Ein Tochterunternehmen der
Kassenärztlichen Bundesvereinigung

Spezifikation KV-Connect Server 2.7

Herausgeber:
KV Telematik GmbH

Inhaltsverzeichnis

1 Historie

Vers.	Datum	Autor	Kap.	Änderung	Status
2.7	15.01.2019	Sascha Fagel	keine	keine	In Kraft
2.7A	20.08.2018	Sascha Fagel	alle	initiale Erstellung	Ankündigung

2 Einleitung

Hier wird die REST-Schnittstelle des KV-Connect Servers beschrieben sowie das Format der Daten, die über diese REST-Schnittstelle übertragen werden. Der Server stellt folgende Dienste zur Verfügung:

- Ändern von Benutzerpasswörtern
- Erstellen, Abrufen und Zurückziehen von Zertifikaten
- Senden von Mails
- Abrufen von Mails
    - Abrufen einzelner Mails über ihre *Message-ID*
    - Abrufen aller Mails eines Benutzers
    - Abrufen aller Header
- Verzeichnisdienste
- diverse Hilfsdienste

3 Inhalt

- Unter Übergreifende Aspekte finden sich grundsätzliche Hinweise zu REST im allgemeinen und zur Benutzung der konkreten Schnittstelle.
- Auf der Seite Kryptographische Standards werden die kryptographischen Standards zusammengefasst, die Clients erfüllen müssen, um auf die REST-Schnittstelle zuzugreifen.
- Unter MIME-Format findet sich Grundsätzliches zum Format von Nachrichten.
- Die gültige Passwortrichtlinie.
- Die Seite REST-Ressourcen enthält einen Überblick aller angebotenen Ressourcen und verweist auf die Detail-Definitionen.
- Weitere REST-Schnittstellen, Formate und Grundsätze zur Anbindung mobiler Geräte.

4 Änderungen gegenüber V2.6

Änderung	Grund / Bemerkung	Auswirkung auf PVS / Client
Ressource zum verschlüsselten Abruf von Zertifikaten hinzugefügt	KV-Connect Mobile in KV-Connect Server integriert	keine
Spezifikation einer externen Ressource zum Abruf von Zertifikaten hinzugefügt		
Formate zum Nachrichtenaustausch mit mobilen Geräten hinzugefügt		
Ressource Abruf von Zertifikaten mittels Email-Adresse hinzugefügt	Vereinfachte optionale Methode des Zertifikats-Abrufs zusätzlich zum Abruf mittels UID	keine, da optional
CSR-Ressource: Response 503 entfernt	obsolet	keine
Suchen von Accounts ohne / vor dem ?	Vereinheitlichung der Spezifikation mit anderen REST-Ressourcen	keine, da die bisherige Schreibweise mit /? weiterhin unterstützt wird

5 Übergreifende Aspekte

- Grundsätzliches
- REST und HTTP
- URL-Format
- Kodierung
- Authentisierung und Authorisierung
- Request Header
- Responses
- Response Header

5.1 Grundsätzliches

Auf dem Server persistent sind Mails, Zertifikate und die relevanten Benutzerdaten. Die Zuordnung der Daten zu Accounts erfolgt über den systemweit eindeutigen Login-Namen, der dem localpart der KV-Connect Adresse entspricht. Jede Mail ist mit einer eindeutigen *Message-ID* gekennzeichnet, mit der man den Inhalt einer Mail abrufen kann. Dabei (und bei fast allen anderen Operationen) muss sich der Empfänger authentisieren. Der Zugriff erfolgt über eine REST-Schnittstelle. In der Nomenklatur von REST bezeichnet man logisch nicht trennbare Daten als *Ressourcen.* Neben der URL ist für das Verhalten der Schnittstelle die gewählte HTTP-Methode (GET, POST, PUT, DELETE) ausschlaggebend. Das Abrufen von Ressourcen geschieht immer mit `GET`, das Anlegen einer Ressource mit `POST`, die Modifikation einer Ressource mit `PUT` und das Löschen einer Ressource mit `DELETE`.

5.2 REST und HTTP

Die Spezifikation bezieht sich auf die HTTP-RFCs 7230, 7231, 7232, 7233, 7234 und 7235 (diese haben den ursprünglichen HTTP-RFC 2616 abgelöst) und den Grundsätzen REST-basierter Webservices.

5.3 URL-Format

Die URLs der Schnittstelle sind immer nach folgendem Schema aufgebaut:

```
<base_url>/<resource_name>[/<resource_id>]
```

wobei die Basis-URL durch den benutzten Server bestimmt wird und als Ganzes konfiguriert wird:

```
<base_url>=http[s]://<host>:[<port>]/<contextpath>
```

Also zum Beispiel für das aktuelle Produktivsystem:

```
https://kvlink1.kv-safenet.de/kvcserver/rest
```

ℹ Für detaillierte Informationen zur Anbindung siehe Anbindung an die Server.

Der `<resource_name>` entspricht dem Namen der angefragten Ressource, z.B. `mails`. Wenn auf eine konkrete Ressource zugegriffen werden soll, ist deren `<resource_id>` erforderlich, im Falle einer Mail z.B. ihre *Message-ID*.

Für sämtliche URLs gilt RFC 3986, wobei Ressourcen und Query-Parameter url-enkodiert werden müssen. Insbesondere muss die Abtrennung des Query-Teils eines Requests durch ein Fragezeichen '?' und nicht mit '%3F' erfolgen.

5.4 Kodierung

Der Server interpretiert jeden Datenstrom als UTF-8 und kodiert alle Responses ebenso im UTF-8 Format. Abweichungen oder Besonderheiten werden bei den einzelnen Ressource beschrieben.

5.5 Authentisierung und Authorisierung

Requests werden mit HTTP Basic Authentication (RFC 2617) authentisiert. Ob für eine Ressource eine Authentisierung erforderlich oder der Zugriff nur dem Besitzer der Ressource erlaubt ist, wird bei den jeweiligen REST-Ressourcen beschrieben.

5.6 Request Header

Die folgenden Request-Header sollen (sofern sinnvoll) bei jedem Request vom Client mitgeschickt werden. Die fett-gedruckten sind laut RFC 7231 verpflichtend und müssen vom Client mitgeschickt werden.

In den Beispielen zu den einzelnen Ressourcen **werden diese Header nicht extra aufgeführt**, um die Übersichtlichkeit zu wahren.

Andere gültige HTTP-Header können natürlich auch gesetzt werden, werden aber ggf. vom KV-Connect Server ignoriert.

Header	Beschreibung	Beispiel	Kommentar
Content-Type	MIME-Type der Nutzdaten	Content-Type: text/plain; charset=UTF-8	muss zur Ressource passen
Host	Hostname des Servers	Host: kvlink1.kv-safenet.de	zwingend Gemäß HTTP/1.1 Spezifikation
Date	Datum des Sendens	Date: Tue, 14 Aug 2018 08:12: 31 GMT	Format siehe RFC 7231 Abschnitt 7.1.1.2
User-Agent	Bezeichnung des Clients	User-Agent: KV-Connect Client v1.2.3	def. in RFC 7231 Abschnitt 5.5.3

ⓘ Sonderfall

Der Request-Header "Accept" kann ausschließlich mit der Funktion/Ressource "Erweiterte Account-Suche" sinnvoll benutzt werden. Nur dort kann zwischen XML und JSON als gewünschtes Response-Format differenziert werden. Bei allen anderen Ressourcen wird dieser Header bestenfalls ignoriert und kann in Einzelfällen zu Fehlern führen.

5.7 Responses

Die folgenden HTTP Status-Codes können bei dem Zugriff auf jede Ressource geschickt werden und werden in den Detailbeschreibungen ggf. nicht extra aufgeführt.

Nummer	Status	Beschreibung
200	OK	Erfolgreicher Abruf
201	CREATED	Neue Ressource angelegt
202	ACCEPTED	Request angenommen, aber noch nicht abgearbeitet
400	BAD REQUEST	Fehler im Request

Nummer	Status	Beschreibung
401	UNAUTHORIZED	keine Authentisierung
403	FORBIDDEN	keine Berechtigung
404	NOT FOUND	Resource nicht vorhanden
408	REQUEST TIMEOUT	
415	UNSOPPORTED MEDIA TYPE	content-Type des Requests entspricht nicht der Spezifikation, z. B. text/plain
500	INTERNAL SERVER ERROR	Serverseitiger Fehler

5.8 Response Header

Die folgenden Response-Header werden (sofern sinnvoll) bei jedem Response vom Server mitgeschickt.

In den Beispielen zu den einzelnen Ressourcen **werden diese Header nicht extra aufgeführt**, um die Übersichtlichkeit zu wahren.

Header	Beschreibung	Beispiele	Kommentar
Allow	erlaubte HTTP-Methoden	Allow: GET, DELETE	def. in RFC 7231 Abschnitt 7.4.1
Cache-Control	definiert Caching-Parameter für Clients und Proxies	Cache-Control: max-age=3600	siehe RFC 7234 Abschnitt 5.2
Content-Language	Sprache der Antwort	Content-Language: de	def. in RFC 7231 Abschnitt 3.1.3.2
Content-Length	Größe der Nutzdaten in Octets	Content-Length: 54353	def. in RFC 7230 Abschnitt 3.3.2
Content-Type	MIME-Type der Nutzdaten im Response	Content-Type: application /xml;charset=UTF-8	def. in RFC 7231 Abschnitt 3.1.1.5
Date	Datum des Sendens	Date: Tue, 14 Aug 2018 08: 12:31 GMT	Format siehe RFC 7231 Abschnitt 7.1.1.2
Last-Modified	Datum der letzten Änderung	Last-Modified: Tue, 14 Aug 2018 08:12:31 GMT	Format siehe RFC 7232 Abschnitt 2.2
Server	Serverdetails	Server: KV-Connect Server REST 2.7.0	def. in RFC 7231 Abschnitt 7.4.2

6 Kryptographische Standards

6.1 Einleitung

Kryptographie spielt in KV-Connect die zentrale Rolle. Da nicht nur native, d.h. von der KV-Telematik GmbH entwickelte Clients eingesetzt werden, sondern auch 3rd-Party Clients von PVS/KIS-Herstellern, ist es sehr wichtig, dass die hier beschriebenen Standards von allen beteiligten Clients eingehalten werden, um den Hersteller-übergreifenden Nachrichtenaustausch zu gewährleisten.

Jeder Client muss dazu in der Lage sein:

- ein RSA-Schlüsselpaar definierter Länge zu erzeugen und den privaten Schlüssel sicher in einem Keystore abzulegen
- aus dem privaten Schlüssel einen Certificate Sigining Request (CSR) gemäß PKCS#10 zu erzeugen und über die REST-Schnittstelle an den KV-Connect Server zu schicken
- das erzeugte x509-Zertifikat abzulegen
- mit diesem Zertifikat und dem dazugehörigen privaten Schlüssel signierte und verschlüsselte Nachrichten im S/MIME-Format zu erzeugen und über die REST-Schnittstelle zu verschicken
- die Gültigkeit von fremden Zertifikaten zu überprüfen
- empfangene Nachrichten zu entschlüsseln und die Signatur des Absenders zu überprüfen

Die Details zu den einzelnen Anforderungen werden im Folgenden ausgeführt. Die Vorgaben für Schlüssellängen und Algorithmen leiten sich aus [7] ab.

6.2 RSA-Schlüsselpaar

Das zu erzeugende Schlüsselpaar ist ein RSA-Schlüsselpaar gemäß PKCS#1 / RFC 3447 mit einer Länge von **2048 Bits**.

6.3 Certificate Signing Request

Ein KV-Connect-CSR ist gemäß RFC 2986 aufgebaut. Im *Subject*-Feld wird ein *X500Uniqueldentifier*-Attribut eingetragen, das den Login-Namen des Benutzers UTF-8-kodiert als *BitString* enthält, siehe RFC 2256. Die Signatur muss mit dem Algorithmus *sha256WithRSAEncryption* erzeugt werden.

Die Ausgabe von OpenSSL zu einem im PEM-Format vorliegenden Beispiel-CSR sieht wie folgt aus:

```
$ openssl req -noout -text -in csr.pem

Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: x500UniqueIdentifier=test.nutzer.kvtg
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:d2:02:ed:f9:1e:f8:1a:af:f0:0e:e2:e7:a1:54:
                    ...
                    45:d9:0d:ee:6c:0a:7b:ab:b6:a7:f5:fe:3a:96:f0:
                    42:69:2d
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: sha256WithRSAEncryption
         25:ff:49:a5:3f:ba:d4:ee:da:d5:b1:64:05:67:49:aa:25:ff:
         ...
         01:61:ec:a5:42:88:31:f1:f2:d3:80:11:24:83:1d:0c:86:ec:
         c5:57:66:47:ff:f0:81:45
```

6.4 S/MIME erzeugen

MIME Nachrichten werden zunächst mit dem privaten Schlüssel des Absenders signiert und anschließend mit den öffentlichen Schlüsseln aller Empfänger und des Absenders verschlüsselt, letzteres dient dazu, dass der Absender seine gesendeten Nachrichten ggf. selbst wieder entschlüsseln kann. Die Header der ursprünglichen Nachricht werden unverändert übernommen.

