Transportprotokolle für die Applikationsschnittstelle Security-Layer der österreichischen Bürgerkarte		Konvention **1.2.2** Empfehlung
Kurzbeschreibung	Das Dokument Applikationsschnittstelle Security-Layer definiert die XML-Struktur sowie die notwendige Funktionalität der Schnittstellenbefehle des *Security-Layer*. Dieses Dokument beschreibt, wie diese XML-Befehle in eine Reihe von Transportprotokollen eingebunden werden können.	
Autoren:	Arno Hollosi Gregor Karlinger Thomas Rössler Martin Centner et al.	Projektteam/Arbeitsgruppe AG Bürgerkarte
Datum:	20.2.2008	

Inhaltsverzeichnis

1. 1 Allgemeines
- 1.1. Namenskonventionen
- 1.2. Schlüsselwörter

2. TCP/IP-Bindung
- 2.1. Bezeichner, Parameter und Default-Adresse
- 2.2. Ablauf
- 2.3. Kodierung

3. HTTP-Bindung
- 3.1. Bezeichner und Parameter
- 3.2. Ablauf
- 3.3. Kodierung

4. SSL/TLS-Bindung
- 4.1. Bezeichner, Parameter und Default-Adresse
- 4.2. Ablauf
- 4.3. Kodierung

5. HTTPS-Bindung
- 5.1. Bezeichner, Parameter und Default-Adresse

5.2. Ablauf

5.3. Kodierung

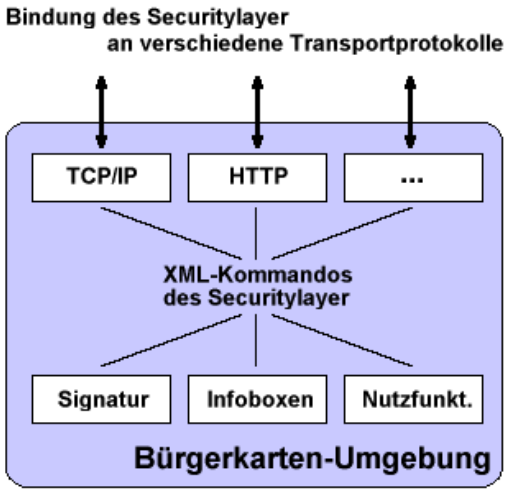
Glossar

Referenzen

A. Historie

1. 1 Allgemeines

Das Dokument Applikationsschnittstelle Security-Layer definiert die XML-Befehle, mit denen eine *Applikation* die Funktionalität der *Bürgerkarte* über die Schnittstelle *Security-Layer* nutzen kann. Dieses Dokument beschreibt, wie diese XML-Befehle in bestimmte Transportprotokolle eingebunden werden. Das folgende Blockbild zeigt die prinzipielle Struktur der *Bürgerkarten-Umgebung*. Die XML-Befehle werden über verschiedene Module an Transportprotokolle gebunden, im Bild z.B. an TCP/IP und HTTP. Die Module sorgen für die notwendige Umsetzung und Kodierung - die funktionalen Module der *Bürgerkarten-Umgebung* arbeiten nur mit den definierten XML-Kommandos und sind gegenüber dem Transport und den Eigenschaften des verwendeten Protokolls ignorant.



Die Adresse einer Bindung ist durch den verwendeten *Internet-Socket* bestimmt. Die *Socket*-Adresse besteht aus zwei Teilen: IP-Adresse und Portnummer. Für jede Bindung einer *lokalen Bürgerkarten-Umgebung* wird eine *Default*-IP-Adresse (bzw. DNS-Name) sowie eine *Default*-Portnummer spezifiziert.

Sofern Bindungen über die Art der übertragenen Daten unterschieden werden können, ist es möglich, dass Bindungen denselben Port benützen. Im Besonderen wird darauf hingewiesen, dass sich die TCP/IP-Bindung und die HTTP-Bindung eine Adresse teilen können MÜSSEN, ebenso wie die TLS- und HTTPS- Bindung (diese Anforderung ergibt sich aus den *Default*-Werten für die Ports). Die *Internet Assigned Numbers Authority* (*IANA*) hat für Bindungen des *Security-Layer* zwei Ports reserviert [port-numbers] (siehe *Default*-Werte in den Abschnitten Abschnitt 2.1, „Bezeichner, Parameter und Default-Adresse“, Abschnitt 3.1, „Bezeichner und Parameter“, Abschnitt 4.1, „Bezeichner, Parameter und Default-Adresse“ und Abschnitt 5.1, „Bezeichner, Parameter und Default-Adresse“).

1.1. Namenskonventionen

Zur besseren Lesbarkeit wurde in diesem Dokument auf geschlechtsneutrale Formulierungen verzichtet. Die verwendeten Formulierungen richten sich jedoch ausdrücklich an beide Geschlechter.

Folgende Namenraum-Präfixe werden in dieser Spezifikation zur Kennzeichnung der Namenräume von XML-Elementen verwendet:

Präfix	Namenraum	Erläuterung
sl	http://www.buergerkarte.at/namespaces/securitylayer/1.2#	Elemente der Applikationsschnittstelle Security-Layer

1.2. Schlüsselwörter

Dieses Dokument verwendet die Schlüsselwörter MUSS, DARF NICHT, ERFORDERLICH, SOLLTE, SOLLTE NICHT, EMPFOHLEN, DARF, und OPTIONAL zur Kategorisierung der Anforderungen. Diese Schlüsselwörter sind analog zu ihren englischsprachigen Entsprechungen MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, RECOMMENDED, MAY, und OPTIONAL zu handhaben, deren Interpretation in [KEYWORDS] festgelegt ist.

2. TCP/IP-Bindung

Die TCP/IP-Bindung ist eine 1:1 Umsetzung der *Security-Layer*-Befehle. Es wird eine übliche TCP/IP-Verbindung über einen *Socket* benutzt. Die XML-Befehle laut Applikationsschnittstelle Security-Layer werden ohne weitere Kodierung oder Umsetzung/Einbettung in der Verbindung übertragen.

2.1. Bezeichner, Parameter und Default-Adresse

Bezeichner und Parameter einer Bindung werden im Befehl *GetProperties* des *Security-Layer* verwendet. Diese Bindung trägt den Bezeichner `TCP/IP` und hat keine Parameter.

Die *Default*-IP-Adresse der TCP/IP-Bindung für *lokale Bürgerkarten-Umgebungen* ist `127.0.0.1`, der *Default*-Port `3495`.

