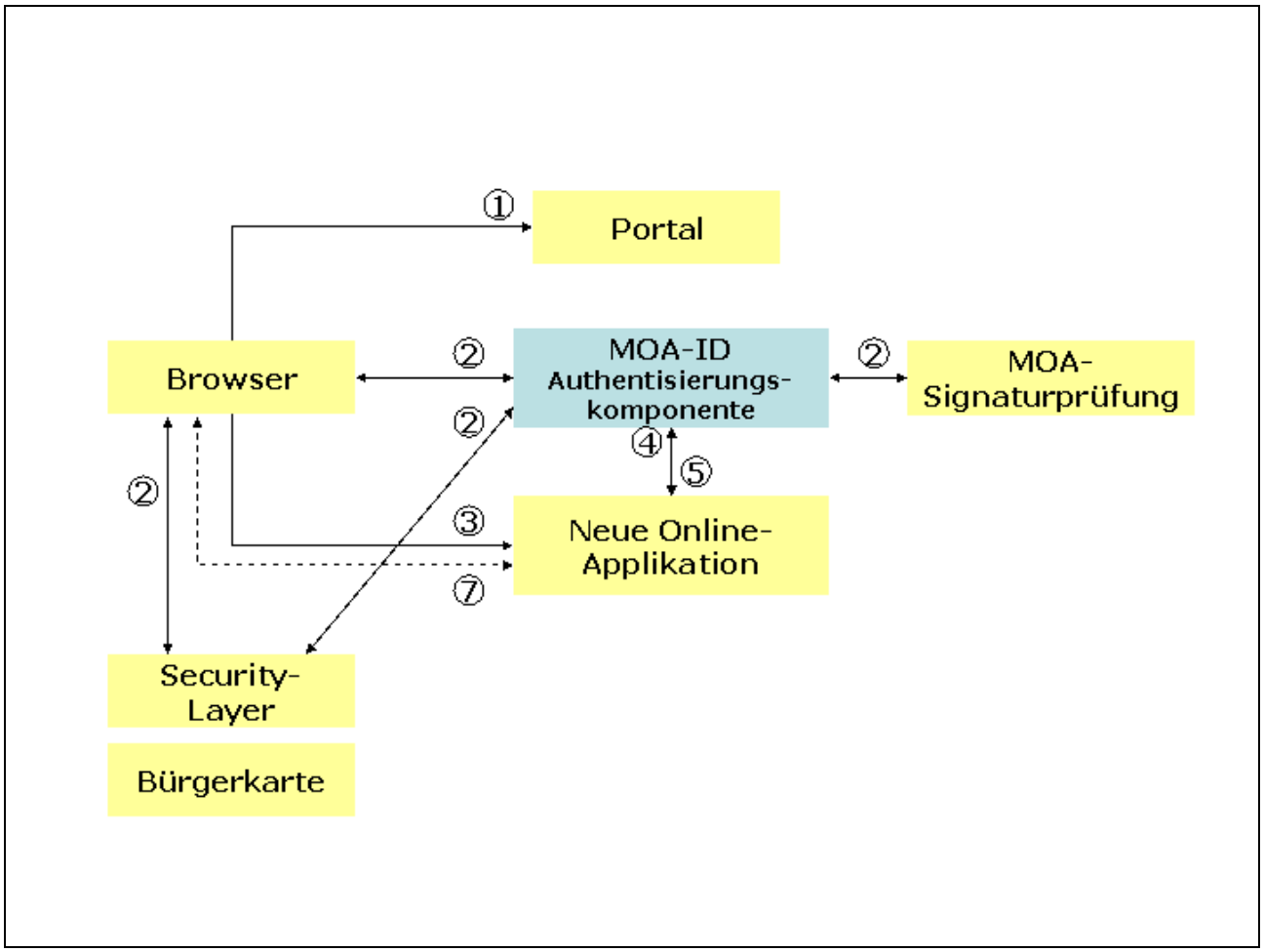
Die Authentisierungskomponente führt die eigentliche Authentisierung des Benutzers durch und übergibt der Proxykomponente die Anmeldedaten. Die Proxykomponente übernimmt die Anmeldedaten von der Authentisierungskomponente, führt die Anmeldung an der Online Applikation durch und schleust in der Folge Daten an die Online-Applikation und Daten an den Benutzer durch. Diese beiden Komponenten können auf unterschiedlichen Rechnern oder auf dem gleichen Rechner eingesetzt werden.

Die folgende Tabelle beschreibt einen typischen An- und Abmeldevorgang in mehreren Schritten (die Nummerierung der Schritte entspricht den Nummern in der oben gezeigten Grafik):

Schritt	Beschreibung
1.	Der Benutzer verbindet sich zu einem Web-Portal, über das die verfügbaren Online-Applikationen (OA) erreichbar sind. Jeder Link zu einer OA verweist auf die Authentisierungskomponente mit zwei Parameter: • URL der tatsächlich zu verwendeten Applikation: z.B. `https://app1.gv.at/Beispiel` • Geschäftsbereich: als String kodiert
2.	Der Benutzer verbindet sich über HTTPS mit der Authentisierungskomponente, die die Authentisierung des Benutzers durchführt:
2.1.	MOA-ID-AUTH bietet dem Benutzer optional eine Auswahl von verfügbaren Bürgerkartenumgebungen an.
2.2.	Die Authentisierungskomponente erzeugt eine HTML-Seite zum Auslesen der Personen-Bindung, mit folgenden Inhalten: • `<FORM>` mit Security-Layer `<InfoboxReadRequest>`-Kommando zum Auslesen der Personenbindung Diese HTML-Seite wird an den Browser geschickt.
2.3.	Der Browser schickt das Kommando per HTTP POST an den Security-Layer. Der Security-Layer • liest die Personenbindung • sendet die Personenbindung über die angegebene `DataURL` (siehe [SecLayer1.1]) an die Authentisierungskomponente zurück.
2.4.	Die Authentisierungskomponente sendet, als Antwort auf die Personenbindung, eine XML Antwortseite, die direkt das Kommando zum Signieren des von der Authentisierungskomponente generierten AUTH-Blocks enthält.
2.5.	Der Request wird vom Security-Layer verarbeitet (Siehe [SecLayer1.1-Bindung] Abschnitt 3.3.2 Punkt 5b). Die signierten Daten werden direkt and die `DataURL` zurückgesendet.
2.6.	Die Authentisierungskomponente überprüft den signierten AUTH-Block und legt für den Benutzer die Anmeldedaten (siehe *Abschnitt 4.5*) an.
2.7.	Ist der obige Authentisierungsvorgang erfolgreich, dann wird eine Redirect-Seite mit einem SAML-Artifact (siehe *Abschnitt 4.3*) erstellt und zum Browser gesendet.
3.	Der Browser führt das Redirect zur angegebenen URL (Proxykomponente von MOA-ID) durch. Als Parameter wird das eindeutige SAML-Artifact übergeben, mit dem die nachfolgende Applikation die Anmeldedaten von der Authentisierungskomponente abfragen kann.
4. 5.	Die nachfolgende Applikation (Proxykomponente) verwendet dieses eindeutige SAML-Artifact, um die Anmeldedaten via SOAP von der Authentisierungskomponente zu erhalten. Danach werden die Anmeldedaten in der Authentisierungskomponente gelöscht.
6.	Die Proxykomponente liest eine Konfigurationsdatei die beschreibt, wie die Anmeldedaten an die nachfolgende Applikation übergeben werden müssen, und meldet den Benutzer bei der Applikation an.
7.	In der Folge werden die Antworten der (stateless) OA an den Benutzer weitergeleitet und die Anfragen des Benutzers an die OA weitergeleitet.