6.4.1 Signatur

Da es sich bei der zu signierenden Nachricht um eine MIME-Nachricht handelt, erfolgt die Signaturerstellung nach den Regeln des S/MIME Standards. Von den zur Verfügung stehenden Signaturformaten *(multipart/signed* oder *application/pkcs7-mime*) wird das Format ***multipart/signed*** verwendet. Das resultierende MIME-Objekt besteht daher aus zwei Teilen bzw. Bodyparts:

- Der erste Teil enthält die ursprüngliche (zu signierende) Nachricht.
- Der zweite Teil ist vom MIME Inhaltstyp *application/pkcs7-signature* und enthält die über den ersten Teil berechnete Signatur und das entsprechende Zertifikat inkl. Zertifikatskette als Base64 kodiertes CMS SignedData Objekt.

Beispiel:

```
Content-Type: multipart/signed; micalg=sha-256;
protocol="application/pkcs7-signature";
boundary="----=_Part_1_8306951.1300877941954"

------=_Part_1_8306951.1300877941954
Content-Type: multipart/mixed;
boundary="----=_Part_0_2704014.1300877941798"
------=_Part_0_2704014.1300877941798
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Das ist eine Test-Nachricht.

------=_Part_0_2704014.1300877941798
Content-Type: application/octet-stream; name=test.pdf
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=test.pdf

JVBERi0xLjQNJeLjz9MNCjYgMCBvYmogPDwvTGluZWFyaXplZCAxL0wgODY2NC9PIDgv
RSA0NTQ3L04gMS9UIDg0OTgvSCBbIDUxNiAxNThdPj4NZW5kb2JqDSAgICAgICAgICAg
...
MDAwMDAgbg0KMDAwMDAwNDYwMyAwMDAwMCBuDQowMDAwMDA0NjUzIDAwMDAwIG4NCjAw
MDAwMDgyNjcgMDAwMDAgbg0KdHJhaWxlcg0KPDwvU2l6ZSA2Pj4NCnN0YXJ0eHJlZg0K
------=_Part_0_2704014.1300877941798--

------=_Part_1_8306951.1300877941954
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIIF
vTCCAwEwggJqoAMCAQICAw5UkTANBgkqhkiG9w0BAQUFADBOMQswCQYDVQQGEwJBVDEN
...
3gfXOm9sUf6MXnYRb7tYKqU3HwAWXwA+RdSXk1Lkr3k5emAUGYzKwJM7JCWWTv7te5oV
hFaqUuHY4FUNXgc80Y0CRJu+GjdgigfRAAAAAAAA

------=_Part_1_8306951.1300877941954--
```

Der Content-Type *application/pkcs7-signature* wird veraltet auch als *application/x-pkcs7-signature* angegeben. Laut RFC 2311 Abschnitt C.1 sollen beide Varianten gleich behandelt werden. Ein Client muss Nachrichten mit der Variante *application/pkcs7-signature* erzeugen, aber bei der Prüfung beide Varianten akzeptieren.**Die Zertifikatskette des Senders** (und nicht nur das Zertifikat) wird in das *SignedData-Objekt* mit eingepackt und mit der signierten Nachricht zusammen gesendet.Um eine kryptographische Bindung des Zertifikates an die Signatur zur erreichen, wird zusätzlich zu den CMS-Standardattributen *ContentType* (CMS-Typ der Nachricht), *MessageDigest* (Hashwert der Nachricht) und *SigningTime* (Zeitpunkt der Signaturerstellung) noch das ESS-Attribut *SigningCertificateV2* mitsigniert. Dieses Attribut enthält einen Hashwert über das Zertifikat und identifiziert dieses somit in eindeutiger Weise. Da dieser Hashwert in die Signatur einfließt, könnte zwar das (Signatur-)Zertifikat von einem Angreifer ausgetauscht werden, nicht aber der Hashwert im *SigningCertificateV2*-Attribut.

Aufgrund der strengen Einschränkungen der kryptographischen Formate, ist es nicht notwendig, die kryptographischen Fähigkeiten des verwendeten S/MIME-Clients über das Hinzufügen eines *SMimeCapabilities* Attributes mitzuteilen.

Die für die Signaturerstellung verwendeten Signatur- und Hash-Algorithmen ergeben sich aus den Empfehlungen in [3] und [4]. Die aktuellen Werte (in fett) ergeben sich aus der festgelegten Schlüssellänge von 2048.

Schlüssellänge	Signaturalgorithmus	Hashalgorithmus

l <= 1024	rsaEncryption (1.2.840.113549.1.1.1)	SHA-1 (1.3.14.3.2.26)
1024 < l <= 3072	**rsaEncryption (1.2.840.113549.1.1.1)**	**SHA-256 (2.16.840.1.101.3.4.2.1)**
3072 < l <= 7680	rsaEncryption (1.2.840.113549.1.1.1)	SHA-384 (2.16.840.1.101.3.4.2.2)
l > 7680	rsaEncryption (1.2.840.113549.1.1.1)	SHA-512 (2.16.840.1.101.3.4.2.3)

Tabelle 1: Signatur- und Hashalgorithmen für RSA Schlüssel

⚠ Jeder Client muss darüber hinaus in der Lage sein, auch Signaturen mit SHA-384 und SHA-512 zu verarbeiten und zu prüfen.

Anmerkungen zu den Algorithmenbezeichnern

Zu beachten ist, dass bei RSA der Signatur-Algorithmus über den Algorithmus-Bezeichner 1.2.840.113549.1.1.1 (rsaEncryption) angegeben wird, der eigentlich das PKCS#1v1.5 Verschlüsselungsschema RSAES-PKCS1-v1_5 referenziert [8,9]. Diese Art der Bezeichnung hat CMS [5] von seinem Vorgänger PKCS#7 [6] übernommen. PKCS#7 war auf das PKCS#1v1.5-RSA Kryptosystem zugeschnitten, in dem bei der Signaturberechnung der Hashwert (und der Hash-Algorithmus-Bezeichner) mit dem privaten Schlüssel des Unterzeichners verschlüsselt wird. Bei PKCS#7 wird deshalb auch nicht von einem Signature-Algorithmus, sondern von DigestEncryption-Algorithmus gesprochen. Da der verwendete Hash-Algorithmus auch bei CMS in einem eigenen Feld angegeben wird, hat CMS die Bezeichnung rsaEncryption für den Signatur-Algorithmus beibehalten. Es ist jedoch auch erlaubt, einen Algorithmus-Bezeichner zu verwenden, der den Hash-Algorithmus inkludiert, wie es z.B. bei ECDSA-Signatur-Algorithmen der Fall ist ([1,2,4,10]).

6.4.2 Verschlüsselung

Beim Verschlüsseln der signierten Nachricht muss berücksichtigt werden, dass die Nachricht an einen oder mehrere Empfänger gesendet werden kann. Für jeden Empfänger wird ein entsprechendes Verschlüsselungszertifikat benötigt. Beim Verschlüsseln einer Nachricht wird das dazu notwendige Verschlüsselungszertifikat des Empfängers vom Verzeichnisdienst oder dem KV-Connect Server bezogen. Deshalb ist es nicht notwendig, Verschlüsselungszertifikate über signierte Nachrichten auszutauschen.

- Entsprechend den Vorgaben des S/MIME Standards wird zur Verschlüsselung ein hybrides Verfahren verwendet, bei dem die (signierte) Nachricht zunächst mit einem temporär erzeugten symmetrischen Schlüssel (*Content-Encryption-Key*) verschlüsselt wird.
- Dazu wird unabhängig von der Schlüssellänge des Zertifikats *AES256-CBC (2.16.840.1.101.3.4.1.42)* verwendet, wobei ein 256-Bit langer *Content-Encryption-Key* erzeugt wird.
- Dieser symmetrische Schlüssel wird dann mit dem öffentlichen Schlüssel des jeweiligen Empfängers per *rsaEncryption(1.2.840.113549.1.1.1)* verschlüsselt.
- Die resultierende verschlüsselte Nachricht hat den MIME Inhaltstyp *application/pkcs7-mime* (mit S /MIME-Typ *enveloped-data*). Ihren Inhalt bildet ein Base64 kodiertes *CMS EnvelopedData Objekt*, das die verschlüsselten Daten mit allen zur Entschlüsselung benötigten Parametern enthält.

⚠ Die Zertifikate der Empfänger müssen zum Zeitpunkt des Sendens vom Verzeichnisdienst abgerufen werden und dürfen nicht lokal gecacht werden. Dadurch wird sicher gestellt, dass immer das aktuelle Zertifikat verwendet wird, insbesondere auch, dass dieses existiert und nicht zurückgerufen wurde. Liegt im Verzeichnisdienst kein Zertifikat vor, darf dem Empfänger keine Nachricht mit einem alten Zertifikat gesendet werden, da die Vertraulichkeit der Daten nicht mehr garantiert werden kann.

Beispiel

```
Message-ID: <7453560.1300877942829.JavaMail.example@com>
Date: Tue, 14 Aug 2018 08:12:31 +0100 (CET)
From: test.sender.kvtg@kv-safenet.de
To: test.recipient.kvtg@kv-safenet.de
Subject: Example Mime-Version: 1.0
X-KVC-Dienstkennung: 1ClickAbrechnung;Lieferung;V2.1
X-KVC-Sendersystem: TestClient;V1.0
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
name=smime.p7m
Content-Transfer-Encoding: base64 Content-Disposition: attachment;
filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggHeMIHsAgEAMFUwTjELMAkGA1UEBhMCQVQxDTALBg
NVBAoTBE1BSUsxFTATBgNVBAsTDEphdmFTZWN1cml0eTEZMBcGA1UEAxMQSUFJSyBSU0Eg
VGVzdCBDQQIDBfWAMA0GCSqGSIb3DQEBAQUABIGAKg7v9BwmyAxM0S7kWmTtnaOlhOMHrG
...
DsEas9Rq3mq59CSgUDB/zlV+Xll9o6Kg9//+rZ7br2h4d/PR5cTFPtdvDWU2ALTWt6oBbt
V/aHRayO/8dBF6gyJdipctqaUCrGkRJRNWqZH0CfLjokzBPJtQu5E/7rDE04xwxsJoOULX
emLUy8e0Of3H208dMCSBHJ31RyxM29PxA1zvIOQg8dJN7z2c3BQr46EGbC4xrchkhVwqZJ
xLcpcSSxkgeIdn8723I063k83Q2c9+9YrldzEKFgAAAAAAAAAAAA=
```

6.5 S/MIME empfangen

Auf der Empfangsseite wird die Nachricht in umgekehrter Reihenfolge zuerst entschlüsselt und dann geprüft.

6.5.1 Entschlüsselung

Es sind alle zur Entschlüsselung benötigten Parameter in der verschlüsselten Nachricht enthalten. Es fehlt lediglich der private Schlüssel des Empfängers, mit dem der *Content-Encryption-Key* wiedergewonnen werden kann. Die signierte Nachricht kann dann mit *AES256-CBC (2.16.840.1.101.3.4.1.42)* wieder entschlüsselt werden.

6.5.2 Signaturprüfung

Im Zuge der Signaturprüfung wird zunächst die mathematische Korrektheit der Signatur mit Hilfe des in der signierten Nachricht mit gesendeten Zertifikats des Senders überprüft. Ist die Signaturprüfung erfolgreich verlaufen, so wird die Zertifikatskette des Senders und der Status des Zertifikates überprüft. Die zur Kettenvalidierung benötigten vertrauenswürdigen Zertifikate (Trust-Anchors) müssen dem Client als vertrauenswürdig bekannt sein. Dies sind die öffentlich publizierten RootCA- bzw. UserCA-Zertifikate der KV Telematik GmbH. Die Kettenvalidierung verlangt nicht, dass der aktuelle Status des Sender-Zertifikates geprüft wird, sondern nur, dass das Zertifikat zum Zeitpunkt des Sendens gültig war.

6.6 Glossar

CA	Certication Authority; Einrichtung zur Austellung von digitalen Zertikaten
CBC	Cipher Block Chaining; Betriebsmodus, um eine Block-Chiffre mit einem Feedback-Mechanismus zu versehen, bei dem jeder Klartextblock vor seiner Verschlüsselung mit dem letzten Ciphertextblock XOR-verknüpft wird
CMS	Cryptographic Message Syntax: Standard zur kryptographischen Absicherung von digitalen Daten; beschrieben in RFC 5751
DH	Diffie-Hellman; Public key Algorithmus; verwendet zum Schlüsselaustausch
DSA	Digital Signature Algorithm; digitaler Public-Key Signatur Algorithmnus. Standardisiert im Digital Signature Standard (DSS).
ESS	Enhanced Security Services for S/MIME; erweitert das S/MIME Protokoll um einige zusätzliche Sicherheitsdienste; beschrieben in RFC 2634, RFC 5035
IAIK-JCE	Software-Paket, das einen JCA/JCE-Provider implementiert
JCA	Java Cryptography Architecture: Java API zur Berechnung von Hashwerten und Signaturen sowie Dekodieren und Verizieren von digitalen Zertikaten
JCE	Java Cryptography Extension: Erweitert die JCA um kryptographische Dienste zur Daten-Ver-/Entschlüsselungs und Berechnung von Message Authentication Codes
MIME	Multipurpose Internet Mail Extensions: Deniert ein Format, um beliebig strukturierte Daten über das SMTP-Protokoll übertragen zu können; beschrieben in RFC 2045 - RFC 2049
OCSP	Online Certicate Status Protocol: Deniert ein Verfahren zur Abfrage von Zertikatstatusinformation, beschrieben in RFC 2560
PKCS	Public Key Cryptography Standard; Serie von Standards für Public-Key Kryptographie, die von den RSA-Laboratories herausgegeben werden
RSA	Public-key Algorithmus, entwickelt von Ron Rivest, Adi Shamir und Leonard Adleman; kann sowohl zum Verschlüsseln als auch zum Signieren verwendet werden.
S /MIME	Secure MIME: Erweiterung des MIME-Standards, um Nachrichten kryptographisch absichern (signieren und/oder verschlüsseln) zu können; beschrieben in RFC 5751
X.509	ITU-T (International Telecommunication Union) Empfehlung für ein Authentizierungssystem und eine Zertikat-Syntax; Zertikat-Prol der IETF-PKIX Arbeitsgruppe

6.7 Literatur

[1] ANSI X9.63-2001: Public Key Cryptograhpy for Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography, November 2001.