2.2. Ablauf

1. Die *Bürgerkarten-Umgebung* öffnet das Service an der definierten Adresse und wartet auf eingehende Verbindungen.
2. Die *Applikation* öffnet eine Verbindung zur Service-Adresse.
3. Die *Applikation* überträgt den XML-*Request*.
4. Die *Bürgerkarten-Umgebung* arbeitet den XML-*Request* ab und schickt eine XML-*Response*.
5. Die *Bürgerkarten-Umgebung* schließt die Verbindung.

2.3. Kodierung

Es erfolgt keine weitere, einbettende Kodierung. Die XML-Befehle werden 1:1 in der TCP/IP-Verbindung übertragen.

3. HTTP-Bindung

Die HTTP-Bindung bietet der *Applikation* die Möglichkeit, über den Web-Browser des Bürgers direkt und ohne weitere aktive Komponenten (wie z.B. ein Applet) auf die Funktionen der *Bürgerkarte* zuzugreifen. Dazu ist eine entsprechende Einbettung des XML-*Requests* in einen HTTP-*Request* erforderlich.

3.1. Bezeichner und Parameter

Bezeichner und Parameter der Bindung werden im Befehl *GetProperties* des *Security-Layer* verwendet. Diese Bindung trägt den Bezeichner `HTTP` und hat keine Parameter.

Die *Default*-IP-Adresse der HTTP-Bindung für *lokale Bürgerkarten-Umgebungen* ist `127.0.0.1`, der *Default*-Port `3495`.

3.2. Ablauf

3.2.1. Allgemeines

3.2.1.1. Terminologie

Die *Applikation* übermittelt der *Bürgerkarten-Umgebung* bei der HTTP-Bindung ein HTML-Formular (vergleiche [HTML4], Abschnitt 17.13), das eine Reihe von Formularelementen enthält. In den weiteren Ausführungen werden die Formularelemente in folgende Kategorien eingeteilt:

Formular-Parameter

Formular-Parameter sind jene Formularelemente, die zur Steuerung des Ablaufs in der Bürgerkartenumgebung angegeben werden. Es handelt sich dabei um die Elemente mit den Namen `XMLRequest`, `RedirectURL`, `DataURL` und `StylesheetURL`. Für detaillierte Informationen siehe Abschnitt 3.2.2, „Formular-Parameter“.

Weitergabe-Parameter

Weitergabe-Parameter sind jene Formularelemente, die bei der Verwendung des Formular-Parameters `DataURL` von der *Bürgerkarten-Umgebung* an den mittels `DataURL` bezeichneten Server als Formularelement weitergegeben werden. Ein Formularelement wird zum Weitergabe-Parameter, wenn sein Name mit dem Zeichen _ (Unterstrich) endet. Siehe auch Abschnitt 3.3.2.2, „HTTP-Request an DataURL“.

Weitergabe-Header

Weitergabe-Header sind jene Formularelemente, die bei der Verwendung des Formular-Parameters `DataURL` von der *Bürgerkarten-Umgebung* an den mittels `DataURL` bezeichneten Server als HTTP-Header weitergegeben werden. Ein Formularelement wird zum Weitergabe-Header, wenn sein Name mit der Zeichenfolge __ (doppelter Unterstrich) endet. Der Name des Weitergabe-Headers ist beliebig; der Wert des Weitergabe-Headers besteht dabei aus dem HTTP-Header-Namen, einem Doppelpunkt sowie dem HTTP-Header-Wert (also genau wie eine Header-Zeile im HTTP-Request, z.B. `Header: Value`. Siehe auch Abschnitt 3.3.2.2, „HTTP-Request an DataURL“.

Sonstige Formular-Felder

Sonstige Formularfelder, sind all jene Formularelemente, die nicht unter eine der oben genannten Kategorien fallen. Sie werden von der *Bürgerkarten-Umgebung* grundsätzlich nicht ausgewertet, können aber von einem Security-Layer-Befehl aus referenziert werden (vergleiche Abschnitt 3.2.1.2, „Referenzieren von Formularfeldern“).

3.2.1.2. Referenzieren von Formularfeldern

Im Kontext der HTTP-Bindung ist es möglich Formularelemente, die im HTTP-Request übergeben werden, vom einem *Security-Layer* Befehl aus zu referenzieren. Dabei können nicht nur die oben definierten Formular-Parameter referenziert werden, sondern beliebige Formularelemente (Formular-Parameter, Weitergabe-Parameter, Weitergabe-Header und sonstige Formular-Felder). Mögliche Anwendungsfälle sind das Referenzieren bei den Signaturbefehlen *Create(XML|CMS)Signature* und *Verify(XML|CMS)Signature*.

Das in einer solchen URL zu verwendende Protokoll lautet `formdata`. Als opaker Teil der URL ist der Name des zu referenzierenden Formularelements anzugeben. Als Auflösung der URL muss die *Bürgerkarten-Umgebung* den Wert des referenzierten Formularelements liefern.

Lautet die HTML-Schreibweise des Formularelements beispielsweise `<input name="summary" value="Eine kurze Zusammenfassung"/>`, so wird auf dieses Element mit der URL `formdata:summary` referenziert. Das Ergebnis der Aulösung ist der String `Eine kurze Zusammenfassung`.

3.2.2. Formular-Parameter

Folgende Formular-Parameter zur Steuerung des Ablaufs in der *Bürgerkarten-Umgebung* stehen zur Verfügung:

`XMLRequest`

MUSS angegeben werden und enthält den *Security-Layer* Befehls-Request als XML-Struktur.

`RedirectURL`

DARF angegeben werden und dient ggf. zur asynchronen Umleitung der *Applikation*. Siehe Abschnitt 3.2.3, „Schrittweiser Ablauf“.

`DataURL`

DARF angegeben werden und dient ggf. dazu, das Ergebnis der Befehlsbearbeitung durch die *Bürgerkarten-Umgebung* an einen bestimmten Server, anstatt es im HTTP-Response an die *Applikation* senden zu lassen. Siehe Abschnitt 3.2.3, „Schrittweiser Ablauf“.

`StylesheetURL`

DARF angegeben werden, und wird ggf. von der *Bürgerkarten-Umgebung* verwendet, um das Ergebnis der Befehlsbearbeitung vor der Retournierung in der HTTP-Response an die *Applikation* zu transformieren. Siehe Abschnitt 3.2.3, „Schrittweiser Ablauf".