Im Falle einer neu entwickelten Online-Applikation kann die Funktionalität und die Schnitt-

stellen der Proxykomponente von der Online-Applikation übernommen werden.



Im Folgenden werden die Proxykomponente und die neu entwickelte Online-Applikation als *nachfolgende Applikation* bezeichnet.

3 Allgemeine Anforderungen

3.1 Unterstützte Plattformen

Es muss Java Runtime Environment ab Version 1.3 unterstützt werden.

3.2 TLS Authentisierung

Sämtliche Schnittstellen zwischen den Komponenten und Applikationen können mittels TLS [TLS] authentisiert werden:

Schnittstelle	TLS Authentisierung	Beschreibung
Browser - Authentisierungskomponente	Server	verpflichtend
Browser – nachfolgende Applikation	Server	verpflichtend
Authentisierungskomponente – MOA Signaturprüfung [MOA SP-SS]	Server, Client	konfigurierbar
nachfolgende Applikation - Authentisierungskomponente	Server, Client	konfigurierbar
Proxykomponente – Online-Applikation	Server, Client	konfigurierbar

3.3 Skalierbarkeit und Verfügbarkeit

MOA-ID muss skalierbar und auf einen 7 * 24h Betrieb ausgelegt sein.

3.4 Namespace

Alle in dieser Spezifikation definierten XML-Elemente sind dem Namespace http://reference.e-government.gv.at/namespace/moa/20020822# zugeordnet.

4 Authentisierungskomponente

Die Funktionsweise der Authentisierungskomponente orientiert sich an dem in [SAML-Binding] im Abschnitt 4.1.1 („Browser/Artifact Profile of SAML“) beschriebenen Vorgang, erweitert durch die Erstellung des vom Benutzer signierten AUTH-Blocks mit dem Security-Layer [SecLayer1.1].

4.1 Schnittstelle zur verweisenden Webseite

Die Authentisierungskomponente wird immer durch eine andere (verweisende) Webseite aufgerufen. Diese Webseite kann z.B. Teil eines Portals sein.

Der Aufruf der Authentisierungskomponente durch die verweisende Webseite (aufrufende Applikation) erfolgt durch das Aufrufen einer von der Authentisierungskomponente bereitgestellten URL durch den Browser des Benutzers.

4.1.1 StartAuthentication

Der Aufruf erfolgt durch einen Verweis der Form:

```
<a href="https://<moa-id-server-und-pfad>/StartAuthentication?
        Target=<geschäftsbereich>&
        OA=<oa-url>&
        Template=<template-url>">
```

Wobei:

- `<moa-id-server-und-pfad>` angibt, wo MOA-ID-AUTH installiert ist.
- `Target=<geschäftsbereich>` (siehe [GB], [GB-BK]) angibt, für welches Verfahren der Benutzer authentisiert werden soll.
- `OA=<oaurl>` mittels URL angibt, an welche Seite in der nachfolgenden Applikation der Benutzer nach erfolgreicher Authentisierung weitergeleitet werden muss. Die URL (bzw. ein eindeutiger Teil davon) dient außerdem als Schlüssel für die Auswahl der applikationsspezifischen Konfiguration in der Authentisierungskomponente (siehe Abschnitt 4.6 *Konfiguration*).
- `Template=<template-url>` eine optionale HTML-Vorlage für der Anmeldeseite von MOA-ID-AUTH angibt, über die der Bürger den Authentisierungsvorgang startet. Über diesen Parameter kann das Aussehen der Anmeldeseite an das Aussehen der Online-Applikation angepasst werden.

  Ein Template für die Anmeldeseite von MOA-ID-AUTH kann aus folgender Grundstruktur aufgebaut werden:

```
<form name="CustomizedForm" action="<BKU>" method="post">
 <input type="hidden"
        name="XMLRequest"
        value="<XMLRequest>"/>
 <input type="hidden"
        name="DataURL"
        value="<DataURL>"/>
 <input type="submit" value="Bürgerkarte lesen"/>
</form>
<form name="CustomizedInfoForm"
 action="<BKU>"
 method="post">
 <input type="hidden"
        name="XMLRequest"
        value="<CertInfoXMLRequest>"/>
 <input type="hidden"
        name="DataURL"
        value="<CertInfoDataURL>"/>
Hier finden Sie weitere Informationen
zur Überprüfung der Zertifikate.<br/>
 <input type="submit" value="Weitere Info"/>
</form>
```

Innerhalb dieser `<form>`-Elemente können Texte, Beschriftungen und Styles modifiziert werden, und es können zusätzliche Elemente darin aufgenommen werden.

Die vorgegebene Grundstruktur ist aber in jedem Fall einzuhalten, und es müssen die speziellen Tags `<BKU>` (kommt 2x vor), `<XMLRequest>`, `<DataURL>`, `<CertInfoXMLRequest>` und `<CertInfoDataURL>` darin enthalten sein.

Die Weiterleitung an die nachfolgende Applikation erfolgt durch einen HTTP-Redirect (Status Code 302).

4.1.2 BKU-Auswahl

MOA-ID-AUTH bietet die Möglichkeit, die Bürgerkartenumgebung (BKU) auszuwählen, über die in weiterer Folge die Bürgerkarte ausgelesen wird. Der Aufruf erfolgt dann durch einen Verweis der Form:

```
<a href="https://<moa-id-server-und-pfad>/SelectBKU?
        Target=<geschäftsbereich>&
        OA=<oa-url>&
        Template=<template-url>&
        BKUSelectionTemplate=<bku-template-url>">
```

Wobei:

- BKUSelectionTemplate=<bku-template-url> eine optionale HTML-Vorlage für der BKU-Auswahlseite von MOA-ID-AUTH angibt. Über diesen Parameter kann das Aussehen der BKU-Auswahlseite an das Aussehen der Online-Applikation angepasst werden.

  Ein Template für die BKU-Auswahl von MOA-ID-AUTH kann aus folgender Grundstruktur aufgebaut werden:

```
<form name="CustomizedForm" method="post" action="<StartAuth>">
 <BKUSelect>
 <input type="submit" value="Auswählen"/>
</form>
```

Innerhalb dieser <form>-Elemente können Texte, Beschriftungen und Styles modifiziert werden, und es können zusätzliche Elemente darin aufgenommen werden.

Auch dabei ist die vorgegebene Grundstruktur einzuhalten, die speziellen Tags <StartAuth> und <BKUSelect> sind verpflichtend.