[2] FIPS 186-4: Digital Signature Standard. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf, Juli 2013.

[3] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. NIST Special Publication 800-57: Recommendation for Key Management - Part 1: General (Revised).
NIST Special Publication 800-57, NIST, March 2007. http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf.

[4] S. Blake-Wilson, D. Brown, and P. Lambert. Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS).
RFC 3278, RFC Editor, April 2002. http://www.ietf.org/rfc/rfc3278.txt.

[5] R. Housley. Cryptographic Message Syntax (CMS). RFC 5751, RFC Editor, July 2004. http://www.ietf.org/rfc/rfc5751.txt.

[6] B. Kaliski. PKCS #7: Cryptographic Message Syntax. RFC 2315, RFC Editor, March 1998. http://www.ietf.org/rfc/rfc2315.txt.

[7] BSI, TR-02102-1, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, 10.2.2014, https://www.bsi.bund.de/cae/servlet/contentblob/477256/publicationFile/30924/BSI-TR-02102_V1_0_pdf.pdf

[8] R. Housley. Cryptographic Message Syntax (CMS) Algorithms. RFC 3370, RFC Editor, August 2002. http://www.ietf.org/rfc/rfc3370.txt.

[9] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1. RFC 3447, RFC Editor, February 2003. http://www.ietf.org/rfc/rfc3447.txt.

[10] S. Turner. Using SHA2 Algorithms with Cryptographic Message Syntax. RFC 5754, RFC Editor, January 2010. http://www.ietf.org/rfc/rfc5754.txt.

7 MIME-Format

7.1 Nachrichtenformat MIME

Nachrichten, die mit KV-Connect versendet werden sollen, sind im MIME-Format zu erstellen, und zwar so, dass sie im nächsten Schritt in S/MIME-Nachrichten transformiert werden können, was zusätzliche Anforderungen an sie stellt. Sie sind gemäß der folgenden Standards aufzubauen:

- RFC 5322 **(Internet Message Format)**
  RFC 5322 beinhaltet unter anderem die Spezifikation von *header fields* wie `message-id` und `orig-date`.

> ⚠ **Wichtiger Hinweis zum Bcc-Feld:** Aufgrund der Verschlüsselungsinformationen in S /MIME-Nachrichten können Empfänger prinzipiell alle anderen Empfänger sichtbar machen. Daher ist BCC nicht zu verwenden.

- RFC 2045 und RFC 2046 **(Multipurpose Internet Mail Extensions (MIME))**
- RFC 5751 **(Secure/Multipurpose Internet Mail Extensions (S/MIME))**
  Bei der Erstellung der MIME-Nachrichten ist aus diesem RFC das Kapitel "3.1. Preparing the MIME Entity for Signing, Enveloping, or Compressing" zu berücksichtigen.

> ⚠ Bitte beachten: Als "end-of-line delimiter" muss immer `<CR><LF>` genutzt werden, ein reines <LF> reicht nicht. Dies gilt auch unter Linux.

7.2 Format der Message-Id

KV-Connect erwartet bei gesendeten Mails eine Message-Id, das Format dieser Id entspricht der RFC 5234. Im wesentlichen besteht die Message-Id aus zwei nicht-leeren Strings die durch ein "@" Zeichen getrennt werden, am Anfang und Ende dieses Ausdrucks stehen ein "<", bzw. ">" Zeichen.

```
message-id = "<" text "@" text ">"
text       = 1*(ALPHA / DIGIT / other )
other      = "!" / "#" / "$" / "%" / "&" / "'" / "*" / "+" / "-" / "/" / "="
```

Bei fehlender Message-Id oder falschem Format der Message-Id sendet der Server als Antwort den HTTP Status Code 400.

> ⚠ **Achtung** Bei Benutzung der REST-Schnittstelle des Servers, müssen die erlaubten Zeichen ggf. gemäss RFC 3986 encodiert werden.

7.3 Header Felder mit Nicht-ASCII Text

Grundsätzlich kann jeder beliebige Zeichensatz verwendet werden. Bitte beachten Sie, dass Sie bei den Headern z.B. im Subject/Betreff für Zeichen ausserhalb des ASCII Zeichensatzes den zugehörigen Zeichensatz mitangeben sollten. Dies wird bei

- RFC 2047 **(MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text)**
- bzw. u.a. RFC 6532 **(Internationalized Email Headers)**

geregelt.

Beispiel:

```
Subject: Eingangsbestätigung

sollte in

Subject: =?UTF-8?Q?Eingangsbest=C3=A4tigung?=

umkodiert werden, wenn der Subject/Betreff in UTF-8 interpretiert werden sol
```

8 Passwortrichtlinie

KV-Connect Passwörter müssen den folgenden Anforderungen genügen:

- mindestens 8 Zeichen lang
  Je länger, desto sicherer; max. 200 Zeichen möglich.
- mindestens zwei Großbuchstaben enthalten
  Als Großbuchstaben zählen die Zeichen A - Z; *nicht* dazu gehören Ä, Ö, Ü .
- mindestens zwei Kleinbuchstaben enthalten
  Als Kleinbuchstaben zählen die Zeichen a - z; *nicht* dazu gehören ä, ö, ü, ß.
- mindestens eine Ziffer enthalten.

Sonderzeichen (z.B. !$%&()=?+#*~-_.,;<>|@^{[]}), ä, Ä, ö, Ö, ü, Ü und ß sowie das Leerzeichen sind ebenfalls erlaubt. Ihre Verwendung erschwert das Erraten des Kennwortes.

Die Einhaltung dieser Anforderungen sowie die Änderung des Initialpasswortes werden systemseitig erzwungen.

Tipp: Ein "gutes" Passwort erhalten Sie in der Regel, wenn Sie die Anfangsbuchstaben der Wörter eines Satzes, den Sie sich leicht merken können, zu einem Passwort zusammensetzen.
Beispiel: "Meine(!) Patienten sind vom 1. Juli an kerngesund" wird zum Passwort "M(!)Psv1.Jak".

Beachten Sie zudem folgende Regelungen des BSI:

- Das Passwort muss regelmäßig gewechselt werden, z.B. alle 90 Tage.
- Ein Passwortwechsel ist durchzuführen, wenn das Passwort unautorisierten Personen bekannt geworden ist oder der Verdacht besteht.
- Alte Passwörter sollten nach einem Passwortwechsel nicht mehr verwendet werden.

9 REST-Ressourcen

9.1 REST API

Die folgende Tabelle enthält alle Ressourcen in Kombination mit den jeweiligen HTTP-Methoden. Die Spalte Operation enthält eine natürlichspracheliche Umschreibung als Link zur Detailspezifikation. Für den Request und den Response ist der erwartete bzw. gelieferte Content-Type des Bodys angegeben. Der Content-Type der Response gilt für den Erfolgsfall und kann für Fehler-Codes abweichen. Für die Autorisierung des Zugriffs auf Ressourcen werden drei Stufen unterschieden:

- keine: es ist keine Authentisierung nötig; sofern ein Authentication Header im Request vorhanden ist, wird er ignoriert.
- Authentisiert: der Benutzer muss authentisiert sein.
- Owner: Enthält die Ressource einen Bezug zu einem Account, muss dieser mit dem des des authentisierten Benutzers übereinstimmen

Die übergreifenden Aspekte sind zu beachten. In den Beispielen auf den Unterseiten werden bei den Requests die **HTTP-Header** nicht vollständig angegeben, um die Beispiele übersichtlicher zu halten.

ⓘ Sonderfall

Der Request-Header "Accept" kann ausschließlich mit der Funktion/Ressource "Erweiterte Account-Suche" sinnvoll benutzt werden. Nur dort kann zwischen XML und JSON als gewünschtes Response-Format differenziert werden. Bei allen anderen Ressourcen wird dieser Header bestenfalls ignoriert und kann in Einzelfällen zu Fehlern führen.

Kategorie	Operation	HTTP Method	Resource-URI	Requ Conte Type
Mail	Senden einer Mail	POST	`/mails`	text /plain
Mail	Abruf aller Mails	GET	`/accounts/<uid>/mails`	-
Mail	Abruf einer Mail	GET	`/accounts/<uid>/mails/<message-id>`	-
Mail	Löschen einer Mail	DELETE	`/accounts/<uid>/mails/<message-id>`	-
Header	Abruf aller Header	GET	`/accounts/<uid>/headers?from=<INT>&to=<INT>`	-

Account	Abruf von Account-Daten	GET	/accounts/<uid>	-
Account	Suchen von Accounts	GET	/accounts?search=<expression> /accounts?&login=<login>	-
VZD	Abruf des Adressbuchs	GET, HEAD	/vzd/accounts.xml.zip /vzd/accounts.json.zip	-
VZD	Erweiterte Account-Suche	GET	/vzd/search?(attribut=wert)(&attribut=wert)	-
Passwort	Passwort ändern	POST	/accounts/<uid>/password	text /plain
Login	Login-UID Mapping	GET	/login/<login>	-
Zertifikat	Abruf eines Zertifikats mittels UID	GET	/accounts/<uid>/certificate	-
Zertifikat	Abruf eines Zertifikats mittels UID oder Email-Adresse	GET, HEAD	/certificates?email=<email-adresse> /certificates?uid=<uid>	-
Zertifikat	Rückruf eines Zertifikats	DELETE	/accounts/<uid>/certificate	-
CSR	Senden eines CSR	POST	/csr	text /plain
CSR	CSR Status	GET	/csr/<csr-id>	-
Server	Version auslesen	GET	/server/version	-

9.2 Senden einer Mail

Diese Ressource dient dem Versenden von signierten und verschlüsselten E-Mails im S/MIME Format. Im Unterschied zu SMTP werden die Empfänger nicht außerhalb der Mail im Protokoll angegeben (es existiert ja auch kein Protokoll) sondern ausschließlich aus den Headern `TO` und `CC` bestimmt. BCC wird nicht unterstützt. Siehe auch Löschen einer Mail.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.2.1 Ressource

URL	<base-url>/mails
HTTP-Method	POST
Request Content-Type	text/plain;charset=UTF-8
Response Content-Type	text/plain oder application/xml
HTTP Status Codes	200, 400, 422
Berechtigung	Authentisiert

9.2.2 Request

Die Mail wird direkt als S/MIME ohne Modifikationen im Request-Body übertragen. Sie wird dabei nicht in eine XML-Struktur eingepackt, sondern so übertragen, wie sie im Client verschlüsselt erzeugt worden ist. Der Request-Body sieht beispielhaft so aus:

```
POST /mails HTTP/1.1

Date: Wed, 15 Aug 2018 17:28:05 +0200
From: "Test Nutzer" <Test.Nutzer.KVTG@kv-safenet.de>
To: Test.Empfaenger.KVTG@kv-safenet.de
Message-ID: <4DF77E05.3060604@kv-safenet.de>
Subject: Arztbrief-Eingangsbestaetigung
X-KVC-Dienstkennung: Arztbrief;Eingangsbestaetigung;V1.2
X-KVC-Sendersystem: MeinPVS;V1.0
MIME-Version: 1.0
Content-Type: application/pkcs7-mime; name=smime.p7m;
    smime-type=enveloped-data
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggNsMIIBsgIBADCBmTCBhjELMAkGA1UEBhMCREUxDzANBgNV
BAgTBkJlcmxpbjEPMA0GA1UEBxMGQmVybGluMQwwCgYDVQQKEwNLQlYxEzARBgNVBAsTCkRlemVy
...
ACSahpAssVXZl6jGDko1AU8oc9cSYGn35UmI4zUTwv/lEHXYVlWyGyzowuFHtnyHSqrubr/IyRJS
nXQKJ4oAAAAAAAAAAAAA
```

Man beachte, dass die HTTP-Header die gleiche Syntax wie die MIME-Header besitzen. Die Trennung erfolgt nur durch die Leerzeile.

Außer den für eine valide MIME-Mail erforderlichen Header-Felder müssen zusätzlich die folgenden Header enthalten sein:

Header	Beschreibung	Beispiel
X-KVC-Sendersystem	Bezeichnung des Clients	`IT-Doktor;V1.2`
X-KVC-Dienstkennung	Name der Anwendung bzw. des Dienstes gemäß der veröffentlichten Spezifikation	`Arztbrief;VHitG-Versand;V1.2`

Um die korrkete Darstellung beim Empfängers zu gewährleisten können Header-Werte gemäß RFC 2047 und RFC 6532 escaped werden.

9.2.3 Response

```
HTTP/1.1 200 OK

Mail erfolgreich gesendet
```

9.2.4 Fehler Response: Empfänger unbekannt

Wird zurückgegeben, wenn mindestens ein Empfänger keine gültige und existierende KV-Connect Adresse ist. Die Mail wird in diesem Fall an keinen der Empfänger ausgeliefert.