PushInfobox

DARF angegeben werden, und dient ggf. dazu der *Bürgerkarten-Umgebung* zu signalisieren, dass die Inhalte der mit diesem Formular-Parameter bezeichneten Infoboxen von dem mit dem Formular-Parameter `DataURL` bestimmten Server zusammen mit dem Ergebnis der Befehlsausführung entgegengenommen und verarbeitet werden können. Ein oder mehere Bezeichner von Infoboxen können, seperiert durch das Zeichen Komma (`,`) angegeben werden.

Anmerkung

Der Formular-Parameter `PushInfobox` darf nicht mit einer Anfrage zum Lesen einer Infobox (siehe Abschnitt 7.5, „Lesen von Daten einer Infobox" in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer) gleichgesetzt werden. Die Übergabe der Inhalte der mit diesem Formular-Parameter bezeichneten Infoboxen muss vom Benutzer gewünscht und entsprechend voreingestellt werden.

Abschnitt 3.2.4, „Kombination der Formular-Parameter" listet sinnvolle Kombinationen der Formular-Parameter `RedirectURL`, `DataURL`, `StylesheetURL` und `PushInfobox`.

3.2.3. Schrittweiser Ablauf

1. Die *Bürgerkarten-Umgebung* öffnet das Service am angegebenen Port und wartet auf eingehende Verbindungen.
2. Eine *Applikation* setzt einen HTTP-Request zur angegebenen Adresse ab. Die so erzeugte Verbindung wird im folgenden als Browser-Verbindung bezeichnet.
3. Ist der Formular-Parameter `RedirectURL` angegeben worden, schickt die *Bürgerkarten-Umgebung* als Antwort einen HTTP-Redirect (HTTP Code: 302 oder 303) mit der in diesem Parameter angegebenen URL an die *Applikation* und schließt die Browser-Verbindung.
4. Die *Bürgerkarten-Umgebung* entnimmt den zu bearbeitenden Befehls-Request aus dem Formular-Parameter `XMLRequest` und bearbeitet ihn.
5. Ist der Formular-Parameter `DataURL` angegeben worden, schickt die *Bürgerkarten-Umgebung* die Befehls-Response ggf. zusammen mit den Inhalten der mit dem Formular-Parameter `PushInfobox` bezeichneten Infoboxen entsprechend Abschnitt 3.3.2.2, „HTTP-Request an DataURL" kodiert an den mittels `DataURL` bezeichneten Server.

   Die Antwort des Servers beeinflusst den Ablauf wie folgt:

   a. HTTP-Code 200, Content-Type: `text/xml` oder `text/plain` oder `text/html` und Inhalt der HTTP-Response gleich `<ok/>` (in exakt dieser Schreibweise): die Verarbeitung setzt in Schritt 6 fort.

   b. HTTP-Code 200, Content-Type: `text/xml` und Inhalt der HTTP-Response ungleich `<ok/>`: Die Daten werden als Befehls-Request ausgewertet und dem Formular-Parameter `XMLRequest` zugewiesen. Der Formular-Parameter `PushInfobox` wird verworfen, die anderen Formular-Parameter (`StylesheetURL`, `RedirectURL` und `DataURL`), Weitergabe-Parameter und -Header sowie die sonstigen (referenzierbaren) Formular-Felder bleiben unverändert. Die Verarbeitung setzt in Schritt 4 fort.

   c. HTTP-Code 307, Content-Type: `text/xml`: Der Inhalt der HTTP-Response wird als Befehls-Request ausgewertet und dem Formular-Parameter `XMLRequest` zugewiesen. Der Inhalt des HTTP-Headers `Location` wird dem Formular-Parameter `DataURL` zugewiesen. Der Formular-Parameter PushInfobox wird verworfen, die anderen Formular-Parameter (`StylesheetURL` und `RedirectURL`), Weitergabe-Parameter und -Header sowie die sonstigen (referenzierbaren) Formular-Felder bleiben unverändert. Die Verarbeitung setzt in Schritt 4 fort.

   d. HTTP-Code 200, Content-Type: `application/x-www-form-urlencoded` oder `multipart/form-data`: Der Inhalt der HTTP-Response wird als Sammlung von Formularelementen ausgewertet. Die bisherigen Formular-, Weitergabe-Parameter und -Header sowie die sonstigen (referenzierbaren) Formular-Felder werden verworfen. Stattdessen werden die in der HTTP-Response enthaltenen Formular-Parameter, Weitergabe-Parameter und -Header sowie die dort ggf. ebenfalls enthaltenen sonstigen (referenzierbaren) Formular-Felder verwendet. Die Verarbeitung setzt in Schritt 3 fort.

e. HTTP-Code 200, wobei die Kombination aus Content-Type und Inhalt der HTTP-Response nicht unter einen der Punkte 5.a, 5.b oder 5.d fällt; HTTP-Code 307, wobei Content-Type nicht unter Punkt 5.c fällt; HTTP-Code 301, 302, 303: Die HTTP-Response wird unverändert an die Browser-Verbindung weitergeleitet, die Browser-Verbindung wird geschlossen und die Verarbeitung beendet.

f. Andere Fälle (z.B. HTTP-Code 404): Die *Bürgerkarten-Umgebung* gibt über die Benutzerschnittstelle eine entsprechende Fehlermeldung aus, übermittelt diese auch an eine eventuell noch bestehende Browser-Verbindung, beendet diese und bricht die Verarbeitung ab.

6. Ist der Formular-Parameter `StylesheetURL` angegeben, liest die *Bürgerkarten-Umgebung* einen XSL-Stylesheet von der darin angegebenen Adresse und transformiert damit die Befehls-Response.
7. Ist der Formular-Parameter `RedirectURL` nicht angegeben, schickt die *Bürgerkarten-Umgebung* die (eventuell transformierte) Befehls-Response in der Browser-Verbindung und schließt die Browser-Verbindung

Eine *serverbasierte Bürgerkarten-Umgebung* DARF in der Browserverbindung zwischen den Schritten 2 und 3 (bei Verwendung eines *Redirects*) bzw. zwischen den Schritten 2 und 7 (sonst) auch die Kommunikation der Benutzerschnittstelle abwickeln (und dabei z.B. auch Cookies setzen). Die *Applikation* DARF also NICHT davon ausgehen, dass als Antwort auf den gesendeten HTTP-Request unmittelbar der *Redirect* bzw. das Befehlsergebnis retourniert wird.