4.2 Schnittstelle zur Bürgerkarte

4.2.1 Information bezüglich Wurzelzertifikate

Für die Sicherheit des Identifikationsmoduls ist es wesentlich, dass der Benutzer zuverlässig die Authentizität der Authentisierungskomponente überprüfen kann. Diese Authentizität wird durch die Verwendung des TLS-Protokolls sichergestellt. Da TLS-Implementierung in gängigen Web-Browsern viele Wurzelzertifikate als vertrauenswürdige vorinstalliert haben, ist es notwendig, dass der Benutzer die Richtigkeit der verwendeten Serverzertifikate manuell überprüft. Der folgende Ablauf soll sicherstellen, dass der Benutzer diese Überprüfung durchführt bzw. bewusst auf diese Überprüfung verzichtet.

Bevor die Personenbindung ausgelesen wird, bekommt der Benutzer eine Möglichkeit (mittels Info-Button) eine signierte Informationsseite von der Authentisierungskomponente zu laden, welche automatisch im Secure-Viewer des Security-Layers sicher angezeigt wird (sofern die SL-Implementierung dies unterstützt). Diese Informationsseite kann eine Beschreibung beinhalten, wie der Benutzer die Authentizität des Servers mittels TLS überprüfen kann, und wird vom CIO mit einem Zertifikat signiert, dessen Wurzelzertifikat in der SL-Implementierung als vertrauenswürdig installiert ist (z.B. ein A-Trust Zertifikat). Die Signatur und der Zertifikatspfad zum Wurzelzertifikat sind vom Benutzer zu überprüfen.

4.2.2 Abfrage der Personenbindung

Diese Schnittstelle dient zum Abfragen der Personenbindung vom Security-Layer (siehe *Abschnitt 2 Überblick, Schritt 2.2*).

Die Authentisierungskomponente erzeugt eine HTML-Seite die zumindest das folgende <LINK>-, <FORM>- und <SCRIPT>-Element enthält:

```
<LINK rel="stylesheet" type="text/css" href="<css-url>">

<FORM name="getIdentityLinkForm"
      action=<bku-url>
      method="POST">
    <INPUT type="hidden" name="XMLRequest"/>
    <INPUT type="hidden" name="DataURL"
           value=<generateSignAuthBlockRequestServletID>?
                 Target=<geschäftsbereich>?Template=<template-url>/>
    <INPUT type="submit" value="Auslesen der Personenbindung"/>
</FORM>

<A href='http://<Infopage>'> Wichtige Information zu Zertifikaten</A>
```

<bku-url> ist die URL, die bei der BKU-Auswahl returniert wird (bzw. http://localhost:3495/http-security-layer-request als Default-Wert).

<template-url> ist die in *Abschnitt 4.1 Schnittstelle zur verweisenden Webseite* beschriebene URL.

Das Feld XMLRequest muss folgenden Wert beinhalten:

```
<InfoboxReadRequest
  xmlns="http://www.buergerkarte.at/namespaces/securitylayer/20020225#">
  <InfoboxIdentifier>
    IdentityLink
  </InfoboxIdentifier>
  <BinaryFileParameters/>
</InfoboxReadRequest>
```

Das Feld DataURL enthält eine URL, welche einen eindeutigen, für jeden Auslesevorgang unterschiedlichen Identifier beinhaltet. Diese URL verweist auf ein Servlet, welches die Personenbindung entgegennimmt und in Folge eine Antwort darauf erzeugt (Beschreibung siehe *Abschnitt 4.2.3*).

4.2.3 Signieren des AUTH-Block

Das im *Abschnitt 4.2.2 Abfrage der Personenbindung* über die DataURL referenzierte Servlet nimmt die Personenbindung entgegen.

Ist der <InfoboxReadRequest> erfolgreich und kann die Signatur der Personenbindung erfolgreich überprüft werden, so erzeugt das Servlet die XML-Antwortseite mit dem Befehl zum Signieren des AUTH-Blocks (siehe *Abschnitt 4.4*). Ist der <InfoboxReadRequest> nicht erfolgreich wird eine HTML-Seite mit einer Fehlerbeschreibung erzeugt und der Authentisierungsvorgang abgebrochen. Die erzeugte XML- bzw. HTML-Seite wird an den Security-Layer retourniert, der je nach Content-Type der Antwort entsprechend reagiert. (siehe [SecLayer1.1-Bindung] Abschnitt 3.3.2 Punkt 5)

Folgende Schnittstelle definiert die XML-Seite, die dazu dient den AUTH-Block mit dem Security-Layer zu signieren (siehe *Abschnitt 2 Überblick, Schritt 2.4*).

XML-Antwortseite zum Signieren des AUTH-Blocks:

```
<CreateXMLSignatureRequest
  xmlns="http://www.buergerkarte.at/namespaces/securitylayer/20020831#">
  <KeyboxIdentifier>CertifiedKeypair</KeyboxIdentifier>
  <DataObjectInfo Structure="enveloping">
    <DataObject>
      <XMLContent>
        [der in Abschnitt 4.4 beschriebene AUTH-Block]
      </XMLContent>
    </DataObject>
    <TransformsInfo>
      [mögliche Transformationen]
    </TransformsInfo>
  </DataObjectInfo>
<SignatureInfo>
  [Spezifikation wo die Signatur eingebettet werden soll]
</SignatureInfo>
</CreateXMLSignatureRequest>
```

<TransformsInfo> enthält die Transformationen, die angewendet werden können, um die XML-Struktur des AUTH-Blocks in ein für den Benutzer verständliches Format zu konvertieren, das im Viewer des Security-Layers angezeigt werden kann. Es müssen sämtliche in *Abschnitt 4.4 AUTH-Block* spezifizierten Informationen dem Benutzer angezeigt werden. Diese Transformationen müssen für alle Security-Layer-Implementierungen, die unterstützt werden, konfiguriert werden können. Alle diese Transformationen müssen dem Security-Layer zur Verfügung gestellt werden, sodass dieser die geeignete Transformation auswählen kann.

Ist die Erstellung bzw. Prüfung der Signatur nicht erfolgreich wird eine HTML-Seite mit einer Fehlerbeschreibung erzeugt und der Authentisierungsvorgang abgebrochen. Die erzeugte HTML-Seite wird an den Browser retourniert.

Ist die Erstellung und Prüfung der Signatur erfolgreich, wird entsprechend der SAML Spezifikation ein Redirect, welches das SAMLartefact (siehe *Abschnitt 4.3*) enthält, vom Server über die Security-Layer-Implementierung an den Browser retourniert.

Als `DataURL` dürfen nur URLs spezifiziert werden, die die Authentisierungskomponente referenzieren.

4.3 Schnittstelle zur nachfolgenden Applikation

Die nachfolgende Applikation wird durch den HTTP-Redirect von der Authentisierungskomponente aufgerufen, und stellt die Authentisierung an der Applikation selbst sicher. Der HTTP-Request, mit dem die nachfolgende Applikation aufgerufen wird, hat die Form:

```
GET https://<oa-url>?Target=<geschäftsbereich>&SAMLArtifact=<saml-artifact>
```

Wobei:

- `<oa-url>` die URL angibt, an der die nachfolgende Applikation aufgerufen werden kann, um den Benutzer zu authentisieren. Sie ist identisch mit dem Wert des beim Aufruf der Authentisierungskomponente übergebenen Parameters OA.
- `Target=<geschäftsbereich>` den Geschäftsbereich angibt, für den der Benutzer authentisiert werden soll.
- `SAMLArtifact=<saml-artifact>` das SAML-Artifact übergibt, das während des Authentisierungsvorganges mit der Authentisierungskomponente erzeugt wurde, und anhand dessen die nachfolgende Applikation die Identität des Benutzers verifizieren kann. Die Codierung des SAML-Artifact ist in [SAML-Binding], Abschnitt 4.1.1.8 beschrieben.