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<unknown_receivers>
  <unknown_receiver>Test.Nutzer.KVXY@kv-safenet.de</unknown_receiver>
</unknown_receivers>
```

9.2.5 Fehler Response: Mail enthält BCC-Empfänger

Wird zurückgegeben, wenn die Mail einen MIME-Header BCC mit mindestens einem Empänger enthält. (Ein leerer MIME-Header BCC ist erlaubt.)

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: text/plain

BCC wird nicht unterstuetzt
```

9.2.6 Fehler Response: Mailformat fehlerhaft

Wird zurückgegeben, wenn die übertragene Mail:

- ein fehlerhaftes MIME Format hat

- ein fehlerhaftes S/MIME Format hat
- mit einem nicht erlaubten Verschlüsselungsalgorithmus verschlüsselt ist
- erforderliche Header nicht vorhanden oder leer sind

Der Text enthält Hinweise auf den konkreten Fehler, z.B.:

```
HTTP/1.1 400 Bad Request

Mailformat fehlerhaft: X-KVC-Sendersystem nicht gesetzt
```

9.2.7 Fehler Response: Fehler beim Versenden

Wird zurückgegeben, wenn die übertragene Mail serverseitig nicht versendet werden konnte.

```
HTTP/1.1 400 Bad Request

Nachricht nicht vorhanden oder unvollstaendig
```

9.3 Abruf aller Mails

Über diese Ressource können alle Mails eines Benutzers vom Server abgerufen werden. Die UID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die UID immer im Klartext.

Mails werden hintereinander durch eine definierte Trennsequenz in den Response geschrieben. Diese ist

```
\r\n###--1122334455667788990O-###\r\n
```

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.3.1 Ressource

URL	<base_url>/accounts/<uid>/mails
HTTP-Method	GET
Request Content-Type	-
Response Content-Type	application/octet-stream
HTTP Status Codes	200
Berechtigung	Owner

Request

```
GET /accounts/301b78d4-e6e2-464e-9192-a14bec427360@00/mails HTTP/1.1
```

Response

```
HTTP/1.1 200 OK

Date: Wed, 15 Aug 2018 10:14:14 +0200
From: Test.Nutzer.KVTG@kv-safenet.de
To: "Test Empfaenger" <Test.Empfaenger.KVTG@kv-safenet.de>
Message-ID: <5a9a042a-5357-4758-b869-d998f803e55d@kv-safenet.de>
Subject: eNachricht
X-KVC-Dienstkennung: eNachricht;Lieferung;V2.1
X-KVC-Sendersystem: MeinPVS;V1.0
MIME-Version: 1.0
Content-Type: application/pkcs7-mime; name=smime.p7m;
    smime-type=enveloped-data
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggLuMIIBcwIBADBbME8xJDAiBgNVBAMMG0tWIFRlbGVtY
XRp
ayBBUkdFIFVzZXIgQ0EgMTEaMBgGA1UECgwRS1YgVGVsZW1hdGlrIEFSR0UxCzAJBgNVBAYTA
kRF
..
AghVKWONpcvE3TANBgkqhkiG9w0BAQEFAASCAQAYyNqELJksKGVytgbuGNEsvy9YzrWJVvp7t
8BB
2oCQ3JwQR3yhloFR1+9BZCQTryjWcHeJ/uLTBdsLbOVfXc2fud0K+drjzOP
/KnI5sz26Uwc8yaOq
###--11223344556677889900-###
Date: Wed, 15 Aug 2018 11:14:14 +0200
From: "Test Sender" <Test.Sender.KVTG@kv-safenet.de>
To: Test.Nutzer.KVTG@kv-safenet.de
Message-ID: <a6935101-2ea7-4569-aab9-4e722caf453e@system.de>
Subject: Processed: eNachricht
X-KVC-Dienstkennung: eNachricht;Lieferung;V2.1
X-KVC-Sendersystem: DeinPVS;V1.0
MIME-Version: 1.0
Content-Type: application/pkcs7-mime; name=smime.p7m;
    smime-type=enveloped-data
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggLuMIIBcwIBADBbME8xJDAiBgNVBAMMG0tWIFRlbGVtY
XRp
ayBBUkdFIFVzZXIgQ0EgMTEaMBgGA1UECgwRS1YgVGVsZW1hdGlrIEFSR0UxCzAJBgNVBAYTA
kRF
..
AghVKWONpcvE3TANBgkqhkiG9w0BAQEFAASCAQCYOGOn3jMxCV+Ug0MgmpxnEKPflOPlFlbyW
g7g
tjrccfDg5kfTwrVC024UPjUlPVjxbPJhlMN59hqVnh6iZ
/3e7rmpfWde9lUdpZmA2GpNbgNhCVnU
```

Response, wenn keine Mails vorhanden sind

In diesem Fall wird ein leerer Body zurückgeliefert.

```
HTTP/1.1 200 Ok
```

9.4 Abruf einer Mail

Ressource zum Abruf einzelner Mails vom Server. Über die eindeutige Message-ID einer Mail (siehe auch MIME) kann eine Mail adressiert und einzeln vom Server abgerufen werden. Die Message-ID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die Message-ID immer im Klartext.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.4.1 Ressource

URL	<base_url>/accounts/<uid>/mails/<message-id>
HTTP-Method	GET
Request Body	-
Response Body	application/octet-stream
HTTP Status Code	200
Berechtigung	Owner

Request

Abgerufen werden soll die Mail mit der Message-ID *<432674637.67676@kv-safenet.de>*.

```
GET /accounts/54345/mails/%3C432674637.67676%40kv-safenet.de%3E HTTP/1.1
```

Response

```
HTTP/1.1 200 OK

Date: Wed, 15 Aug 2018 11:14:14 +0200
From: "Test Sender" <Test.Sender.KVTG@kv-safenet.de>
To: Test.Nutzer.KVTG@kv-safenet.de
Message-ID: <4DF77E05.3060604@kv-safenet.de>
Subject: eNachricht
X-KVC-Dienstkennung: eNachricht;Lieferung;V2.1
X-KVC-Sendersystem: DeinPVS;V1.0
MIME-Version: 1.0
Content-Type: application/pkcs7-mime; name=smime.p7m;
    smime-type=enveloped-data
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggNsMIIBsgIBADCBmTCBhjELMAkGA1UEBhMCREUxDzANB
gNV
BAgTBkJlcmxpbjEPMA0GA1UEBxMGQmVybGluMQwwCgYDVQQKEwNLQlYxEzARBgNVBAsTCkRle
mVy
...
63EWKBjAmRRNUYXc1KiAoyxoUyrWyQnT+N4OGCtuTezhGEPKFx8GuG6hjkTB7F1nfgTaiwk6T
1fX
```

```
ACSahpAssVXZl6jGDko1AU8oc9cSYGn35UmI4zUTwv/lEHXYVlWyGyzowuFHtnyHSqrubr
/IyRJS
```

Fehler Response: Mail nicht vorhanden

```
HTTP/1.1 404 Not Found
```

9.5 Löschen einer Mail

Ressource zum Löschen einer Mail über die Message-ID, vgl. auch Senden einer Mail. Die Message-ID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die Message-ID immer im Klartext.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.5.1 Ressource

URL	<base_url>/accounts/<uid>/mails/<message-id>
HTTP-Method	DELETE
Request Content-Type	-
Response Content-Type	text/plain
HTTP Status Codes	200
Berechtigung	Owner

Request

```
DELETE /accounts/543543/mails/%3C463276473.4343@kv-safenet.de%3E HTTP/1.1
```

Response, falls Mail vorhanden und gelöscht wurde

```
HTTP/1.1 200 OK

Mail <463276473.4343@kv-safenet.de> gelöscht
```

9.6 Abruf aller Header

Ressource zum Abruf aller Mail Header vom Server. Es werden alle Header gruppiert nach den Mails aus denen Sie stammen zurückgegeben. Das Ergebnis ist ein XML-Dokument, dass die Gruppierung abbildet und in dem alle Header-Werte mit einem CDATA Tag eingeschlossen sind. Je nach URL können die gelieferten Header auch gefiltert werden, wobei die Header case-insensitiv interpretiert

werden. Mehrfach auftretende Header, genauer Header mit gleichem "Key", werden auch mehrfach zurückgegeben. Die Header sind dabei absteigend nach Alter sortiert, d.h. die Header der ältesten Mails stehen am Anfang. Die UID muss URL-enkodiert (RFC 3986) angegeben werden.

Die Query-Parameter *from* und *to* sind optional und erlauben Paging beim Abrufen der Header. *from* ist dabei der Index der ersten Mail (Zählung startet bei 1), deren Header geliefert werden, *to* der Index der letzten Mail. Der Default für *from* ist 1 und für *to* 4294967295. Für alle praktischen Zwecke werden beim Weglassen beider Parameter alle Header geliefert. Ist *from>to* oder einer der beiden Werte negativ oder größer als 4294967295 wird ein entsprechender Fehlerresponse erzeugt.

Empfehlung: Zur Massenverarbeitung von Mails bietet es sich an, immer die ersten X Header abzurufen (*to*=X), diese zu verarbeiten und anschließend zu löschen. Beim nächsten Abruf der ersten X Header werden dann andere Header geliefert. Insgesamt werden die Mails so in der Reihenfolge ihres Eintreffens verarbeitet.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.6.1 Ressource

URL für alle Header	<base_url>/accounts/<uid>/headers?from=<INT>&to=<INT>
URL für die Header Message-Id, From und Subject	<base_url>/accounts/<uid>/headers(short)? from=<INT>&to=<INT>
URL für ausgewählte Header	<base_url>/accounts/<uid>/headers(header1,header2,...)? from=<INT>&to=<INT>
HTTP-Method	GET
Request Content-Type	-
Response Content-Type	application/xml
HTTP Status Codes	200, 400
Berechtigung	Owner

9.6.2 Request 1 - Anfrage aller Mail Headers eines Accounts

```
GET /accounts/53212/headers HTTP/1.1
```

9.6.3 Response 1

```
HTTP/1.1 200 OK


<headers>
  <header>
    <message-id><![CDATA[<4DF77E05.3060604@kv-safenet.de>]]></message-id>
    <from><![CDATA[test.nutzer.kvtg@kv-safenet.de]]></from>
    <subject><![CDATA[Arztbrief]]></subject>
    <content-disposition><![CDATA[attachment; filename=smime.p7m]]><
/content-disposition>
```

```
    <x-kvc-dienstkennung><![CDATA[Arztbrief;VHitG-Versand;V1.2]]></x-kvc-
dienstkennung>
    <x-kvc-sendersystem><![CDATA[Mein-PVS;V1.2]]></x-kvc-sendersystem>
    <date><![CDATA[Wed, 15 Aug 2018 11:14:14 +0200]]></date>
    <mime-version><![CDATA[1.0]]></mime-version>
    <to><![CDATA[Test.Empfaenger.KVTG@kv-safenet.de]]></to>
    ...
  </header>
  <header>
     <message-id><![CDATA[<4DF77E05.3434343@kv-safenet.de>]]></message-
id>
    ...
  </header>
  ...
<headers>
```

9.6.4 Request 2 - Abruf der Header From, Message-ID und Subject

```
GET /accounts/53212/headers(short) HTTP/1.1
```

9.6.5 Response 2

```
HTTP/1.1 200 OK


<headers>
  <header>
    <message-id><![CDATA[<4DF77E05.3060604@kv-safenet.de>]]></message-id>
    <from><![CDATA[test.nutzer.kvtg@kv-safenet.de]]></from>
    <subject><![CDATA[Arztbrief]]></subject>
  </header>
  <header>
     <message-id><![CDATA[4DF77E05.3434343@kv-safenet.de]]></message-id>
     <from><![CDATA[test.nutzer.kvtg@kv-safenet.de]]></from>
     <subject><![CDATA[eNachricht]]></subject>
  </header>
  ...
<headers>
```

9.6.6 Request 3 - Abruf spezieller Header, hier Subject

```
GET /accounts/53212/headers(Subject) HTTP/1.1
```

9.6.7 Response 3

```
HTTP/1.1 200 OK


<headers>
  <header>
    <message-id><![CDATA[<4DF77E05.3060604@kv-safenet.de>]]></message-
id>
```

```
      <subject><![CDATA[Arztbrief]]></subject>
    </header>
    <header>
       <message-id><![CDATA[<4DF77E05.3434343@kv-safenet.de>]]></message-
 id>
      <subject><![CDATA[eNachricht]]></subject>
    </header>
    ...
 <headers>
```

9.6.8 Request 4 : Unrecognised Header

Es kann vorkommen, daß es Headerfelder gibt, die sich nicht als XML-Tag abbilden lassen. In diesen Fällen wird das XML-Tag "x-unrecognised-#" generiert, wobei der "defekte" Mail-Header im CDATA-Format als Inhalt des Tags abgelegt wird.

```
GET /accounts/53212/headers HTTP/1.1
```

9.6.9 Response 4

```
HTTP/1.1 200 OK


<headers>
  <header>
    <message-id><![CDATA[<4DF77E05.3060604@kv-safenet.de>]]></message-id>
    <from><![CDATA[test.nutzer.kvtg@kv-safenet.de]]></from>
    <subject><![CDATA[Arztbrief]]></subject>
    <content-disposition><![CDATA[attachment; filename=smime.p7m]]><
/content-disposition>
    <x-kvc-dienstkennung><![CDATA[Arztbrief;VHitG-Versand;V1.2]]></x-kvc-
dienstkennung>
    <x-kvc-sendersystem><![CDATA[Mein-PVS;V1.2]]></x-kvc-sendersystem>
    <date><![CDATA[Wed, 15 Aug 2018 11:14:14 +0200]]></date>
    <mime-version><![CDATA[1.0]]></mime-version>
    <to><![CDATA[Test.Empfaenger.KVTG@kv-safenet.de]]></to>
    <x-unrecognised-1><![CDATA[name="smime.p7m":name="smime.p7m"]]></x-
unrecognised-1>
    <x-unrecognised-2><![CDATA[y><df:dfdf]]></x-unrecognised-2>
    ...
  </header>
  <header>
     <message-id><![CDATA[<4DF77E05.3434343@kv-safenet.de>]]></message-
id>
    ...
  </header>
  ...
<headers>
```