Anmerkung

Ist eine `RedirectURL` angegeben, wird im Schritt 3 die Browser-Verbindung beendet. Ein ggf. zu einem späteren Zeitpunkt notwendiges Senden von Daten an die Broswerverbindung (z.B. im Fall 5.e) ist dann nicht mehr möglich. In diesem Fall MUSS die *Bürgerkarten-Umgebung* eine entsprechende Fehlermeldung über die Benutzerschnittstelle ausgeben.

Anmerkung

Wenn im obigen Ablauf als Bedingung formuliert ist, dass der HTTP-Header Content-Type beispielsweise gleich `text/plain` sein muss, bedeutet dass lediglich, dass der Media-Type grundsätzlich `text/plain` sein muss. Lautet der tatsächliche Wert von Content-Type etwa `text/plain; charset=ISO-8859-1`, ist die Bedingung ebenfalls erfüllt.

Anmerkung

Aus den Erläuterungen zu den Fällen 5.b, 5.c und 5.d ergibt sich, dass sowohl die veränderten als auch die unverändert gebliebenen Parameter im nächsten HTTP-Response erneut an den mittels DataURL bezeichneten Server gesendet werden müssen.

Anmerkung

Eine *Applikation*, welche den Formular-Parameters `DataURL` verwendet, sollte im Allgemeinen davon ausgehen, dass eine *Bürgerkarten-Umgebung* kein Cookie-Management implementiert hat und daher auf den Cookie-Mechanismus von HTTP verzichten.

3.2.4. Kombination der Formular-Parameter

Aus dem in Abschnitt 3.2.3, „Schrittweiser Ablauf“ beschriebenen Ablauf ergeben sich sinnvolle und nicht sinnvolle Kombinationen der Formular-Paramter `RedirectURL`, `DataURL`, `StylesheetURL` und `PushInfobox`. Der Formular-Parameter `PushInfobox` ist nur in Kombination mit dem Formular-Parameter `DataURL` sinnvoll. Die folgende Tabelle gibt eine Übersicht und erläutert kurz die Auswirkungen:

RedirectURL	DataURL	StylesheetURL	Sinn	Auswirkung
-	-	-	beschränkt sinnvoll	Befehls-Response wird direkt an die aufrufende *Applikation* geschickt. Nur für Auswertung durch Programme oder Scripts geeignet. Für Endanwender nicht geeignet, da sie mit XML als Klartext konfrontiert werden.
-	-	X	sinnvoll	Befehls-Response wird nach HTML transformiert und an aufrufende *Applikation* geschickt.
-	X	-	sinnvoll	Sinnvoll. Befehls-Response wird an den mittels `DataURL` bezeichneten Server geschickt. Der Server

				ist für eine sinnvolle abschließende Antwort an die *Applikation* (Fälle 5a oer 5e) verantwortlich.
-	X	X	sinnvoll	Das Resultat wird an den mittels `DataURL` bezeichneten Server geschickt. Die aufrufende *Applikation* erhält ein nach HTML transformiertes Resultat: dies entspricht einer synchronen Benutzerführung. Nur sinnvoll in den Fällen 5a, 5b, 5c. In den Fällen 5d, 5e, 5f ist die Angabe der `StylesheetURL` nutzlos.
X	-	- / X	nicht sinnvoll	Die *Applikation* wird umgeleitet, aber die Befehls-Response kann von niemandem in Empfang genommen werden.
X	X	-	sinnvoll	Die *Applikation* erhält sofort einen Redirect, die Befehls-Response wird an den mittels `DataURL` bezeichneten Server geschickt: dies entspricht einer asynchronen Benutzerführung.
X	X	X	nicht sinnvoll	`StylesheetURL` ist nutzlos. Stattdessen sollte der vorhergehende Fall verwendet werden.

3.3. Kodierung

3.3.1. HTTP-Request

Der HTTP-Request der *Applikation* an die *Bürgerkarten-Umgebung* muss entsprechend [HTTP1.1] kodiert sein. Die Methode MUSS entweder `GET` oder `POST` sein, es MÜSSEN beide Varianten von der *Bürgerkarten-Umgebung* unterstützt werden. (Anmerkung: manche Webclients unterstützen nur eine limitierte Größe der `GET` Requests.) Die Kodierung kann als `application/x-www-form-urlencoded` oder als `multipart/form-data` erfolgen, beide Formate MÜSSEN von der *Bürgerkarten-Umgebung* unterstützt werden.

Wird der HTTP-Request an eine *lokale Bürgerkarten-Umgebung* gesendet, MUSS er an die URL `/http-security-layer-request` erfolgen. Andernfalls muss die *Bürgerkarten-Umgebung* mit einer HTTP 404 Fehlermeldung antworten. Wird der HTTP-Request hingegen an eine *serverbasierte Bürgerkarten-Umgebung* gesendet, darf der Diensteanbieter eine beliebige URL vorgeben.

3.3.2. HTTP-Response bzw. HTTP-Request an DataURL

3.3.2.1. HTTP-Response an die Browser-Verbindung

Identifkation der Bürgerkarten-Umgebung

Sendet die *Bürgerkarten-Umgebung* die Befehls-Response in der Browser-Verbindung (vgl. Abschnitt 3.2.3, „Schrittweiser Ablauf“, Ziffer 7), MUSS sich die *Bürgerkarten-Umgebung* unter Verwendung des HTTP-Headers `Server` (vgl. [HTTP1.1], Abschnitt 14.38) identifizieren. Der HTTP-Header `Server` MUSS dabei folgendem Muster entsprechen:

```
Server = "Server" ":" "citizen-card-environment" "/" cceVersion " " productName "/" productVersion
```

Es sind also zwei Produkt-Token anzugeben: Der erste drückt aus, dass es sich beim beim Server um eine *Bürgerkarten-Umgebung* handelt; `cceVersion` gibt die neueste Version der Security-Layer-Spezifikation an, welche die *Bürgerkarten-Umgebung* unterstützt (also etwa `1.2` für diese Version). Der zweite Produkt-Token bezeichnet den Hersteller der *Bürgerkarten-Umgebung*; `productName` und `productVersion` sind innerhalb der Grenzen von [HTTP1.1] frei wählbar.

Die *Bürgerkarten-Umgebung* MUSS sich nach der selben Methode auch dann identifizieren, wenn sie in der Browser-Verbindung einen HTTP-Redirect sendet (vgl. Abschnitt 3.2.3, „Schrittweiser Ablauf“, Ziffer 3).