Die nachfolgende Applikation muss mit Hilfe des SAML-Artifact einen SOAP-Request an die Authentisierungskomponente zusammenstellen, wie in [SAML-Binding], Abschnitte 3 und 4.1.1.6 beschrieben. Bei erfolgreicher Rückmeldung der SOAP-Response, wie in Abschnitt 4.1.1.6 in [SAML-Binding] beschrieben, ist der Benutzer an der nachfolgenden Applikation angemeldet.

4.4 AUTH-Block

Der AUTH-Block ist eine Datenstruktur, die von der Authentisierungskomponente und vom Security-Layer mittels eines XSLT Stylesheets aufgebaut wird, und enthält folgende Informationen:

- Vorname, Nachname aus Personenbindung
- URL der Authentisierungskomponente
- URL und Geschäftsbereich der nachfolgenden Applikation
- Aktuelle Zeit

Die Personenbindung ist nicht im AUTH-Block enthalten weil:

- die Stammzahl nicht gespeichert werden darf
- die Daten der Personenbindung (öffentliche Schlüssel, Stammzahl) für den Benutzer wenig aussagekräftig sind und daher eine Anzeige im Secure Viewer des Security-Layer nicht sinnvoll ist

- die Daten der Personenbindung würden drei mal zwischen Security-Layer und Authentisierungskomponente übertragen werden

Der AUTH-Block wird als SAML-Assertion [SAML-Assertions] dargestellt und vom Benutzer signiert.

Das <saml:Assertion>-Element muss folgende verpflichtende Attribute enthalten:

Name	Beschreibung
MajorVersion	Muss den Wert 1 haben
MinorVersion	Muss den Wert 0 haben
AssertionID	Die Assertion-ID wird in der Folge nicht verwendet und kann daher auf einen beliebigen Wert gesetzt werden.
Issuer	Der Vorname und Nachname des Benutzers durch ein Leerzeichen getrennt (der Vorname wird aus dem <GivenName>-Element, der Nachname aus dem <FamilyName>-Element der Personenbindung entnommen.
IssueInstant	Zeitpunkt der Erstellung der Assertion

Das Element enthält weiters genau ein <saml:AttributeStatement>-Element und genau ein <ds:Signature>-Element.

4.4.1 <saml:AttributeStatement>

Das <saml:AttributeStatement>-Element enthält genau ein <saml:Subject>-Element und ein <saml:Attribute>-Elemente.

4.4.2 <saml:Subject>

<saml:Subject> enthält ein <saml:NameIdentifier> als Inhalt.

4.4.3 <saml:NameIdentifier>

Enthält die URL der Authentisierungskomponente als Inhalt.

4.4.4 <saml:Attribute>

<saml:Attribute> kommt zweimal Mal vor um den Geschäftsbereich und die URL der nachfolgenden Applikation darzustellen.

Jedes <saml:Attribute> enthält genau ein <saml:AttributeValue>.

4.4.4.1 Geschäftsbereich

<saml:Attribute> für den Geschäftsbereich hat folgende Attribute:

Name	Beschreibung
AttributeName	Geschäftsbereich
AttributeNamespace	http://reference.e-government.gv.at/namespace/moa/20020822#

<saml:Attribute> für den Geschäftsbereich enthält genau ein <saml:AttributeValue> mit dem Geschäftsbereich als xs:token. Der Geschäftsbereich wurde vorher als Parameter dem Servlet übergeben.

4.4.4.2 URL der nachfolgenden Applikation

<saml:Attribute> hat folgende Attribute:

Name	Beschreibung
AttributeName	OA
AttributeNamespace	http://reference.e-government.gv.at/namespace/moa/20020822#

<saml:Attribute> für die URL der nachfolgenden Applikation enthält genau ein <saml:AttributeValue> mit der <oaurl> als xs:anyURI. Die URL wurde vorher als Parameter dem Servlet übergeben.

4.4.5 <ds:Signature>

Die XML Signatur der <saml:Assertion> wird gemäß Security-Layer Version 1.1 [SecLayer1.1] erstellt.

Die Signatur enthält ein <dsig:Reference> Element, das wie folgt auszuführen ist:

```
<dsig:Reference URI="">
  <dsig:Transforms>
    <dsig:Transform
      Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
      [XSLT Transformationen]
  </dsig:Transforms>
  <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
  <dsig:DigestValue> ..... </dsig:DigestValue>
</dsig:Reference>
```

Weiters enthält die Signatur ein <ds:KeyInfo> Element, welches ausreichend Information enthalten muss, dass eine automatische Validierung der Signatur möglich ist. Jedenfalls muss das X509 Signaturzertifikat eingebunden werden.

4.5 Anmeldedaten

Die Anmeldedaten werden in Form einer SAML-Assertion dargestellt, die, basierend auf dem signierten AUTH-Blocks und der vorher übersendeten zugehörigen Personenbindung, von der Authentisierungskomponente der nachfolgenden Applikation zur Verfügung gestellt wird. Die SAML-Assertion wird nicht signiert und enthält folgende Informationen:

- bPK
- signierter AUTH-Block (konfigurierbar)
- signierte Personenbindung
  (konfigurierbar: keine Personenbindung, Personenbindung mit Stammzahl, Personenbindung ohne Sammzahl)
- PersonData Struktur (konfigurierbar: mit oder ohne Stammzahl)
- Zertifikatstype (qualifiziertes Zertifikat oder einfaches Zertifikat)
- Information zur Behörde (optional)

Der Aufbau der SAML-Assertion für die Anmeldedaten erfolgt analog zum Aufbau des AUTH-Blocks (*Abschnitt 4.4*)

Das <saml:Assertion>-Element muss folgende verpflichtende Attribute enthalten:

Name	Beschreibung
MajorVersion	muss den Wert 1 haben
MinorVersion	muss den Wert 0 haben
AssertionID	Die Assertion-ID wird in der Folge nicht verwendet und kann daher auf einen eindeutigen Wert gesetzt werden.
Issuer	URL der Authentisierungskomponente
IssueInstant	Zeitpunkt der Erstellung der Assertion

Das Element enthält weiters genau ein <saml:AttributeStatement>-Element.

4.5.1 <saml:AttributeStatement>

Das <saml:AttributeStatement>-Element enthält genau ein <saml:Subject>-Element und mehrere <saml:Attribute>-Elemente.

4.5.2 <saml:Subject>

<saml:Subject> enthält ein <saml:NameIdentifier> Element.