9.6.10 Response, falls keine Mails vorhanden

Sind keine Mails vorhanden, wird ein XML-Dokument mit einem leeren <headers>-Tag zurückgegeben.

```
HTTP/1.1 200 OK
```

```
<headers />
```

9.6.11 Request 5 mit ungültigen Query-Parameter

```
GET /accounts/53212/headers&from=100&to=5 HTTP/1.1
```

9.6.12 Response 5

```
HTTP/1.1 400 Bad Request

from/to ungültig
```

9.7 Abruf von Account-Daten

Die Kontodaten sind nicht auf dem Server persistent, können aber über das REST-Interface zu einer UID abgerufen werden. Die UID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die UID immer im Klartext. Der Server gibt die folgenden Daten zurück:

- UID
- E-Mail-Adresse
- Zeitpunkt der letzten Passwortänderung
- Notwendigkeit der Passwortänderung

Die UID kann über die Login-UID Mapping Ressource bestimmt werden.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.7.1 Ressource

URL	<base_url>/accounts/<uid>
HTTP Method	GET
Request Body	-
Response Body	application/xml
HTTP Status Code	200, 403
Berechtigung	Owner

9.7.2 Request

```
GET /accounts/5beb3286-f797-40a2-b584-4890e557333c@00 HTTP/1.1
```

9.7.3 Response

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <account uid="5beb3286-f797-40a2-b584-4890e557333c@77">
  <email>test.nutzer@kv-safenet.de</email>
  <passwordLastChange>2017-11-11T11:11:11+02:00</passwordLastChange>
  <passwordChangeNeeded>false</passwordChangeNeeded>
</account>
```

9.7.4 Fehler Response: keine Authentisierung oder falscher Account abgefragt

```
HTTP/1.1 403 Forbidden
```

9.8 Suchen von Accounts

Über diese Ressource können Benutzerdaten gesucht werden. Mögliche Filter sind:

- search: Account-Email muss im localpart (links vom @) den Suchausdruck enthalten ("Unscharfe Suche")
- login: Account-Email muss im localpart (links vom @) mit Suchausdruck übereinstimmen ("Scharfe Suche")

⚠ Die Filter *search* und *login* schliessen sich gegenseitig aus, einer von beiden muss aber verwendet werden.

ℹ Alle Filter sind case-insensitiv.

Der Server gibt für die zugehörigen Accounts die folgenden Daten zurück:

- UID
- EMail-Adresse

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.8.1 Ressource

URL	<base_url>/accounts/?search=<expression> <base_url>/accounts/?login=<login>
HTTP Method	GET
Request Body	-
Response Body	application/xml
HTTP Status Code	200, 400, 404
Berechtigung	Authentisiert

9.8.2 Request - Unscharfe Suche

```
GET /accounts/?search=hans HTTP/1.1
```

9.8.3 Response - Unscharfe Suche

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<accounts>
  <account uid="5beb3286-f797-40a2-b584-4890e557333c@00">
    <email>hans.tester.KVTG@kv-safenet.de</email>
 </account>
  <account uid="5beb3286-f797-40a2-b584-4890e557333c@99">
    <email>hans.nutzer.KVXY@kv-safenet.de</email>
  </account>
</accounts>
```

9.8.4 Request - Scharfe Suche

```
GET /accounts/?login=hans.tester HTTP/1.1
```

9.8.5 Response - Scharfe Suche

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<accounts>
  <account uid="5beb3286-f797-40a2-b584-4890e557333c@00">
    <email>hans.tester.KVTG@kv-safenet.de</email>
 </account>
</accounts>
```

9.8.6 Response, falls kein Account zu den Suchkriterien gefunden wird

Hier wird ein 404 Not found mit zurückgegeben.

```
HTTP/1.1 404 Not found

No accounts found while searching for <search-expresion>
```

9.8.7 Response, falls Filter falsch angegeben wurden

Hier wird ein 400 Bad request zurückgegeben.

```
HTTP/1.1 400 BAD REQUEST

Either the 'search' parameter or the 'login' parameter have to be set!
```

9.9 Abruf des Adressbuches

Über diese Ressource kann das gesamte Adressbuch abgerufen werden. Diese Ressource wird regelmäßig (mindestens täglich) neu erzeugt, entspricht aber nicht sekundengenau dem aktuellen Stand.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.9.1 Ressource

URL	<base_url>/vzd/accounts.xml.zip <base_url>/vzd/accounts.json.zip
HTTP Method	GET, HEAD
Request Body	-
Response Body	application/zip
HTTP Status Code	200, 304
Berechtigung	-

9.9.2 Request - Plain

```
GET /vzd/accounts.xml.zip HTTP/1.1
```

9.9.3 Response

```
HTTP/1.1 200 OK
Last-Modified: Thu, 22 Sep 2016 12:45:26 GMT

<binary data>
```

Der Response ist für beide Ressourcen ein ZIP-File, dass genau eine Datei und keine Unterordner enthält. Die Datei ist entweder im XML- oder im JSON-Format gemäß Abschnitt Attribute und Dateiformate. Es sind alle Accounts enthalten, die ein gültiges Zertifikat besitzen.

Im Response ist der HTTP-Header *Last-Modified* gesetzt, der angibt, wann die Ressource das letzte mal aktualisiert wurde. Dies kann benutzt werden, um nicht unnötig ein lokal bereits bekannten Stand herunterzuladen.

9.9.4 Request - Nur Header

```
HEAD /vzd/accounts.xml.zip HTTP/1.1
```

9.9.5 Response

```
HTTP/1.1 200 OK
Last-Modified: Thu, 22 Sep 2016 12:45:26 GMT
```

Es werden nur die Header im Response übertragen. Der Zeitpunkt der letzten Aktualisierung kann so in Erfahrung gebracht werden. Existiert eine neue Version, kann diese anschließend mit einem GET-Request runtergeladen werden.

9.9.6 Request - If-modified-since

```
GET /vzd/accounts.xml.zip HTTP/1.1
If-Modified-Since: Thu, 22 Sep 2016 12:45:26 GMT
```

9.9.7 Response

```
HTTP/1.1 200 OK
Last-Modified: Fri, 23 Sep 2016 12:45:11 GMT

<binary data>
```

oder

```
HTTP/1.1 304 Not modified
Last-Modified: Fri, 22 Sep 2016 12:45:26 GMT
```

Wird im Request der HTTP-Header *If-modified-since* gesetzt, wird die Ressource nur übertragen, falls sie zu einem späteren Zeitpunkt aktualisiert wurde. Hier kann direkt der Wert des Response-Headers *Last-Modified* des letzten erfolgreichen Requests benutzt werden. Liegt keine aktuellere Version vor, wird *304 Not modified* zurückgegeben.

9.9.8 Attribute und Dateiformate

siehe: Attribute und Dateiformate

9.10 Erweiterte Account-Suche

Über diese Ressource kann gezielt nach Accounts gesucht werden. Im Gegensatz zum Abruf des gesamten Adressbuches sind die Ergebnisse aktuell.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.10.1 Ressource

URL	<base_url>/vzd/search?(attribut=wert)(&attribut=wert)*
HTTP Method	GET
Request-Header Accept	application/xml, application/json
Request Body	-
Response Body	application/xml, application/json
HTTP Status Code	200, 400
Berechtigung	-

Der Request muss mindestens einen URL-Query-Paramter enthalten. Als Such-Paramter können alle Felder aus der Spalte **Attribut** im Abschnitt Attribute und Dateiformate in Abruf des Adressbuches durchsucht werden. Der Wert eines Attributes kann den Platzhalter * enthalten, der beliebig viele Zeichen ersetzt. Ein Wert ohne Platzhalter kann mit ~ beginnen, um eine phonetische Suche gemäß Kölner Phonetik durchzuführen. Eine Kombination von Platzhalter * und phonetischer Suche ~ ist nicht erlaubt.

⚠ Es ist wichtig, die Platzhalter in der URL passend kodiert zu übermitteln.

Es gelten die folgenden Besonderheiten:

- Die folgenden Felder können nicht durchsucht werden:
  - titel
  - certificate
- Eine phonetische Suche ist für die folgenden Attribute möglich:
  - vorname
  - nachname
- Der Parameter für *dienstkennungen* heißt *dienstkennung.* Sein Wert wird mit jeder einzelnen Dienstkennung verglichen
- Der Parameter für *fachgruppen* heißt *fachgruppe.* Sein Wert wird mit jeder einzelnen Fachgruppe verglichen
- Unbekannte Query-Parameter werden ignoriert
- Die vorhandenen Parameter werden mit logischem UND verknüpft
- Das Verhalten bei mehrfach vorkommenden Query-Parametern ist nicht definiert
- Es findet keine Überprüfung auf sich widersprechende Werte statt (z.B. plz passend zur Stadt), das Ergebnis ist dann ggf. leer.
- Es werden maximal 100 Ergebnisse zurückgeliefert. Das Vorhandensein weiterer Ergebnisse wird nicht angezeigt.

Der Response wird in Abhängigkeit vom HTTP-Header *Accept* im Request ein XML- oder JSON-Dokument sein, ersteres ist der Default. Wird mindestens ein aber kein richtiger Accept Header gesetzt, wird immer 406 NOT ACCEPTABLE geantwortet.

Im Gegensatz zu Abruf des gesamten Adressbuches, wird das Ergebnis nicht als ZIP-Datei übertragen.

9.10.2 Request

```
GET /vzd/search?mail=test.nutzer* HTTP/1.1
```

9.10.3 Response

```
HTTP/1.1 200 OK

{
    "created": "2018-08-15 13:25:05",
    "accounts": [
        {
            "id": "81286d4d-b13b-49ae-b6c7-beb6b4c23680@99",
            "mandant": "KVTG",
            "titel": "",
            "vorname": "Test",
            "nachname": "Nutzer",
            "arzt": false,
            "lanr": "",
            "fachgruppen": [],
            "ou": "KV Telematik GmbH",
            "bsnr": "",
            "iknr": "",
            "strasse": "Herbert-Lewin-Platz 2",
            "plz": "10623",
            "stadt": "Berlin",
            "mail": "Test.Nutzer.KVTG@kv-safenet.de",
            "dienstkennungen": [],
            "certificate": "https://kvc-1.kvtg.kbv.de:8443/kvconnect/rest
/accounts/81286d4d-b13b-49ae-b6c7-beb6b4c23680@99/certificate"
        }
    ]
}
```

9.10.4 Response, falls kein Filter angegeben wurde

```
HTTP/1.1 400 Bad Request
```

9.10.5 Beispiele

(Zur besseren Lesbarkeit ohne URL-Enkodierung)

Request	Response	Bemerkung
/vzd/search?stadt=Hamburg	200 OK	Alle Accounts in Hamburg
/vzd/search?stadt=Hamburg&dienstkennung=eArztbrief;*;V1.0	200 OK	Alle Accounts in Hamburg,

Request	Response	Bemerkung
		die Arztbriefe versendet haben
/vzd/search?stadt=Hamburg&nachname=~Meier	200 OK	Alle Accounts in Hamburg für Meier, Meyer, Mair etc.
/vzd/search	400 Bad request	Kein Parameter angegeben
/vzd/search?stadt=~Hamburg	400 Bad request	Keine phonetische Suche für stadt möglich
/vzd/search?stadt=Hamburg&strasse=Konstantin-Strasse	200 Ok	Leere Account-Liste, keine Konstantin-Strasse in Hamburg

9.10.6 Attribute und Dateiformate

siehe: Attribute und Dateiformate

9.11 Passwort ändern

Über diese Ressource kann das Passwort geändert werden. Die UID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die UID immer im Klartext.

⚠ Der Content-Type: text/plain;charset=UTF-8 ist zu verwenden.

⚠ Die Passwortrichtlinien sind beim Setzen der Passworts zu beachten.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.11.1 Ressource

URL	<base_url>/accounts/<uid>/password
HTTP-Method	POST
Request Body	text/plain;charset=UTF-8
Response Body	text/plain
HTTP Status Code	201, 400, 422
Berechtigung	Owner

9.11.2 Request

```
POST /accounts/cc7a603f-1499-4ee3-a844-082511bd6d58@75/password HTTP/1.1
Content-Type: text/plain;charset=UTF-8

?Neues#1Passwort+2Gemäß*3Richtlinie!
```

9.11.3 Response

```
HTTP/1.1 201 Created


Changed password successfully
```

9.11.4 Response bei leerem Passwort

```
HTTP/1.1 400 Bad Request

Unvollstaendige Eingabe: Das neue Passwort darf nicht leer sein.
```

9.11.5 Response bei ungültigem Passwort

```
HTTP/1.1 422 Unprocessable Entity

Das neue Passwort entspricht nicht den Passwortrichtlinien.
```

9.12 Login-UID Mapping

Diese Ressource gibt die UID für das übergebene Login zurück, sofern das Login zum angemeldeten User gehört.

9.12.1 Ressource

URL	<base_url>/login/<login>
HTTP-Method	GET
Request Body	-
Response Body	text/plain
HTTP Status Code	303, 404
Berechtigung	Authentisiert

9.12.2 Request

```
GET /rest/login/test.nutzer.KVTG HTTP/1.1
```

9.12.3 Response

Der Server antwortet im Erfolgsfall mit einem Redirect zu der entsprechenden Ressource mit den Account-Details, siehe Abruf von Account-Daten. Die entsprechende URL wird im Location Header übermittelt.