Kodierung der HTTP-Response

Die Befehls-Response wird als Nutzinhalt der HTTP-Response an den Browser übermittelt (vgl. Abschnitt 3.2.3, „Schrittweiser Ablauf“, Fall 5.a). Jedenfalls muss die *Bürgerkarten-Umgebung* den HTTP-Header `Content-Type` setzen:

- Liegt die Befehls-Response als XML-Struktur vor, muss entweder der Wert `text/xml` verwendet

werden (wenn die Befehlsresponse laut Abschnitt 3.2.3, „Schrittweiser Ablauf“, Schritt 6 nicht mit Hilfe eines *Stylesheets* transformiert wurde), oder aber *jener MIME Media Type*, der sich aus der angewandten Stylesheet-Transfomation ergibt.

- Liegt die Befehls-Response nicht als XML-Struktur vor (z.B. Binär-Response auf die Befehle *DecryptCMS* oder *DecryptXML*), muss der Content-Type entsprechend den binär übermittelten Daten gesetzt werden. Ist der Content-Type der binär übermittelten Daten nicht bekannt, muss der Wert `application/octet-stream` verwendet werden.

3.3.2.2. HTTP-Request an DataURL

Identifkation der Bürgerkarten-Umgebung

Sendet die *Bürgerkarten-Umgebung* eine Befehls-Response per HTTP-Request an den mittels `DataURL` bezeichneten Server (vgl. Abschnitt Abschnitt 3.2.2, „Formular-Parameter“, Ziffer 5), MUSS sich die *Bürgerkarten-Umgebung* unter Verwendung des HTTP-Headers `User-Agent` (vgl. [HTTP1.1], Abschnitt 14.43) identifizieren. Der HTTP-Header `User-Agent` MUSS dabei folgendem Muster entsprechen, wobei die im Muster verwendeten Variablen wie in Abschnitt Abschnitt 3.3.2.1, „HTTP-Response an die Browser-Verbindung“ zu interpretieren sind:

```
User-Agent = "User-Agent" ":" "citizen-card-environment" "/" cceVersion " " productName "/" productVersion
```

Kodierung des HTTP-Requests

Der HTTP-Request an den mittels `DataURL` bezeichneten Server MUSS nach der Methode `POST` (vgl. [HTTP1.1], Abschnitt 9.5) erfolgen, als `multipart/form-data` kodiert sein, und folgende Formularfelder enthalten:

Formular-Parameter `ResponseType`

Wird immer auf den Wert `HTTP-Security-Layer-RESPONSE` gesetzt (kann in späteren Versionen zur Unterscheidung herangezogen werden).

Formular-Parameter `XMLResponse`

Enthält die Befehls-Response, falls sie als XML-Struktur vorliegt. Der Header `Content-Type` MUSS für diesen *Mime Part* verwendet werden und ist fix mit dem Wert `text/xml` zu belegen (siehe auch 2. Anmerkung in Abschnitt 3.2.3).

Formular-Parameter `BinaryResponse`

Enthält die Befehls-Response, falls sie nicht als XML-Struktur vorliegt (z.B. Binär-Response auf die Befehle `sl12:DecryptCMSRequest` oder `sl12:DecryptXMLRequest`). Der Header `Content-Type` MUSS für diesen *Mime Part* verwendet werden, um den Content-Type der Binär-Response zu bezeichnen. Ist der Content-Type nicht bekannt, so MUSS der Feldwert mit `application/octet-stream` angegeben werden (siehe auch 2. Anmerkung in Abschnitt 3.2.3).

Weitergabe-Parameter

Die gegebenenfalls im HTTP-Request an die *Bürgerkarten-Umgebung* enthaltenen Weitergabe-Parameter (vgl. Abschnitt 3.2.1.1, „Terminologie“) werden in den HTTP-Request an den durch die `DataURL` bezeichneten Server unverändert übernommen.

Weitergabe-Header

Die gegebenenfalls im HTTP-Request an die *Bürgerkarten-Umgebung* enthaltenen Weitergabe-Header (vgl. Abschnitt 3.2.1.1, „Terminologie“) werden in den HTTP-Request an den durch die `DataURL` bezeichneten Server als HTTP-Header aufgenommen.

Wenn die Befehls-Response *keine* Fehler-Antwort (siehe Abschnitt 10, „Fehlerbehandlung“ in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer) darstellt, DARF der HTTP-Request an den mittels `DataURL` bezeichneten Server Forumlar-Parameter für Infoboxen enthalten, deren Bezeichner

- im Formular-Parameter `PushInfbox` im HTTP-Request an die *Bürgerkarten-Umgebung* (siehe Abschnitt 3.2.2, „Formular-Parameter“) angeführt sind und
- sich von den Namen der oben angeführten Formular-Parameter (`ResponseType`, `XMLResponse`, `BinaryResponse` und Weitergabe-Parameter) unterscheiden.

Der Name eines solchen Formular-Parameters MUSS dem Bezeichner der entsprechenden Infobox entsprechen; der Wert des Formular-Parameters MUSS der Antwort einer Anfrage zum Lesen der entsprechend bezeichneten Infobox (siehe Abschnitt 7.5, „Lesen von Daten einer Infobox“ in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer) entsprechen. Beim Zugriff auf die Infobox MÜSSEN die selben Regeln für den Zugriffschutz angewendet werden, wie bei einer Anfrage zum Lesen der entsprechenden Infobox nach Abschnitt 7.5, „Lesen von Daten einer Infobox“ in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer. Wird der Zugriff auf eine Infobox entsprechend Zugriffschutz verweigert, DARF der HTTP-Request an den mittels `DataURL` bezeichneten Server den Formular-Parameter für die entsprechende Infobox NICHT enthalten.

Ist die Befehls-Response die Antwort auf eine Anfrage zum Lesen einer Infobox, dann MÜSSEN ggf. in der Anfrage angegebene boxspezifische Parameter sofern anwendbar auch auf die als Formular-Parameter übergebenen Infoboxen angwendet werden.

> **Anmerkung**
>
> Erfolgt die Übergabe von elektronischen Vollmachten (Infobox Mandates) beispielsweise mit der Befehls-Response auf die Anfrage zum Lesen der Personenbindung (Infobox IdentityLink), dann muss eine in der Anfrage eventuell vorhandener boxspezifischer Parameter (z.B: sl:IdentityLinkDomainIdentifyer auch auf die übermittelten elektronischen Vollmachten angewandt werden.

3.3.3. Protokolle für DataURL und StylesheetURL

Für Zugriffe auf die `DataURL` und die `StylesheetURL` MÜSSEN HTTP und HTTPS als Protokolle unterstützt werden.