4.5.3 <saml:NameIdentifier>

Das Element muss folgendes Attribut enthalten:

Name	Beschreibung
NameQualifier	Beschreibt die Domäne des Namens und muss den String urn:publicid:gv.at:cdid+ enthalten.

Als Inhalt enthält <saml:NameIdentifier> die bPK in Base64 kodierter Darstellung.

4.5.4 <saml:SubjectConfirmation>

Ist konfiguriert, dass AUTH-Block und/oder Personenbindung mitgegeben werden sollen, werden diese im Element <saml:SubjectConfirmation> eingebunden.

Als <saml:ConfirmationMethod> wird
http://reference.e-government.gv.at/names/moa/20020822#cm benutzt.

<saml:SubjectConfirmationData> enthält den AUTH-Block und/oder die Personenbindung (in dieser Reihenfolge). Die Stammzahl der Personenbindung wird je nach Konfiguration ausgeblendet oder mitgegeben.

4.5.5 <saml:Attribute>

<saml:Attribute> kommt mehrmals vor um die oben erwähnten Informationen darzustellen:

Jedes <saml:Attribute> enthält genau ein <saml:AttributeValue>.

4.5.5.1 PersonData

Das Attribut für die Personendaten ist immer vorhanden und enthält abhängig von der Konfiguration die Stammzahl (siehe *Abschnitt 4.6*).

<saml:Attribute> für PersonData hat folgende Attribute:

Name	Beschreibung
AttributeName	PersonData
AttributeNamespace	http://reference.e-government.gv.at/namespace/persondata/20020228#

<saml:AttributeValue> enthält genau die PersonData-Struktur (identisch zu der in der Personenbindung strukturiert). Diese PersonData-Struktur enthält: Vorname, Nachname, Geburtsdatum, Stammzahl (optional).

4.5.5.2 Zertifikatstyp

Dieses Attribut beschreibt den Typ des Zertifikats, mit dessen zugehörigen privaten Schlüssel der AUTH-Block vom Benutzer signiert wurde. <saml:Attribute> für den Zertifikatstyp hat folgende Attribute:

Name	Beschreibung
AttributeName	isQualifiedCertificate
AttributeNamespace	http://reference.e-government.gv.at/namespace/moa/20020822#

<saml:Attribute> für den Zertifikatstyp enthält genau ein <saml:AttributeValue> mit folgendem Inhalt:

true	wenn das Zertifikat ein qualifiziertes Zertifikat ist
false	wenn das Zertifikat ein einfaches Zertifikat ist

4.5.5.3 Behördenzertifikat

Dieses Attribut beschreibt ob bei der Signatur des AUTH-Block ein Behördenzertifikat verwendet wurde. Ein Behördenzertifikat ist ein Zertifikat mit der Erweiterung *Behördeneigenschaft* (siehe [OID]).

Ein <saml:Attribute> für die Information ob ein Behördenzertifikat vorliegt ist nur dann in den Anmeldedaten vorhanden, wenn zur Signaturerstellung ein Behördenzertifikat verwendet wurde.

Das <saml:Attribute> hat folgende Attribute:

Name	Beschreibung
AttributeName	isPublicAuthority
AttributeNamespace	urn:oid:1.2.40.0.10.1.1.1 (siehe [RFC3061])

Das <saml:Attribute> für die Information ob ein Behördenzertifikat vorliegt enthält genau ein <saml:AttributeValue> mit folgendem Inhalt:

True	wenn das Zertifikat ein Behördenzertifikat ist, aber kein

	Behördenkennzeichen im Zertifikat verfügbar ist
Behördenbezeichnung	wenn das Zertifikat ein Behördenzertifikat ist und ein Behördenkennzeichen im Zertifikat verfügbar ist

4.6 Konfiguration

Folgende Daten müssen für die Kommunikation mit allen zu unterstützenden Security-Layer Implementierungen konfiguriert werden können:

- Transformationen für unterschiedliche Datenformate für die sichere Anzeige des AUTH-Blocks.

Folgende Daten müssen für die Kommunikation mit MOA-Signaturprüfung konfigurierbar sein:

- Konfiguration ob die Kommunikation mit MOA-Signaturprüfung via SOAP- oder via API-Aufrufe erfolgen soll.
- Angabe, ob die Schnittstelle zur MOA-Signaturprüfung mit TLS-Client- und/oder TLS-Server-Authentisierung geschützt werden muss.
    - verwendetes TLS-Client-Zertifikat der Authentisierungskomponente
    - akzeptierte TLS-Server-Zertifikate der MOA-Signaturprüfung
    - vertrauenswürdige Wurzelzertifikate für akzeptierte TLS-Server-Zertifikate der MOA-Signaturprüfung
- `TrustProfileID` und `VerifyTransformsInfoID` für die Überprüfung der Signatur der Personenbindung
- `TrustProfileID` und `VerifyTransformsInfoID` für die Überprüfung der Signatur des AUTH-Blocks

Folgende Daten müssen für jede nachfolgende Applikation in der Authentisierungskomponente konfigurierbar sein:

- Angabe, ob die Stammzahl in der Personenbindung und in der PersonData-Struktur an die nachfolgende Applikation übergeben wird oder nicht
- Angabe, ob der AUTH-Block an die nachfolgende Applikation übergeben wird oder nicht.
- Angabe, ob die Personenbindung an die nachfolgende Applikation übergeben wird oder nicht.
- Angabe, ob die Schnittstelle zur nachfolgenden Applikation mit TLS-Client- und/oder TLS-Server-Authentisierung geschützt werden muss.
    - verwendetes TLS-Server-Zertifikat
    - akzeptierte TLS-Client-Zertifikate der nachfolgenden Applikation
    - vertrauenswürdige Wurzelzertifikate für akzeptierte TLS-Client-Zertifikate der nachfolgenden Applikation

Mehrere nachfolgende Applikationen müssen innerhalb derselben Authentisierungskomponente konfiguriert werden können. Die im Aufruf übergebene URL der nachfolgenden Applikation (siehe Abschnitt 4.1 *Schnittstelle zur verweisenden Webseite*) wird zur Auswahl der Konfiguration herangezogen. Ein eindeutiges Präfix der URL der nachfolgenden Applikation muss deshalb als Schlüssel verwendet werden. Die URL muss zumindest aus `https://` und einem gültigen Domänen-Namen bestehen.

4.7 Funktionsbeschreibung

4.7.1 Überprüfung der Signatur der Personenbindung

Die Überprüfung der Signatur der Personenbindung erfolgt gemäß der Spezifikation für die Personenbindung [PersBind]. Die Überprüfung wird mittels des Kommandos `<VerifyXMLSignatureRequest>` des MOA-Modul für Signatur-Prüfung mit folgenden verpflichtenden Eingangs-Parametern durchgeführt:

`<VerifySignatureInfo>`	enthält die signierte Personenbindung
`<SignatureManifestCheckParams>`	Das Attribut `ReturnReferenceInputData` muss auf `false` gesetzt sein
`<TrustProfileID>`	Eine Referenz auf ein Profil von Wurzelzertifikaten

Anschließend wird überprüft ob die Personenbindung mit einem für die Personenbindung zugelassenem Zertifikat signiert wurde.