```
HTTP/1.1 303 See Other
Location: <base-url>/rest/accounts/aeaae8ce-5616-4468-b89a-
552ed2e8b64b@00
```

9.12.4 Response, falls Login unbekannt

```
HTTP/1.1 404 Not Found
```

9.13 Abruf eines Zertifikats mittels UID

Die Ressource repräsentiert das aktuelle Zertifikat für einen Account. Die UID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die UID immer im Klartext.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.13.1 Ressource

URL	<base_url>/accounts/<uid>/certificate
HTTP-Method	GET
Request Body	-
Response Body	application/xml
HTTP Status Code	200, 404
Berechtigung	Keine

9.13.2 Request

```
GET /accounts/9811964c-4106-4367-aa7b-781480376e8c@99/certificate HTTP/1.
1
```

9.13.3 Response

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<certificate>
  <id>9811964c-4106-4367-aa7b-781480376e8c@99</id>
  <email>Test.Nutzer.KVTG@kv-safenet.de</email>
  <validFrom>Wed Aug 15 15:30:21 GMT 2018</validFrom>
  <validTo>Sun Aug 15 15:30:21 GMT 2021</validTo>
  <body>-----BEGIN CERTIFICATE-----&#xD;
MIIE9TCCAt2gAwIBAgIIRWPxUPqe7uYwDQYJKoZIhvcNAQENBQAwTzEkMCIGA1UE&#xD;
AwwbS1YgVGVsZW1hdGlrIEdtYkggVXNlciBDQSAxMRowGAYDVQQKDBFLViBUZWxl&#xD;
....
6qKxtEYoYi9t77OlxBJq39Od9szR6FX4Qi7ycNTRAoZdD53QHu2cJRmSqgXUvXkv&#xD;
RzXY8sN5c5Zbv8ZvWO640JSC3L8N88D3ug==&#xD;
-----END CERTIFICATE-----&#xD;
</body>
</certificate>
```

9.13.4 Fehler Response: kein Zertifikat vorhanden

```
HTTP/1.1 404 Not Found
```

9.14 Zertifikat holen

Die Ressource repräsentiert das aktuelle Zertifikat für einen Account. Neben GET ist auch HEAD definiert. Die Email-Adresse bzw. UID müssen URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheinen die Email-Adresse bzw. UID immer im Klartext.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.14.1 Ressource

URL	<base-url>/certificates?email=<email-adresse> <base-url>/certificates?uid=<uid>
HTTP-Method	GET, HEAD
Request Body	-
Response Content-Type	application/xml
HTTP Status Codes	200, 404

9.14.2 Request

```
GET /certificates?email=petra.pan.kvtg%40kv-safenet.de HTTP/1.1
```

```
GET /certificates?uid=bbf41219-6680-4c29-b232-f770d9120a98%4099 HTTP/1.1
```

9.14.3 Response im Erfolgsfall

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<certificate>
  <id>bbf41219-6680-4c29-b232-f770d9120a98@99</id>
  <email>Test.Nutzer.KVTG@kv-safenet.de</email>
  <validFrom>Thu Feb 15 15:30:21 CET 2018</validFrom>
  <validTo>Mon Feb 15 15:30:21 CET 2021</validTo>
  <body>-----BEGIN CERTIFICATE-----&#xD;
MIIE9TCCAt2gAwIBAgIIRWPxUPqe7uYwDQYJKoZIhvcNAQENBQAwTzEkMCIGA1cv&#xD;
AwwbS1YgVGVsZW1hdGlrIEdtYkggVXNlciBDQSAxMRowGAYDVQQKDBFLViBUZWdf&#xD;
...
6qKxtEYoYi9t77OlxBJq39Od9szR6FX4Qi7ycNTRAoZdD53QHu2cJRmSqgXUvXeR&#xD;
RzXY8sN5c5Zbv8ZvWO640JSC3L8N88D3ug==&#xD;
-----END CERTIFICATE-----&#xD;
</body>
</certificate>
```

9.14.4 Response im Fehlerfall

```
HTTP/1.1 404 Not Found
```

9.15 Rückruf eines Zertifikats

Ein Zertifikat kann vom Eigentümer zurück gerufen werden. Der Server entfernt das Zertifikat aus dem Verzeichnisdient und invalidiert es in der PKI. Die UID muss URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheint die UID immer im Klartext.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.15.1 Ressource

URL	<base_url>/accounts/<uid>/certificate
HTTP-Method	DELETE
Request Body	text/plain
Response Body	-
HTTP Status Code	204
Berechtigung	Owner

Da es nur ein Zertifikat je Account geben darf, ist die Seriennummer des Zertifikats nicht relevant.

9.15.2 Request

```
DELETE /accounts/345/certificate HTTP/1.1

<Grund für den Rückruf (optional)>
```

9.15.3 Response

```
HTTP/1.1 204 No Content
```

9.16 Senden eines Certificate Signing Request (CSR)

Um ein Zertifikat zu erstellen und in den Verzeichnisdienst zu übertragen sendet der Client über die folgende Ressource einen CSR an den Server. Dieser sendet den CSR zusammen mit den Stammdaten weiter an die PKI, empfängt das fertige Zertifikat, speichert dieses und gibt einen Redirect auf dessen Status zurück. Das fertige Zertifikat kann mittels Abruf eines Zertifikats mittels UID abgerufen werden.

Wenn für ein Konto ein CSR gesendet wird, obwohl bereits ein gültiges Zertifikat vorliegt, wird der Request dennoch bearbeitet. Das bisherige Zertifikat wird dann automatisch ungültig und durch das neue ersetzt. Sollte noch ein vorheriger CSR in Bearbeitung sein, wird dieser beendet.

Der CSR wird im PKCS10/PEM-Format erwartet, vgl. Kryptographische Standards. Dieser darf keine Metadaten außer der UID enthalten; die Metadaten werden vom Server ergänzt.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.16.1 Ressource

URL	<base_url>/csr
HTTP-Method	POST
Request Body	text/plain
Response Body	text/plain
HTTP Status Code	201
Berechtigung	Owner

9.16.2 Aufbau Certificate Signing Request (CSR)

```
$ openssl req -noout -text -in csr.pem

Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: x500UniqueIdentifier=TesT.Nutzer.KVTG
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:bc:b5:a8:c3:9f:62:1d:68:ba:74:dc:7f:48:c4:
                    d6:2b:52:88:23:53:6e:96:80:97:55:01:d3:b9:d5:
                    76:87:99:06:57:0c:7e:1c:e9:4a:34:fd:a6:e3:ec:
                    94:03:fc:a5:b9:b6:1c:dd:e2:2f:9c:e0:d0:3f:86:
                    7d:7f:6b:89:c6:37:9d:1d:50:23:e3:0a:a6:68:38:
                    9c:91:b8:16:ab:96:51:f0:98:9d:aa:fb:db:d0:9b:
                    af:2e:f8:e6:fe:bd:c1:bc:f8:e4:2c:a3:d5:af:a1:
                    1f:04:89:a9:6f:36:70:26:ce:43:46:f8:a5:e0:7b:
                    be:6d:33:84:5e:a9:12:82:ca:55:1a:91:6f:72:4a:
                    6f:62:44:4e:1c:3e:a8:e5:b8:d9:49:29:55:76:49:
                    cd:e6:ed:87:52:e8:ed:15:bd:91:e6:20:59:4b:fc:
                    ab:b7:d1:d2:c0:15:29:5a:e8:cc:49:51:3e:79:53:
                    bd:17:66:68:a7:da:aa:2c:75:f7:bf:7d:78:2e:3c:
                    d1:8b:15:b0:b5:9e:7b:00:32:b4:29:5d:85:2a:3b:
                    0e:04:b6:04:48:ae:df:7d:f3:28:f7:c3:34:50:a6:
                    75:c1:fc:be:cd:45:11:73:41:b1:8c:a1:c4:d9:d0:
                    dd:a8:4c:e7:b2:2c:9d:bf:d3:93:8e:e8:cd:60:d9:
                    8e:eb
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: sha256WithRSAEncryption
        9f:fd:e8:d4:77:bc:69:46:47:27:d3:20:4a:bc:22:86:2a:22:
        7e:82:ef:cd:12:1c:f7:4a:96:2e:f6:90:dd:41:09:fc:78:9b:
        38:6a:ce:ca:30:ef:ff:9f:9a:50:d6:ee:79:b2:cd:61:8d:d3:
        b4:c8:4e:1f:95:7c:8e:6a:d7:e9:3e:dd:9c:51:b9:98:3d:99:
```

```
        7a:36:2b:59:5b:ae:87:89:88:1d:ed:ce:28:a0:cb:0f:1e:ef:
        c4:b5:24:e4:c3:59:3c:21:5d:6f:00:1e:27:74:2d:f6:95:12:
        0d:2e:3e:39:f5:c1:80:2a:dd:bb:e4:9e:73:52:76:85:79:9b:
        5f:d2:da:3c:74:ed:39:a7:ea:40:aa:95:3b:0d:30:4c:37:0a:
        36:83:0d:e6:40:e7:1e:40:b6:35:e6:4e:66:21:2b:c0:64:a5:
        62:81:3b:3b:40:d3:67:1a:56:0e:0c:50:17:84:e2:1a:93:ef:
        c6:19:d7:db:41:ce:1e:eb:3a:5d:f8:94:09:7c:18:07:a4:c1:
        9f:74:b2:a8:a6:62:01:ce:11:43:d5:47:78:ee:d2:75:3a:96:
        74:a4:e3:a5:1e:e5:e9:ff:4d:96:6b:de:9a:a9:a5:3e:fe:cd:
        9a:99:12:20:40:74:c6:45:71:2e:9c:52:91:1f:3c:fe:33:00:
        d9:49:48:29
```

9.16.3 Request

```
POST /csr HTTP/1.1

-----BEGIN CERTIFICATE REQUEST-----
MIICYzCCAUsCAQAwHjEcMBoGA1UELQMTAG1hcmsuc2NoYWVmZXIua3Z3bDCCASIw
DQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALy1qMOfYh1ounTcf0jE1itSiCNT
...
GlYODFAXhOIak+/GGdfbQc4e6zpd+JQJfBgHpMGfdLKopmIBzhFD1Ud47tJ1OpZ0
pOOlHuXp/02Wa96aqaU+/s2amRIgQHTGRXEunFKRHzz+MwDZSUgp
-----END CERTIFICATE REQUEST-----
```

9.16.4 Response

Der Server antwortet mit 201 Created, die URL zum CSR Status wird im Location-Header und im Body übermittelt. Die <csr-id> ist eine rein interne Id des Servers und hat keinen Bezug zum Inhalt des CSRs.

```
HTTP/1.1 201 Created
Location: <baseurl>/csr/<csr-id>

CSR akzeptiert, siehe Status unter <baseurl>/csr/<csr-id>
```

9.17 CSR Status

Nach dem Senden eines Certificate Signing Request (CSR), kann es ggf. länger dauern, bis das Zertifikat vorliegt bzw. der CSR abgelehnt oder abgebrochen wurde. Die hier beschriebene Ressource erlaubt den Zugriff auf den Status der Bearbeitung. Der Status wird als XML-Dokument geliefert, die möglichen Statuscodes finden sich am Ende des Abschnitts.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.17.1 Ressource

URL	<base_url>/csr/<csr-id>
HTTP-Method	GET
Request Body	-
Response Body	application/xml
HTTP Status Code	200, 404
Berechtigung	Owner

9.17.2 Request

```
GET /csr/5454 HTTP/1.1
```

9.17.3 Responses

Der Response enthält eine Liste verschiedener definierter Status-Codes (siehe unten), die ggf. mit einer Klartext-Meldung versehen sind. Diese Liste ist nicht zwingend kumulativ, d.h. sie enthält ggf. nur den letzten gültigen Status.

CSR erfolgreich bearbeitet

```
HTTP/1.1 200 Ok

<?xml version="1.0" encoding="UTF-8"?>
<csr-status>
<account uid="5beb3286-f797-40a2-b584-4890e557333c@77"><base_url>
/accounts/5beb3286-f797-40a2-b584-4890e557333c@77</account>
<status-entries>
  <status-entry date="15.08.2018 14:25:23" status="999">Erfolgreich<
/status-entry>
</status-entries>
<csr-status>
```

9.17.4 Status-Codes

Status-Code	Bedeutung
100 - 199	***Annahme des Zertifikats***
100	CSR empfangen
110	Altes Zertifikat als Ressource gelöscht
120	Altes Zertifikat aus CA gelöscht
200 - 299	***Bearbeitung durch CA***
210	CSR an CA geleitet
299	Zertifikat von CA empfangen
300 - 399	***Verzeichnisdienst***
399	Zertifikat in Verzeichnisdienst veröffentlicht
900 - 999	***Abschluss-Codes***
900	Allgemeiner Fehler, z.B. CA nicht erreichbar
901	Durch neuen CSR abgebrochen
902	CSR abgelehnt
903	Falsches Subject
999	CSR erfolgreich bearbeitet

9.17.5 Fehler Response: CSR nicht vorhanden

```
HTTP/1.1 404 Not Found
```

9.18 Versionsnummer des Servers auslesen

Diese Ressource repräsentiert die Version des Servers.

Es gelten die übergreifenden Aspekte zur Benutzung der REST-Schnittstelle.

9.18.1 Ressource

URL	<base_url>/server/version
HTTP-Method	GET
Request Body	-
Response Body	text/plain
HTTP Status Code	200
Berechtigung	Keine

9.18.2 Request