4. SSL/TLS-Bindung

Die TLS-Bindung ist eine 1:1 Umsetzung der *Security-Layer*-Befehle. Es wird eine übliche TLS-Verbindung [TLS] über einen *Socket* benutzt. Die XML-Befehle laut Applikationsschnittstelle Security-Layer werden ohne weitere Kodierung oder Umsetzung/Einbettung in der Verbindung übertragen.

Die TLS-Bindung entspricht von Ablauf und Aufbau her der TCP/IP-Bindung, mit dem Unterschied, dass das zugrundeliegende Transportprotokoll TLS ist.

4.1. Bezeichner, Parameter und Default-Adresse

Bezeichner und Parameter einer Bindung werden im Befehl *GetProperties* des *Security-Layer* verwendet. Diese Bindung trägt den Bezeichner `TLS` und hat keine Parameter.

Die *Default*-IP-Adresse der TLS-Bindung für *lokale Bürgerkarten-Umgebungen* ist `127.0.0.1`, der *Default*-Port `3496`.

4.2. Ablauf

Der Ablauf ist analog zu jenem der TCP/IP-Bindung (vergleiche Abschnitt 2.2, „Ablauf“), mit dem Unterschied, dass im Punkt 2 beim Aufbau der Verbindung zusätzlich entsprechend TLS ein *Handshake* und eine Verschlüsselung des Kanales stattfindet.

4.3. Kodierung

Es erfolgt keine weitere, einbettende Kodierung. Die XML-Befehle werden 1:1 in der TLS-Verbindung übertragen.

5. HTTPS-Bindung

Die HTTPS-Bindung bietet wie die HTTP-Bindung der *Applikation*die Möglichkeit direkt und ohne weitere aktive Komponenten (wie z.B. ein Applet) auf die Funktionen der *Bürgerkarte* zuzugreifen. Dazu ist eine entsprechende Einbettung des XML-*Requests* in einen HTTP-*Request* erforderlich.

Die HTTPS-Bindung entspricht der HTTP-Bindung mit dem Unterschied, dass das zugrundeliegende Transportprotokoll HTTPS [HTTPS] (HTTP mit TLS [TLS]) ist.

5.1. Bezeichner, Parameter und Default-Adresse

Bezeichner und Parameter einer Bindung werden im Befehl *GetProperties* des *Security-Layer* verwendet. Diese Bindung trägt den Bezeichner `HTTPS` und hat keine Parameter.

Die *Default*-IP-Adresse der HTTPS-Bindung für *lokale Bürgerkarten-Umgebungen* ist `127.0.0.1`, der *Default*-Port `3496`.

5.2. Ablauf

Der Ablauf ist analog zu jenem der HTTP-Bindung, mit dem Unterschied, dass im Punkt 2 beim Aufbau der Verbindung zusätzlich entsprechend TLS ein Handshake und eine Verschlüsselung des Kanals stattfindet.

5.3. Kodierung

Die Kodierung erfolgt wie in Abschnitt 3.3, „Kodierung“ für die HTTP-Bindung festgelegt, mit folgenden Ausnahmen:

- Wird der HTTPS-Request an eine *lokale Bürgerkarten-Umgebung* gesendet, MUSS er an die URL `/https-security-layer-request` erfolgen.
- Wird die XML-Antwort per *HTTP(S)*-Request an einen *DataURL-Server* gesendet, so MUSS der Formularparameter `ResponseType` auf den Wert `HTTPS-Security-Layer-RESPONSE` gesetzt werden.

Glossar

Glossar

Applikation

Jenes Programm, das Anfragen an die *Bürgerkarten-Umgebung* über den *Security-Layer* richtet und die entsprechenden Antworten entgegennimmt und auswertet.

Benutzer-Schnittstelle

Jene Schnittstelle, über die der *Bürger* mit der *Bürgerkarten-Umgebung* kommuniziert. Über diese Schnittstelle wird einerseits die Benutzerinteraktion abgewickelt, die gegebenenfalls zur Abwicklung eines Befehls des *Security-Layers* notwendig ist (z.B. die Anzeige eines zu signierenden Dokuments beim Befehl zur Erzeugung einer XML-Signatur); andererseits kann der *Bürger* über diese Schnittstelle seine *Bürgerkarten-Umgebung* nach seinen persönlichen Bedürfnissen konfigurieren (z.B. kann er Einstellungen zum Zugriffsschutz auf seine Infoboxen verändern). Die Vorgaben an die *Benutzer-Schnittstelle* sind in Minimale Umsetzung des Security-Layers geregelt.

Bürger

Jene Person, die die Funktionen der *Bürgerkarten-Umgebung* für die sichere Abwicklung von E-Government oder E-Commerce verwenden möchte. Die Ansteuerung der *Bürgerkarten-Umgebung* erfolgt in der Regel nicht durch den *Bürger* selbst, sondern durch die *Applikation*, welche die E-Government oder E-Commerce Anwendung repräsentiert.

Bürgerkarte

Laut [E-GovG], §10 ZI 10 ist die *Bürgerkarte* „die unabhängig von der Umsetzung auf unterschiedlichen technischen Komponenten gebildete logische Einheit, die eine elektronische Signatur mit einer Personenbindung (§ 4 Abs. 2) und den zugehörigen Sicherheitsdaten und -funktionen sowie mit allenfalls vorhandenen Vollmachtsdaten verbindet“. Im Sinne der in den Spezifikationen zur österreichischen Bürgerkarte gebrauchten Terminologie ist die *Bürgerkarten-Umgebung* die Implementierung der logischen Einheit *Bürgerkarte*.

Bürgerkarten-Umgebung

Jenes Programm bzw. jener Dienst, der die Funktionalität der *Bürgerkarte* zur Verfügung stellt. Grundsätzlich vorstellbar ist die Ausführung als Programm, das lokal am Rechner des *Bürgers* läuft (*lokale Bürgerkarten-Umgebung*), oder als serverbasierter Dienst, der über das Internet angesprochen wird (*serverbasierte Bürgerkarten-Umgebung*). Die Interaktion mit diesem Programm bzw. Dienst wird über zwei Schnittstellen abgewickelt: Über die *Benutzer-Schnittstelle* sowie über den *Security-Layer*.

Hash-Eingangsdaten

Jene Daten, die für die Berechnung des Hash-Wertes für eine `dsig:Reference` verwendet werden. Sind für die `dsig:Reference` Transformationen angegeben, entsprechen diese Daten dem Ergebnis der letzten Transformation. Sind keine Transformationen spezifiziert, gleichen die Hash-Eingangsdaten den *Referenz-Eingangsdaten*.