4.7.2 Überprüfung des signierten AUTH-Blocks

Die Überprüfung der Signatur wird mittels des Kommandos `<VerifyXMLSignatureRequest>` des MOA-Modul für Signatur-Prüfung mit folgenden verpflichtenden Eingangs-Parametern durchgeführt:

`<VerifySignatureInfo>`	enthält den signierten AUTH-Block
`<SignatureManifestCheckParams>`	Das Attribut `ReturnReferenceInputData` muss auf `true` gesetzt sein.
`<ReturnHashInputData>`	muss enthalten sein
`<TrustProfileID>`	Eine Referenz auf ein Profil von Wurzelzertifikaten

Der AUTH-Block muss mit einem privaten Schlüssel des Benutzers signiert sein, dessen korrespondierender öffentliche Schlüssel Teil der Personenbindung ist.

5 Proxykomponente

Die Proxykomponente ermöglicht eine einfache Anbindung bestehender Online-Applikationen an den Authentisierungs-Mechanismus des MOA Identifikationsmoduls, indem es:

- die SAML-Struktur von der Authentisierungskomponente anfordert, um die Identität des Benutzers zu sichern
- eine einmalige Authentisierung an der Online-Applikation vornimmt („stateful" Online-Applikation) oder die Login-Parameter bei jedem Zugriff auf die Online-Applikation übergibt („stateless" Online-Applikation).
- die Online-Applikation durch transparentes Proxying vor nicht authentisierten Zugriffen schützt.

5.1 Schnittstelle zum Aufbau des Requests an die Online-Applikation

Zum Aufbau des Requests an die Online-Applikation muss innerhalb der Proxykomponente eine Klasse aufgerufen werden, die folgendes Interface implementiert. Die implementierende Klasse muss in der Proxykomponente global konfiguriert werden können. Die Proxykomponente muss eine Default-Implementierung bereitstellen, die die im Abschnitt 5.5 *Konfiguration* beschriebenen Konfigurationsmöglichkeiten abdeckt.

```
/**
 * Builds an HttpURLConnection to a java.net.URL which is derived
 * from an {@link HttpServletRequest} URL, by substitution of a
 * public URL prefix for the real URL prefix.<br>
 * The HttpURLConnection has been created by java.net.URL#openConnection, but
 * it has not yet been connected to by java.net.URLConnection#connect.<br>
 * The field settings of the HttpURLConnection are:
 * <ul>
 * <li><code>allowUserInteraction = false</code></li>
 * <li><code>doInput = true</code></li>
 * <li><code>doOutput = true</code></li>
 * <li><code>requestMethod = request.getMethod()</code></li>
 * <li><code>useCaches = false</code></li>
 * </ul>
 *
 * @param request
 *        the incoming request which shall be forwarded
 * @param publicURLPrefix
 *        the public URL prefix to be substituted by the real URL prefix
 * @param realURLPrefix
 *        the URL prefix to substitute the public URL prefix
 * @param sslSocketFactory
 *        factory to be used for creating an SSL socket in case
 *        of a URL for scheme <code>"https:"</code>;
 *        if <code>null</code>, the default SSL socket factory would be used
 * @return a URLConnection created by java.net.URL#openConnection,
 *         connecting to the requested URL with <code>publicURLPrefix</code>
 *         substituted by <code>realURLPrefix</code>
 * @throws IOException if an I/O exception occurs during opening the connection
 * @see java.net.URL#openConnection()
 * @see com.sun.net.ssl.HttpsURLConnection#getDefaultSSLSocketFactory()
 */
```

5.2 Schnittstelle zur Online-Applikation

Online-Applikationen werden unterschieden in stateless und stateful Online-Applikationen.

Stateless Online-Applikationen verlieren alle Benutzerinformation, wenn die Verbindung abgebaut wird (typischerweise am Ende eines HTTP-Requests). Es ist also notwendig, Benutzerinformatin (z.B. bPK) bei jedem Request mitzugeben.

Stateful Online-Applikationen behalten die Benutzerinformation auch wenn die Verbindung abgebaut wird. Üblicherweise werden für stateful Online-Applikationen Cookies eingesetzt. Benutzerinformation ist also nur beim Session-Aufbau notwendig.

Aus Sicht der Proxykomponente sind stateful Online-Applikationen eine Untermenge (ein Spezialfall) von stateless Online-Applikationen. Damit ist das Session-Handling der Online-Applikation (Timeout, etc.) für die Proxykomponente transparent.

5.2.1 Typisches Setup für stateless Online-Applikationen

Bei stateless Online-Applikationen muss jeder HTTP-Request und die dazugehörige Antwort der Online-Applikation durch die Proxykomponente gehen. Um performance-kritisches URL-Rewriting zu vermeiden, muss folgende Lösung verwendet werden:

- Der DNS-Eintrag der Online-Applikation verweist auf die Proxykomponente
- Die Proxykomponente kennt zu jedem DNS-Eintrag die wirkliche Adresse der Online-Applikation
- Die Online-Applikation ist mittels Virtual Hosting so konfiguriert, dass sie Requests, die auf den nach außen sichtbaren Domänen-Namen der Online-Applikation lauten, entgegennimmt

5.2.2 Typisches Setup für stateful Online-Applikationen

- Die Proxykomponente und die Online-Applikation haben unterschiedliche DNS-Namen, wobei einer der Namen eine Subdomain der anderen ist.
- Der erste Zugriff nach dem Authentisierungsvorgang erfolgt auf eine „Login-URL", deren DNS-Name auf die Proxykomponente auflöst. Die Proxykomponente leitet wie im Fall einer stateless Online-Applikation den Request an die Online-Applikation weiter (die Online-Applikation muss also auch per Virtual Hosting diese URL auflösen können).
- Das Login-Script der Online-Applikation prüft, ob der Request von der Proxykomponente kommt. Wenn ja, baut sie die Session auf (z.B. durch Setzen eines Cookies seitens der Online-Applikation im Browser des Benutzers, das für die Domäne der Online-Applikation gültig ist). Enstprechendes Firewall-Setup vorausgesetzt genügt ein einfaches Prüfen der IP-Adresse.
- Von diesem Zeitpunkt an gehen alle Requests an der Proxykomponente vorbei direkt an die Online-Appliktion, da der Benutzer an eine entsprechende Session gebunden ist.
- Um das Problem des Timeouts bzw. des Logout zu umgehen, besitzt die Proxykomponente ein sehr kurzes Timeout (Sekunden bzw. Minutenbereich). Das Session-Management obliegt ganz der Online-Appliktion.

5.3 Schnittstelle zum Browser des Benutzers

Die Proxykomponente verfügt über eine eigene HTTP-Session mit dem Benutzer, in der folgende Informationen server-seitig hinterlegt werden:

- Die Anmeldedaten des Benutzers

5.4 Proxy-Funktion

Die Proxykomponente muss bei stateless Online-Applikationen als transparenter Proxy zur Online-Applikation fungieren:

- Alle Requests zur Online-Applikation müssen von der Proxykomponente auf ihre Berechtigung geprüft werden: ein direkter Zugriff ohne vorherige Verwendung der Authentisierungskomponente darf nicht möglich sein.
- Alle Requests müssen mit allen Parametern zur Online-Applikation weitergeleitet werden. Ausnahme: die in der Konfiguration der Online-Applikation angegebenen Login-Parameter müssen gefiltert werden. Im Falle einer stateless Online-Applikation müssen die Login-Parameter bei jedem Request mitgegeben werden.
- Die von der Online-Applikation generierte Antwort muss unverändert an den Benutzer zurückgegeben werden.