```
GET /server/version HTTP/1.1
```

9.18.3 Response

```
HTTP/1.1 200 Ok

1.2.3
```

9.19 Attribute und Dateiformate

Die folgenden Attribute werden bei der Suche nach Accounts oder dem Abruf des gesamten Adressbuches immer zurückgegeben.
Sollte für ein Attribut kein Wert vorhanden sein, so wird ein dem Datentyp entsprechender Default zurückgegeben. (Boolean: false, String/Ziffer: leerer String, Liste: leere Liste)

Attribut	Datentyp	Beispiel	Bemerkung
id	String	009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@72	die eindeutige ID des Accounts
mandant	String	KVBW	Organisation, die diesen Account verwaltet
titel	String	Dr.	Persönliche Daten des Account-Inhabers
vorname	String	Karl	
nachname	String	Meier	

Attribut	Datentyp	Beispiel	Bemerkung
arzt	boolean	true	Spalte der Account Tabelle
lanr	Ziffern, 7-stellig, führenden Nullen	2222222	Spalte der Doctor Tabelle
fachgruppen	Liste von Fachgruppen als String (Code mit Klartext)	(012 HNO, 060 FA Hals-Nasen-Ohrenheilkunde)	Aktuell nur ein Fachgruppe - Spalte der Doctor Tabelle
ou	String	Praxis am Rathaus	Name der Organisationseinheit, Praxis, Kranknehaus etc. sowie zugehörige Stammdaten
bsnr	Ziffern, 9-stellig, führenden Nullen	514343511	
strasse (inkl. hausnummer)	String	Rathausplatz 1	
~~hausnummer~~	~~String~~	~~1~~	
plz	Ziffern, 5-stellig, führenden Nullen	34534	
stadt	Stadt	Stuttgart	
iknr	Ziffern, 9-stellig	1234567	Instituts-KennNummer (DB Person.IkNr)
mail	String	test.nutzer.kvtg@kv-safenet.de	KV-Connect-spezifische Daten
certificate	URL	https://kvlink1.kv-safenet.de:8443 /kvcserver/rest/account /009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@72/certificate	
dienstkennungen	Liste von Dienstkennungen	(Arztbrief;VHitG-Versand;V1.0, eNachricht;Lieferung;V2.0)	Enthält nur Dienstkennungen, die für andere Nutzer sichtbar sind

Hinweis: Das Attribut "hausnummer" wurde entfernt und wird nicht separat ausgeliefert. Die Hausnummer ist in der Straße enthalten

Das Adressbuch oder eine Liste von Suchergebnissen kann als XML- oder JSON-File abgerufen werden.

```
Liste von Accounts als XML

<?xml version="1.0" encoding="UTF-8"?>
```

```
<accounts date="2017-10-02 11:13:00">
    <account>
        <id>009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@00</id>
        <mandant>KVTG</mandant>
        <titel>Dr.</titel>
        <vorname>Karl</vorname>
        <nachname>Tester</nachname>
        <lanr>2222222</lanr>
        <bsnr>123456789</bsnr>
        <arzt>true</arzt>
        <fachgruppen>
            <fachgruppe>012 HNO</fachgruppe>
            <fachgruppe>060 FA Hals-Nasen-Ohrenheilkunde</fachgruppe>
        </fachgruppen>
        <dienstkennungen>
            <dienstkennung>Arztbrief;VHitG-Versand;V1.2</dienstkennung>
            <dienstkennung>eNachricht;Lieferung;V2.1</dienstkennung>
        </dienstkennungen>
        <iknr>1234567</iknr>
        <ou>Praxis am Rathaus</ou>
        <strasse>Rathausplatz 1</strasse>
        <plz>53442</plz>
        <stadt>Stuttgart</stadt>
        <mail>karl.tester.kvtg@kv-safenet.de</mail>
        <certificate>https://kvlink1.kv-safenet.de:8443/kvcserver/rest
/account/009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@00/certificate
        </certificate>
    </account>
    <account>
        <id>111d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@99</id>
        <mandant>KVXY</mandant>
        <titel>Dr.</titel>
        <vorname>Gernhart</vorname>
        <nachname>Muster</nachname>
        <lanr>3333333</lanr>
        <bsnr>123456789</bsnr>
        <arzt>true</arzt>
        <fachgruppen>
            <fachgruppe>012 HNO</fachgruppe>
            <fachgruppe>060 FA Hals-Nasen-Ohrenheilkunde</fachgruppe>
        </fachgruppen>
        <dienstkennungen>
            <dienstkennung>Arztbrief;VHitG-Versand;V1.0</dienstkennung>
            <dienstkennung>eNachricht;Lieferung;V2.1</dienstkennung>
        </dienstkennungen>
        <iknr>1234567</iknr>
        <ou>Praxis am Rathaus</ou>
        <strasse>Rathausplatz 1</strasse>
        <plz>53442</plz>
        <stadt>Stuttgart</stadt>
        <mail>gernhart.muster.kvxy@kv-safenet.de</mail>
        <certificate>https://kvlink1.kv-safenet.de:8443/kvcserver/rest
/account/111d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@99/certificate
        </certificate>
    </account>
</accounts>
```

Liste von Accounts als JSON

```
{
    "accounts": {
```

```
        "created": "2017-10-02 11:13:00",
        "accounts": [
            {
                "id": "009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@72",
                "mandant": "KVTG",
                "titel": "Dr.",
                "vorname": "Karl",
                "nachname": "Tester",
                "lanr": "2222222",
                "arzt": true,
                "fachgruppen": [
                    "012 HNO",
                    "060 FA Hals-Nasen-Ohrenheilkunde"
                ],
                "ou": "Praxis am Rathaus",
                "bsnr": "123456789",
                "iknr": "1234567",
                "strasse": "Am Rathaus 1",
                "plz": "53442",
                "stadt": "Stuttgart",
                "mail": "karl.tester.kvtg@kv-safenet.de",
                "certificate": "https://kvlink1.kv-safenet.de:8443
/kvcserver/rest/account/009d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@00
/certificate",
                "dienstkennungen": [
                    "Arztbrief;VHitG-Versand;V1.0",
                    "eNachricht;Lieferung;V2.0"
                ]
            },
            {
                "id": "111d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@99",
                "mandant": "KVXY",
                "titel": "Dr.",
                "vorname": "Gernhart",
                "nachname": "Muster",
                "lanr": "3333333",
                "arzt": true,
                "fachgruppen": [
                    "012 HNO",
                    "060 FA Hals-Nasen-Ohrenheilkunde"
                ],
                "ou": "Praxis am Rathaus",
                "bsnr": "123456789",
                "iknr": "1234567",
                "strasse": "Am Rathaus 1",
                "plz": "53442",
                "stadt": "Stuttgart",
                "mail": "gernhart.muster.kvxy@kv-safenet.de",
                "certificate": "https://kvlink1.kv-safenet.de:8443
/kvcserver/rest/account/111d28a0-82f5-4bd8-a0fc-a8db1c1eac5f@99
/certificate",
                "dienstkennungen": [
                    "Arztbrief;VHitG-Versand;V1.0",
                    "eNachricht;Lieferung;V2.1"
                ]
            }
        ]
    }
}
```

10 Anbindung mobiler Geräte

10.1 Einleitung

In diesem Teil wird definiert, wie KV-Connect Nachrichten zwischen KV-Connect Servern und Servern externer Anbieter ausgetauscht werden. Der Austausch der Daten zwischen KV-Connect Server und Anbieter erfolgt über einen Mailer-Account auf Seiten der KVTG und einen KV-Connect Mailer-Account des Anbieters in Form von hier spezifizierten S/MIME Mails. Außerdem gibt es eine Schnittstelle zum Abrufen der Zertifikate der Ärzte in der KV-Connect Benutzerverwaltung und eine Schnittstelle zum Abrufen der Zertifikate der externen Nutzer vom externen Anbieter.

10.2 Inhalte der KV-Connect Nachrichten

Als Inhalt einer zu empfangenden und zur Freigabe weiterzuleitenden Nachricht ist definiert:

- Arztbrief V1.2 mit Unterstützung V1.1 (siehe Spezifikation eArztbrief) inkl. MDN

Zusätzlich ist für den Empfang folgender Nachrichtentyp definiert. Nachrichten dieses Typs können für die Initiierung der Kommunikation mit externen Nutzern durch einen Arzt verwendet werden und dürfen nicht zur Freigabe weitergeleitet werden.

- eNachricht V2.1 (siehe Spezifikation eNachricht) inkl. MDN

10.3 REST Ressourcen

Die folgende Tabellen enthält alle Ressourcen in Kombination mit den jeweiligen HTTP-Methoden. Die Spalte Operation enthält eine natürlichsprachliche Umschreibung als Link zur Detailspezifikation. Für den Request und den Response ist der erwartete bzw. gelieferte Content-Type angegeben des Bodys angegeben. In jedem Fall sind die übergreifenden Aspekte zu beachten. In den Beispielen auf den Unterseiten werden bei den Requests die **HTTP-Header** nicht vollständig angegeben, um die Beispiele übersichtlicher zu halten.

10.3.1 Bereitgestellte REST Ressource

Kategorie	Operation	HTTP Method	Resource-URI	Request Content-Type	Response Content-Type	Re Val Scl
Zertifikat	Aktualisieren der Zertifikate aus der KV-Connect Benutzerverwaltung	POST	`/certificates`	text/plain; charset=UTF-8 entschlüsselt: certificatesIntern.xsd	text/plain; charset=UTF-8 entschlüsselt: certificatesIntern.xsd	S/M

10.3.2 Angesprochene REST Ressource

Diese Ressource wird nicht vom KV-Connect Server bereitgestellt; sie wird von diesem genutzt, um Zertifikate von mobilen Geräten abzurufen:

Kategorie	Operation	HTTP Method	Resource-URI	Request Content-Type	Response Content-Type	Response Value / Schema	HT St Co Sc
Zertifikat	Abrufen der Zertifikate vom externen Anbieter	GET, HEAD	/certificates	text/plain; charset=UTF-8	application /xml text/plain	certificatesExtern. xsd	20 40

10.4 Mail-Formate

Aufbauend auf den Inhalt der Nachricht (siehe oben) sind folgende Formate zum sicheren Nachrichtenaustausch definiert:

- Mail-Versand zum externen Nutzer
- Mail-Versand zum Freigabeempfänger
- MDN vom externen Nutzer an sendenden Arzt
- MDN vom empfangenden Arzt an den externen Nutzer

10.5 KV-Connect-Adresse und UID

Die Struktur der KV-Connect-Adresse und der UID eines externen Nutzers sind in Weitere Formate definiert.

10.6 Verbindung zwischen den Servern der KVTG und des Anbieters

Bedingungen für die Verbindung zwischen den Servern der KVTG und des Anbieters sind in Verbindung zwischen den Servern der KVTG und denen des Anbieters beschrieben.

10.7 Beispiel-Code

Informell sind in Beispiel-Code Beispiele für die kryptografischen Operationen beschrieben.

10.8 Zertifikat holen

Zum Versenden einer signierten und verschlüsselten E-Mail muss vor jedem Versand das aktuelle Zertifikat des Empfängers angefordert werden. Dafür steht die hier beschriebene Schnittstelle zur Verfügung. Über diese Schnittstelle könnten berechtigte Nutzer Zertifikate anderer Nutzer herunterladen.

Um diese Schnittstelle verwenden zu können, muss der Nutzer im Besitz eines eigenen gültigen Zertifikates sein. Zusätzlich benötigt der Nutzer ein (Alt-)Zertifikat des Nutzers, dessen Zertifikat er herunterladen möchte. Dieses (Alt-)Zertifikat dient der Legitimation zum Benutzen dieser Schnittstelle, ohne das diese Schnittstelle nicht verwendet werden.

Damit Anbieter nur beschränkt Informationen über die jeweiligen Kommunikationspartner beim E-Mail Austausch erhalten können, wird die Abfrage eines Zertifikates teilweise verschlüsselt. Hierfür verpackt der Nutzer das Alt-Zertifikat in eine XML Struktur und diese wiederum in eine S/MIME, die mit seinem privaten Schlüssel signiert und mit dem öffentlichen Schlüssel des KVTG Mailer-Accounts mailer.sy.kvtg verschlüsselt wird. Dabei dient S/MIME nur als Transportmedium und es sind außer MIME-Version, Content-Disposition, Content-Type und Content-Transfer-Encoding keine zusätzlichen Header-Infomationen erforderlich. Das öffentliche Zertifikat der Certificate Authority, mit dem das Enduser-Zertifikat erstellt wurde, muss ebenfalls in der S/MIME übermittelt werden.

Nach Erhalt des signierten und verschlüsselten Alt-Zertifikates im Request prüft der Server die Legitimation und sendet ebenfalls signiert und verschlüsselt das aktuelle Zertifikat des angefragten Nutzers (Besitzer des Alt-Zertifikates) im Response zurück. Für die Signatur wird dabei der private Schlüssel des KVTG Mailer-Accounts mailer.sy.kvtg verwendet und für die Verschlüsselung der öffentliche Schlüssel des anfragenden Nutzers.

10.8.1 Ressource

URL	<base-url>/certificates
HTTP-Method	POST
Request Content-Type	text/plain;charset=UTF-8
Response Content-Type	text/plain;charset=UTF-8
HTTP Status Codes	200, 304 400, 403, 404, 422

10.8.2 Request

```
POST /certificates HTTP/1.1

MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime; smime-type=enveloped-data; name="
smime.p7m"
Content-Transfer-Encoding: base64

U2FsdGVkX1+EwIoKQRi0SsVHJNJp5lb8igad+6iFASvM+BA7zUuwOsKfOD+qjWrD
DAPw0mQSOvbEtoCFief2V6zWmWVwlB/jOANd8eUGIQSajjBx8eYdh3NYD1PO+6u7
lh9AilTRKkxb5i2akOu7Csm7yj9FKO4F357wC4YZ7SwJqjPYMp3CTwjEX4GjloBR
...
FpquOYh/WuawyBnpzY9I0Ug2PbIgegXHynUmB8j4wRJSnV6qxm84L8vFfjhvBUli
3tiT+qJ1lhs87sxmTN1tnQ==
```

Entschlüsselter Inhalt des Requests

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<certificate>
  <email>petra.meier@kv-safenet.de</email>
  <validFrom>Wed Feb 14 15:30:21 CET 2018</validFrom>
  <validTo>Sun Feb 14 15:30:21 CET 2021</validTo>
  <body>-----BEGIN CERTIFICATE-----&#xD;
MIIE9TCCAt2gAwIBAgIIRWPxUPqe7uYwDQYJKoZIhvcNAQENBQAwTzEkMCIGA1UE&#xD;
AwwbS1YgVGVsZW1hdGlrIEdtYkggVXNlciBDQSAxMRowGAYDVQQKDBFLViBUZWxl&#xD;
...
6qKxtEYoYi9t77OlxBJq39Od9szR6FX4Qi7ycNTRAoZdD53QHu2cJRmSqgXUvXkv&#xD;
RzXY8sN5c5Zbv8ZvWO640JSC3L8N88D3ug==&#xD;
-----END CERTIFICATE-----&#xD;
</body>
</certificate>
```

Die XML Struktur ist hier die gleiche, wie sie auch zum Abruf eines Zertifikats mittels UID beim KV-Connect Server verwendet wird. Lediglich der ID Parameter ist nicht vorhanden.