Impliziter Transformationsparameter

Siehe Abschnitt 2.2.2.2, „Implizite Transformationsparameter“ in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer

Referenz-Eingangsdaten

Jene Daten, die sich aus der Auflösung der im Attribut `URI` der `dsig:Reference` angegebenen URI ergeben. Sind für die `dsig:Reference` Transformationen angegeben, werden diese Daten als Eingangsdaten zur Berechnung der ersten Transformation verwendet. Sind keine Transformationen spezifiziert, gleichen die Referenz-Eingangsdaten den *Hash-Eingangsdaten*.

Security-Layer

Jene Schnittstelle, über die die *Applikation* mit der *Bürgerkarten-Umgebung* kommuniziert. Das genaue Protokoll, das über diese Schnittstelle gesprochen werden kann, wird in Applikationsschnittstelle Security-Layer spezifiziert. Die möglichen Bindungen dieses Protokolls an Transportschichten wie HTTP oder TCP wird in Transportprotokolle Security-Layer geregelt.

Signaturmanifest

Siehe Abschnitt 2.2.2.2, „Implizite Transformationsparameter“ in Die österreichische Bürgerkarte - Applikationsschnittstelle Security-Layer .

Referenzen

[CMS] BHously, R.: *RFC 3369: Cryptographic Message Syntax (CMS)* , IETF Request For Comment, August 2002

[CMS-AES] chaad, J.: *RFC 3565: Use of the Advanced Encryption Standard (AES) Encryption Algorithm in Cryptographic Message Syntax (CMS)* . IETF Request For Comment, Juli 2003.

[CMS-Alg] Hously, R.: *RFC 3370: Cryptographic Message Syntax (CMS) Algorithms* . IETF Request For Comment, August 2002.

[CMS-RSAES-OAEP] Hously, R.: *RFC 3560: Use of the RSAES-OAEP Key Transport Algorithm in the Cryptographic Message Syntax (CMS)* . IETF Request For Comment, Juli 2003.

[CSS 2] Bert Bos, Håkon Wium Lie, Chris Lilley und Ian Jacobs: *Cascading Style Sheets, level 2* . W3C Recommendation, Mai 1998.

[EC14N] Boyer, John, Eastlake, Donald und Reagle, Joseph: *Exclusive XML Canonicalization. W3C Recommendation, Juli 2002* .

[ECDSA-CMS] Blake-Wilson, S., Brown, D., Lampert, D.: *RFC 3278: Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)* . IETF Request For Comment, April 2002.

[ECDSA-XML] Blake-Wilson, S., Karlinger, G. und Wang, Y.: *ECDSA with XML-Signature Syntax* . Internet-Draft, Jänner 2004.

[E-GovG] *BGBl. I Nr. 10/2004.*

[ESS-S/MIME] Hoffman, P.: *RFC 2634: Enhanced Security Services for S/MIME* , IETF Request For Comment, Juni 1999

[ETSICMS] European Telecommunications Standards Institute: *ETSI TS 101733: Electronic Signature Formats, v1.5.1* , Technical Specification, Dezember 2003

[ETSIQCert] European Telecommunications Standards Institute: *ETSI TS 101 862: Qualified certificate profile, v1.2.1* , Technical Specification, Juni 2001

[ETSIXML] European Telecommunications Standards Institute: *ETSI TS 101903: XML Advanced Electronic Signatures (XAdES), v1.2.2* , Technical Specification, April 2004

[GIF] *Graphics Interchange Format, Version 89a* . CompuServe Incorporated, Juli 1990.

[HTML4] Dave Ragget, Arnaud Le Hors und Ian Jacobs: *HTML 4.01 Specification* . W3C Recommendation, Dezember 1999.

[HTTP1.1] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leech und T. Berners-Lee: *Hypertext Transfer Protocol -- HTTP/1.1*. IETF Request For Comment, Juni 1999.

[HTTPS] E. Rescorla *HTTP over TLS*. IETF Request For Comment, Mai 2000

[ISO-8859-1] *ISO/IEC 8859-1*:1998: Information technology -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1.

[ISO-8859-10] *ISO/IEC 8859-10*:1998: Information technology -- 8-bit single-byte coded graphic character sets -- Part 10: Latin alphabet No. 6.

[ISO-8859-15] *ISO/IEC 8859-15*:1999: Information technology -- 8-bit single-byte coded graphic character sets -- Part 15: Latin alphabet No. 9.

[ISO-8859-2] *ISO/IEC 8859-2*:1999: Information technology -- 8-bit single-byte coded graphic character sets -- Part 2: Latin alphabet No. 2.

[ISO-8859-3] *ISO/IEC 8859-3*:1999: Information technology -- 8-bit single-byte coded graphic character sets -- Part 3: Latin alphabet No. 3.

[ISO-8859-9] *ISO/IEC 8859-9*:1999: Information technology -- 8-bit single-byte coded graphic character sets -- Part 9: Latin alphabet No. 5.

[JPEG] Eric Hamilton: *JPEG File Interchange Format, Version 1.02* . C-Cube Microsystems, September 1992.

[KEYWORDS] Bradner, S.: *RFC 2119: Key words for use in RFCs to Indicate Requirement Levels* , IETF Request For Comment, März 1997

[MIME] Freed, N. und Borenstein, N.: *RFC 2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types* , IETF Request For Comment, November 1996

[PersBin] Hollosi, Arno und Karlinger, Gregor: *XML-Definition der Personenbindung* . Konvention zum E-Government Austria erarbeitet von der Stabsstelle IKT-Strategie des Bundes, Technik und Standards. Öffentlicher Entwurf, Version 1.2.2, 14. Februar 2005.

[PersonData] Naber, Larissa: *PersonData Struktur - XML Spezifikation* . Konvention zum E-Government Austria erarbeitet von der Arbeitsgruppe Kommunikationsarchitekturen. Öffentlicher Entwurf, Version 2.0.0, 14. Oktober 2004.

[PKCS#12] RSA Laboratories: *PKCS#12 v1.0: Personal Information Exchange Syntax* , Juni 1999.

[port-numbers] Internet Assigned Numbers Authority: *Port Numbers*

[QCert] Santesson, S. und Nystrom M.: *RFC 3739: Internet X.509 Public Key Infrastructure: Qualified Certificates Profile* , IETF Request For Comment, März 2004

[SigG] *BGBl I Nr. 190/1999* idF BGBl I Nr. 152/2001.

[SigV] *BGBl II Nr. 30/2000* idF BGBl II Nr. 527/2004.