5.5 Konfiguration

5.5.1 Konfiguration in der Proxykomponente

Für die Proxykomponente muss folgende Konfiguration durchgeführt werden können:

- Angabe des HTTP-Session-Timeout des Benutzers in der Proxykomponente. Nach einem Ablauf der Session ist der Benutzer nicht mehr an der Proxykomponente authentisiert. Alle server-seitig in der Session gespeicherten Benutzerdaten müssen gelöscht werden.
- Angabe der Klasse, die das `LoginParameterResolver` Interface implementiert (optional)

Folgende Daten müssen für die Kommunikation mit der Authentisierungskomponente konfigurierbar sein:

- Konfiguration ob die Kommunikation mit der Authentisierungskomponente via SOAP- oder via API-Aufrufe erfolgen soll.
- Angabe, ob die Schnittstelle zur Authentisierungskomponente mit TLS-Client und/oder TLS-Server-Authentisierung geschützt werden muss.
    - verwendetes TLS-Client-Zertifikat
    - akzeptierte TLS-Server-Zertifikate der Authentisierungskomponente
    - vertrauenswürdige Wurzelzertifikate für akzeptierte TLS-Server-Zertifikate der Authentisierungskomponente

In der Proxykomponente müssen je Online-Applikation folgende Konfigurationsmöglichkeiten gegeben sein:

- Eine optionale URL der Konfigurationsdatei für die Online-Applikation (siehe nächster Abschnitt). Falls die URL nicht angegeben ist, wird implizit folgende URL angenommen: `http://<oa-domain>/moaconfig.xml`, wobei `<oa-domain>` die nach außen sichtbare Domäne der Online-Applikation angibt.
- Das Mapping des nach außen sichtbaren Domänen-Namens (`<oa-domain>`) auf die wirkliche Domäne der Online-Applikation.
- Angabe, ob die Schnittstelle zur Online-Applikation mit TLS-Client und/oder TLS-Server-Authentisierung geschützt werden muss.
    - verwendetes TLS-Client-Zertifikat
    - akzeptierte TLS-Server-Zertifikate der Online-Applikation

- vertrauenswürdige Wurzelzertifikate für akzeptierte TLS-Server-Zertifikate der Online-Applikation

Ein eindeutiges Präfix der URL der Online-Applikation muss deshalb als Schlüssel verwendet werden. Die URL muss zumindest aus `https://` und einem gültigen Domänen-Namen bestehen.

5.5.2 Konfiguration der Online-Applikation

Die Konfigurationsdatei beschreibt die Art und Weise, wie die Proxykomponente die Anmeldung an der Online-Applikation durchführt. Die Lage der Konfigurationsdatei wird in der Konfiguration der Proxykomponente als URL hinterlegt. Die Konfigurationsdatei soll standardmäßig vom Web-Server der nachfolgenden Applikation geladen werden.

Die Konfigurationsdatei ist eine XML-Datei, deren Format in [MOA-ID Config Schema] beschrieben ist, und die folgende Element enthält:

Name	Beschreibung
`login-type`	Gibt an, ob die Online-Applikation ein einmaliges Login erwartet (`stateful`), oder ob die Login-Parameter bei jedem Request mitgegeben werden müssen (`stateless`). Im Fall einer stateful Online-Applikation werden die in der HTTP-Session der Proxykomponente gespeicherten Anmeldedaten nur für den Aufruf des Login-Scripts verwendet. Unmittelbar nach dem Aufruf müssen sie gelöscht werden.
`param-auth`	Konfiguriert die Übergabe der Authentisierungs-Parameter an die Online-Applikation mittels URL-Parametern.
`basic-auth`	Konfiguriert die Übergabe der Authentisierungs-Parameter an die Online-Appliktion mittels HTTP Basic Authentication.
`header-auth`	Konfiguriert die Übergabe der Authentisierungs-Parameter an die Online-Applikation in HTTP Request Headern.

5.5.2.1 Login mittels URL-Parameter

Das Element `<param-auth>` enthält eine Aufzählung von `<parameter>` Elementen, die folgende Attribute besitzen:

Name	Beschreibung
`Name`	Beschreibt den Namen des Parameters und ist ein frei zu wählender String.
`Value`	Beschreibt den Inhalt des Parameters und kann einen der durch MOAAuthDataType beschriebenen Werte annehmen.

Anhand der `<parameter>`-Elemente wird der Request für den Login-Vorgang (für stateful Online-Applikationen) folgendermaßen zusammengestellt:

`GET https://<login-url>?<p1.name=p1.resolvedValue>&<p2.name=p2.resolvedValue>...`

Die <login-url> ergibt sich aus dem Parameter OA des ursprünglichen Requests an die Authentisierungskomponente (siehe 4.1 *Schnittstelle zur verweisenden Webseite*). Der Wert `resolvedValue` wird in der Proxykomponente je nach Wert des Platzhalters eingesetzt.

5.5.2.2 Login mittels Basic Authentication

Das Element `<basic-auth>` enthält folgende 2 Elemente, die die UserId und das Passwort spezifizieren und für eine Basic Authentication analog [HTTP-Auth] angeben:

Name	Beschreibung
Userid	Die UserId des zu authentisierenden Benutzers; kann einen der durch MOAAuthDataType beschriebenen Werte annehmen.
Password	Das Passwort des zu authentisierenden Benutzers; kann einen der durch MOAAuthDataType beschriebenen Werte annehmen.

Aus diesen Angaben müssen je nach Wert der Platzhalter die entsprechenden HTTP-Header für die Basic Authentication erstellt und in den HTTP-Request an die Online-Applikation eingefügt werden.

5.5.2.3 Login mittels HTTP Request Header

Das Element <header-auth> enthält eine Aufzählung von <header> Elementen, die folgende Attribute besitzen:

Name	Beschreibung
Name	Beschreibt den Namen des Headers und ist ein frei zu wählender String.
Value	Beschreibt den Inhalt des Headers und kann einen der durch MOAAuthDataType beschriebenen Werte annehmen.

Die Header werden folgendermaßen in den Request an die Online-Applikation eingefügt:

<h1.name>:<h1.resolvedValue>

<h2.name>:<h2.resolvedValue>

...

Der Wert resolvedValue wird in der Proxykomponente je nach Wert des Platzhalters eingesetzt. Etwaige Header aus dem ursprünglichen Request an die Proxykomponente, die den selben Namen haben, müssen überschrieben werden.