[Stammzahl] Hollosi, Arno und Hörbe, Rainer: *Bildung von Stammzahl und bereichsspezifischem Personenkennzeichen (bPK)* . Konvention zum E-Government Austria erarbeitet von der Stabsstelle IKT-Strategie des Bundes, Technik und Standards sowie vom Bundesminsterium für Inneres. Öffentlicher Entwurf, Version 1.0, 2. Februar 2004.

[TLS] T. Dierks und C. Allen: *The TLS Protocol Version 1.0* . IETF Request For Comment, Januar 1999.

[Unicode] The Unicode Consortium. *The Unicode Standard, Version 4.0.0* , defined by: The Unicode Standard, Version 4.0 (Boston, MA, Addison-Wesley, 2003. ISBN 0-321-18578-1).

[URI] Berners-Lee, T. , Fielding, R. und Masinter, L.: *RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax* , IETF Request For Comment, August 1998

[VerwEig] Hollosi, Arno: *X.509 Zertifikatserweiterungen für die Verwaltung* . Konvention zum E-Government Austria erarbeitet von der Stabsstelle IKT-Strategie des Bundes, Technik und Standards. Öffentlicher Entwurf, Version 1.0.3, 21. Februar 2005.

[X509] Polk, W., Ford, W., Solo, D.: *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* . IETF Request For Comment, April 2002.

[XHTML 1.1] Murray Altheim, Frank Boumphrey, Sam Dooley, Shane McCarron, Sebastian Schnitzenbaumer und Ted Wugofski: *Modularization of XHTML* . W3C Recommendation, April 2001.

[XHTML MOD] Daniel Austin, Subramanian Peruvemba, Shane McCarron, Masayasu Ishikawa: *Modularization of XHTML in XML Schema* . W3C Working Draft, Oktober 2003.

[XML] Bray, Tim, Paoli, Jean, Sperberg-McQueen, C.M. und Maler, Eve: *Extensible Markup Language (XML) 1.0 (Second Edition)* , W3C Recommendation, Oktober 2000.

[XMLDecTF] Hughes, Merlin, Imamura, Takeshi und Maruyama, Hiroshi: *Decryption Transform for XML Signature* . W3C Recommendation, Dezember 2002.

[XMLDSIG] Eastlake, Donald, Reagle, Joseph und Solo, David: *XML-Signature Syntax and Processing* , W3C Recommendation, Februar 2002

[XMLDSIG-URI] Eastlake, Donald: *RFC 4051: Additional XML Security Uniform Resource Identifiers (URIs)* , IETF Request For Comments, April 2005

[XMLEnc] Eastlake, Donald und Reagle, Joseph: *XML Encryption Syntax and Processing* , W3C Recommendation, Dezember 2002

[XML-Schema] Thompson, Henry S., Beech, David, Maloney, Murray und Mendelson, Noah: *XML Schema Part 1: Structures* , W3C Recommendation, Mai 2001

[XMLTYPE] Murata, M., St.Laurent, S., und Kohn, D.: *RFC 3023: XML Media Types* , IETF Request For Comment, Jänner 2001.

[XPath] Clark, James und DeRose, Steven: *XML Path Language* , W3C Recommendation, November 1999

[XPF2] Boyer, John, Hughes, Merlin und Reagle, Joseph: *XML-Signature XPath Filter 2.0* . W3C Candidate Recommendation, Juli 2002.

[XPointer] Grosso, Paul, Maler, Eve, Marsh, Jonathan und Walsh, Norman: *XPointer Framework* . W3C Recommendation, März 2003.

[XSS-FAQ] Cgisecurity.com: *The Cross Site Scripting FAQ* .

A. Historie

Datum	Version	Änderungen
20.02.2008	1.2.2	• Änderungen aus *Erweiterung Spezifikation Bürgerkarte - Auslesen von Infoboxen im Push-Verfahren* in Abschnitt 3.2, „Ablauf“ und Abschnitt 3.3, „Kodierung“ aufgenommen.
01.03.2005	1.2.1	• Erratum Erratum 31 in Die österreichische Bürgerkarte - Errata korrigiert.
14.05.2004	1.2.0	• Errata 16 und 19 laut Errata ausgebessert. • Bestimmungen zur Identifikation der BKU mittels HTTP-Headers Server und User-Agent in Abschnitt 3.3.2 aufgenommen. • Unklarheiten in Abschnitt 3.2.3 ausgeräumt. • Weitergabe-Header eingeführt. • Dokument reorganisiert. • Umgebungsvariablen gestrichen. • Erfolgt der HTTP(S)-Request an eine serverbasierte BKU, darf die URL vom Diensteanbieter beliebig gewählt werden. • Eine serverbasierte BKU darf zwischen den Schritten 2 und 3 bzw. 2 und 7 des Ablaufs in Abschnitt 3.2 auch die Kommunikation der Benutzerschnittstelle in der Broswerverbindung abwickeln.
31.08.2003	1.1.0	• Überarbeitung der HTTP-Bindung. Response und Auswirkungen der DataURL genau spezifiziert. (Abschnitt 3.3) • Auswirkungen auf Sicherheit bei Verwendung der HTTP-Bindung genauer erläutert. (Abschnitt 3.6) • RedirectURL kann auch Code 302 liefern. (Abschnitt 3.3) • Multipart/form-data als Kodierung bei HTTP-Bindung auch akzeptiert (Abschnitt 3.4) • Weitergabe-Felder für HTTP-Bindung definiert (Abschnitt 3.4) • Referzierung von Formularfeldern möglich (Abschnitt 3.5) • Beispiele üebrarbietet (Abschnitt 3.7), sowie Anwendungsfälle skizziert (Abschnitt 3.8) • Durchgängiges Ersetzen des Begriffs "Security-Kapsel" durch "Bürgerkarten-Umgebung"
18.03.2003	1.0.1	• Eintragung der IANA-registrierten Portnummern als Defaultportnummern für TCP/IP, TLS, HTTP, HTTPS • Referenz auf IANA Liste der registrierten Portnummern eingefügt. • Fehler bei Kapitelnumerierung in Abschnitt 3 korrigiert. • Zu benützende Anfrage-URL in Abschnitten 3.4 und 5.4 auf durchgängige

		Kleinschreibung umgestellt. • In Abschnitt 5.4 ResponseType entsprechend mit in Dokumentation aufgenommen • In Abschnitt 3.4 zu DataURL "*Als Protokolle sind verpflichtend HTTP und HTTPS zu unterstützen.*" explizit in Dokument aufgenommen.