5.5.2.4 MOAAuthDataType

Der Typ MOAAuthDataType dient an vielen Stellen der in diesem Abschnitt beschriebenen Konfiguration als Platzhalter für Anmeldedaten des Benutzers. Im Zuge des Anmeldevorgangs an der Online-Applikation müssen diese Platzhalter durch die entsprechenden Anmeldedaten des Benutzers ersetzt werden. Folgende Platzhalter sind definiert:

- MOAGivenName: der Vorname des Benutzers, wie in der PB enthalten
- MOAFamilyName: der Nachname des Benutzers, wie in der PB enthalten
- MOADateOfBirth: das Geburtsdatum des Benutzers, wie in der PB enthalten
- MOABPK: die BPK des Benutzers, wie von der Authentisierungskomponente
- MOAPublicAuthority: wird durch true ersetzt, falls es sich um einen Benutzer einer Behörde handelt, andernfalls wird false gesetzt
- MOABKZ: das Behördenkennzeichen (nur sinnvoll, wenn MOAPublicAuthority den Wert true ergibt)
- MOAQualifiedCertificate: wird durch true ersetzt, falls das Zertifikat des Benutzers qualifiziert ist, andernfalls wird false gesetzt
- MOAStammzahl: die Stammzahl des Benutzers; diese ist nur dann verfügbar, wenn die Online-Applikation die Stammzahl bekommen darf (und daher in der Personenbindung enthalten ist).
- MOAIPAddress: die IP-Adresse des Client des Benutzers

6 Definition der Schnittstelle zwischen Authentisierungskomponente und nachfolgenden Applikation

Über eine echte Client/Server-Schnittstelle verfügt nur die Authentisierungskomponente. Deren spezifizierte Funktionalität (die Abfrage der Identität) kann sowohl über SOAP (für Aufrufe über das Netzwerk) als auch über API Aufrufe verwendet werden.

6.1 SOAP

Im Dokument [SAML-Binding], Abschnitt 3, ist die Bindung von SAML-Anfragen an SOAP beschrieben. Dieses Binding muss zur Anfrage an die Authentisierungskomponente verwendet werden.

[MOA-ID WSDL] beschreibt die MOA-Schnittstelle bei Verwendung als Web-Service mit SOAP als Transportprotokoll. Aus dem WSDL-Dokument können mit geeigneten Tools Client-Stub- und Server-Tie-Objekte generiert werden.

Die SOAP Spezifikation basiert auf SOAP Version 1.1 [SOAP], die WSDL Spezifikation auf WSDL Version 1.1 [WSDL].

Das Dokument folgt dem hierarchischen Aufbau von WSDL-Dokumenten (siehe [WSDL] Abschnitt 1).

6.1.1 Beispiele für SOAP Messages

Beispiele für SOAP Requests und Response Messages zur Abfrage der SAML-Struktur sind ebenfalls in [SAML-Binding], Abschnitt 3, beschrieben.

6.1.1.1 Response im Fehlerfall:

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV :Server</faultcode>
      <faultstring>Server Error</faultstring>
      <detail>
        <moa:ErrorResponse
       xmlns:moa="http://reference.e-government.gv.at/namespace/moa/20020822#">
          <ErrorCode>2000</ErrorCode>
          <Info>Fehler im MOA Modul</Info>
        </moa:ErrorResponse>
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.2 API

Die in [MOA-ID WSDL] definierten Aufrufe sind in geeigneten Java Interfaces modelliert. Die Art der Modellierung orientiert sich an der im Package `javax.xml.soap` definierten Interfaces ohne diese jedoch direkt zu referenzieren.

Die Server-seitige Schnittstelle ist durch das Interface `AuthenticationService` definiert.

Die Definitionen der weiteren verwendeten Interfaces sind in [MOA-ID API] dokumentiert.

6.2.1 Interface IdentificationService

```
package at.gv.egovernment.moa;
/**
 * Interface providing functions for querying authentication data.
 */
public interface AuthenticationService {
/**
 * Gets authentication data.
 *
 * @param request A request for authentication data containing a SAML artifact.
 * @return User authentication data.
 * @throws MOAException Error in server side MOA module.
 */
public GetAuthenticationDataResponse
       getAuthenticationData(GetAuthenticationDataRequest request)
       throws MOAException;
}
```

7 Referenzen

Referenz	Titel	Verfasser	Datum
[MOA-Auftrag]	Projekt Module für Online Applikationen (MOA) –Design- und Spezifikationsphase – Auftragsbeschreibung	BMF	05.07.2002
[UC BRZ]	Anwendungsfälle Spezifikation, Version 4.1	BRZ	03.09.2002
[SecLayer1.1]	Schnittstellenspezifikation Security-Layer Version 1.1; http://www.buergerkarte.at/konzept/securitylayer/spezifikation/20020831/	CIO	31.08.2002
[SecLayer1.1-Bindung]	Bindung des Schnittstellenprotokolls des Security-Layers für das Konzept Bürgerkarte an Transportprotokolle, Version 1.1; http://www.buergerkarte.at/konzept/securitylayer/spezifikation/20020831/	CIO	31.08.2002
[PersBind]	XML Definition der Personenbindung, Version 1.1; http://www.buergerkarte.at/konzept/personenbindung/spezifikation/20020506	CIO	06.05.2002
[OID]	Object Identifier und Einsatz in X.509 Zertifikaten, Version 1.1	CIO	06.08.2002
[GB]	E-Government Geschäftsbereiche, Version 1.0; http://reference.e-government.gv.at/uploads/media/gb-1-0-20020524.pdf		24.05.2002
[GB-BK]	E-Government verwaltungsinterne Geschäftsbereiche - Spezifikation für den Datenschutz und die verwaltungsbereich-spezifisch abgeleitete Personenkennzeichnung, Version 1.0; http://reference.e-government.gv.at/uploads/media/gb-bk-1-0-20020524_02.pdf		24.05.2002
[MOA-ID Config Schema]	Datei „MOA-ID-Configuration-1.1.xsd “	ARGE	30.06.2003
[MOA-ID WSDL]	Datei „MOA ID 1.x.wsdl“	ARGE	30.06.2003
[MOA-ID API]	Java Package `at.gv.egovernment.moa`	ARGE	30.06.2003
[HTTP-Auth]	HTTP Authentication: Basic and Digest Access Authentication, IETF RFC 2069	IETF	06.1999
[SAML-Assertions]	Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML), Committee Specification 01; http://www.oasis-open.org/committees/security/docs/cs-sstc-core-01.pdf	OASIS	31.05.2002

[SAML-Binding]	Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML), Committee Specification 01; http://www.oasis-open.org/committees/ security/cs-sstc-bindings-01.pdf	OASIS	31.05.2002
[TLS]	The TLS Protocol V.1.0, IETF RFC 2046	IETF	01.1999
[RFC3061]	A URN Namespace of Object Identifiers; IETF RFC 3061; http://www.ietf.org/rfc/rfc3061.txt	IETF	02.2001
[SOAP]	Simple Object Access Protocol (SOAP) V.1.1	W3C	08.05.2000
[WSDL]	Web Services Description Language (WSDL) V.1.1	W3C	15.03.2001
[XMLDSig]	XML-Signature Syntax and Processing; http://www.w3.org/TR/xmldsig-core/	W3C	12.02.2002
[MOA SP-SS]	Spezifikation MOA SP-SS 1.1	BMF/CIO	30.06.2003