XML Tag	Beschreibung
email	die KV-Connect Adresse, zu der das Zertifikat gehört
validFrom	Datum und Uhrzeit der Ausstellung des übermittelten Zertifikates
validTo	Ablaufdatum und Uhrzeit des übermittelten Zertifikates
body	das übermittelte Zertifikat

10.8.3 Response im Erfolgsfall

Response mit dem aktuellen Zertifikat des Nutzers

```
HTTP/1.1 200 OK

MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime; smime-type=enveloped-data; name="
smime.p7m"
Content-Transfer-Encoding: base64

U2FsdGVkX18Z4lHJjb4mcYh4/OOtutTgxrfPa12eHeAQgbHi9R8C3gudL80/thqS
xMvAiXe5IdO9+++VMf9yzX7IC8pANFjrxhEzlWSq3sNwZVQPzhMUYggq3f/nO5yP
bHwr5omHGMk1bXyrHwc0HbNcLP+QkNX5CxlH/tFNkH1WC7ccF7Y36A8LHg1pIh1v
...
1t4r8vxUPFEaker5jTVBPxvfcq+R6dRWbGZp8qbTErIeNNCdkpcA7noJGF2PhiCw
BxHcypz0+LuDNEb27GFXAw==
```

Entschlüsselter Inhalt der Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<certificate>
  <email>petra.meier@kv-safenet.de</email>
  <validFrom>Thu Feb 15 15:30:21 CET 2018</validFrom>
  <validTo>Mon Feb 15 15:30:21 CET 2021</validTo>
  <body>-----BEGIN CERTIFICATE-----&#xD;
MIIE9TCCAt2gAwIBAgIIRWPxUPqe7uYwDQYJKoZIhvcNAQENBQAwTzEkMCIGA1XO&#xD;
AwwbS1YgVGVsZW1hdGlrIEdtYkggVXNlciBDQSAxMRowGAYDVQQKDBFLViBUZWqr&#xD;
...
6qKxtEYoYi9t77OlxBJq39Od9szR6FX4Qi7ycNTRAoZdD53QHu2cJRmSqgXUvXQW&#xD;
RzXY8sN5c5Zbv8ZvWO640JSC3L8N88D3ug==&#xD;
-----END CERTIFICATE-----&#xD;
</body>
</certificate>
```

Response, wenn das eingereichte Zertifikat dem aktuellen Zertifikat des Nutzers entspricht

```
HTTP/1.1 304 Not Modified
```

10.8.4 Response im Fehlerfall

```
HTTP/1.1 400 Bad Request

Inhalt konnte nicht entschlüsselt werden
```

```
HTTP/1.1 400 Bad Request

Signatur konnte nicht verifiziert werden
```

```
HTTP/1.1 400 Bad Request

Signatur-Zertifikat ist ungültig
```

```
HTTP/1.1 400 Bad Request

xml konnte nicht gelesen werden
```

```
HTTP/1.1 422 unprocessable entity

<required_element> konnte nicht verarbeitet werden
```

```
HTTP/1.1 403 Forbidden

Gesendetes Zertifikat konnte nicht verifiziert werden
```

```
HTTP/1.1 404 Not Found

Nutzer mit dieser email wurde nicht gefunden
```

Beispiele für die kryptografischen Operationen sind in Beispiel-Code beschrieben.

10.9 Zertifikat holen

Diese Schnittstelle wird vom KV-Connect Server angesprochen um das Zertifikat eines externen Nutzers vom Server seines Anbieters abzurufen. Sowohl die Email-Adresse als auch die in Weitere Formate beschriebene UID können in der Anfrage verwendet werden. Die Email-Adresse bzw. UID müssen URL-enkodiert (RFC 3986) angegeben werden. Die Kodierung bezieht sich nur auf die URL; im Response erscheinen die Email-Adresse bzw. UID immer im Klartext.Neben GET ist auch HEAD definiert.

10.9.1 Ressource

URL	<base-url>/certificates?email=<email-adresse> <base-url>/certificates?uid=<uid>
HTTP-Method	GET, HEAD
Response Content-Type	application/xml text/plain
HTTP Status Codes	200, 404

10.9.2 Request

```
GET /certificates?email=michaela.mueller.123456.anbieter%40kv-safenet.de
HTTP/1.1
```

```
GET /certificates?uid=f81d4fae7dec11d0a76500a0c91e6bf6 HTTP/1.1
```

10.9.3 Response im Erfolgsfall

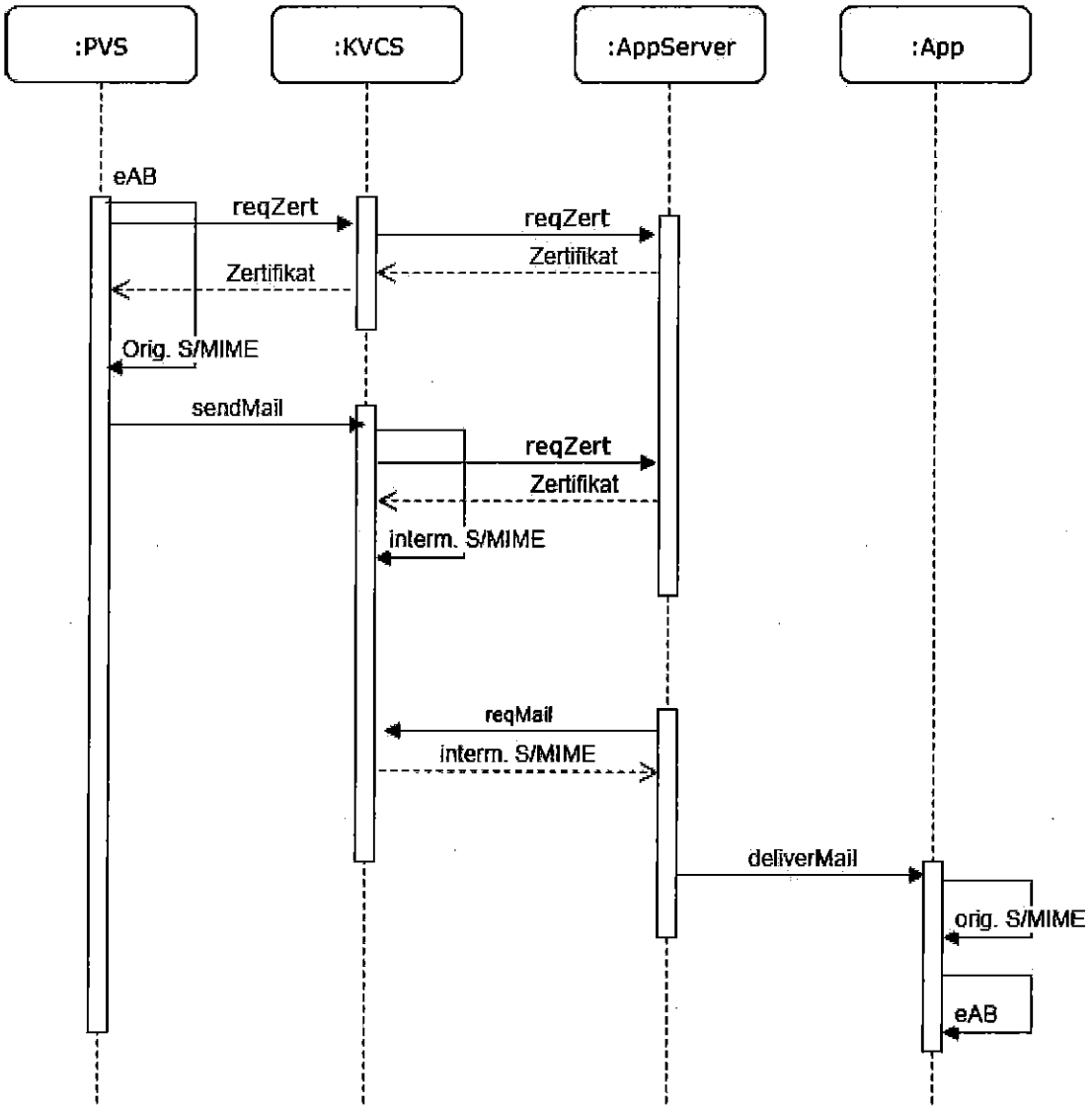
```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<certificate>
  <uid>9811964c41064367aa7b781480376e8c</uid>
  <email>michaela.mueller.123456.anbieter@kv-safenet.de</email>
  <validFrom>Thu Feb 15 15:30:21 CET 2018</validFrom>
  <validTo>Mon Feb 15 15:30:21 CET 2021</validTo>
  <body>-----BEGIN CERTIFICATE-----&#xD;
MIIE9TCCAt2gAwIBAgIIRWPxUPqe7uYwDQYJKoZIhvcNAQENBQAwTzEkMCIGA1cv&#xD;
AwwbS1YgVGVsZW1hdGlrIEdtYkggVXNlciBDQSAxMRowGAYDVQQKDBFLViBUZWdf&#xD;
...
6qKxtEYoYi9t77OlxBJq39Od9szR6FX4Qi7ycNTRAoZdD53QHu2cJRmSqgXUvXeR&#xD;
RzXY8sN5c5Zbv8ZvWO640JSC3L8N88D3ug==&#xD;
-----END CERTIFICATE-----&#xD;
</body>
</certificate>
```

10.9.4 Response im Fehlerfall

```
HTTP/1.1 404 Not Found

Zertifikat zu dieser email wurde nicht gefunden
```

10.10 Mail-Versand zum externen Nutzer

Das folgende Vorgehen wird für jeden Anbieter durchgeführt, für den Nutzer als To- oder CC-Adressaten in einer KV-Connect Nachricht enthalten sind.



Ausgegangen wird hierbei von einer MIME-Nachricht im unter Inhalt definierten Format (z.B. eArztbrief, im Ablaufdiagramm "eAB"). Diese wird entsprechend der Kryptographischen Standards mit dem privaten Schlüssel des Absenders im PVS signiert und mit dem vom KV-Connect Server abzurufenden (öffentlichen) Zertifikat des Empfängers verschlüsselt. Das Zertifikat wird dabei über die REST-Ressource Abruf eines Zertifikats mittels UID vom KV-Connect Server abgerufen (im Ablaufdiagramm "reqZert"), welcher seinerseits das Zertifikat mittels Abrufen der Zertifikate vom externen Anbieter abruft. Diese Original-S/MIME Nachricht wird dann vom PVS über die REST-Ressource Senden einer Mail an den KV-Connect Server übertragen. Der Server signiert die Original-S/MIME inklusive Header mit dem privaten Schlüssel des KVTG Mailer-Accounts mailer.sy.kvtg und verschlüsselt sie für alle externen Empfänger des Anbieters mittels der vom Anbieter abzurufenden (öffentlichen) Zertifikate der Empfänger in eine intermediäre S/MIME. Die Header-Felder sind:

- Date: <aktuelles Datum>
- From: mailer.sy.kvtg@kv-safenet.de
- To: und CC: <externe Nutzer des Anbieters aus dem Original-Header>
- Subject: eMail-Dienst;Lieferung;V1.0
- Message-ID: <neue zufällige Message-ID>
- MIME-Version: 1.0
- X-KVC-Sendersystem: KVC-Mobile;V1.0
- X-KVC-Dienstkennung: eMail-Dienst;Lieferung;V1.0

Diese intermediäre S/MIME dient der Verschleierung der Arzt-Adressen, die somit für den empfangenden externen Nutzer, nicht aber für den Anbieter sichtbar sind. Sie wird später auf dem Endgerät des externen Nutzers empfangen, entschlüsselt und Signatur-geprüft. Anschließend wird die darin enthaltene Original-S/MIME entschlüsselt Signatur-geprüft.

Beispiel für eine intermediäre S/MIME:

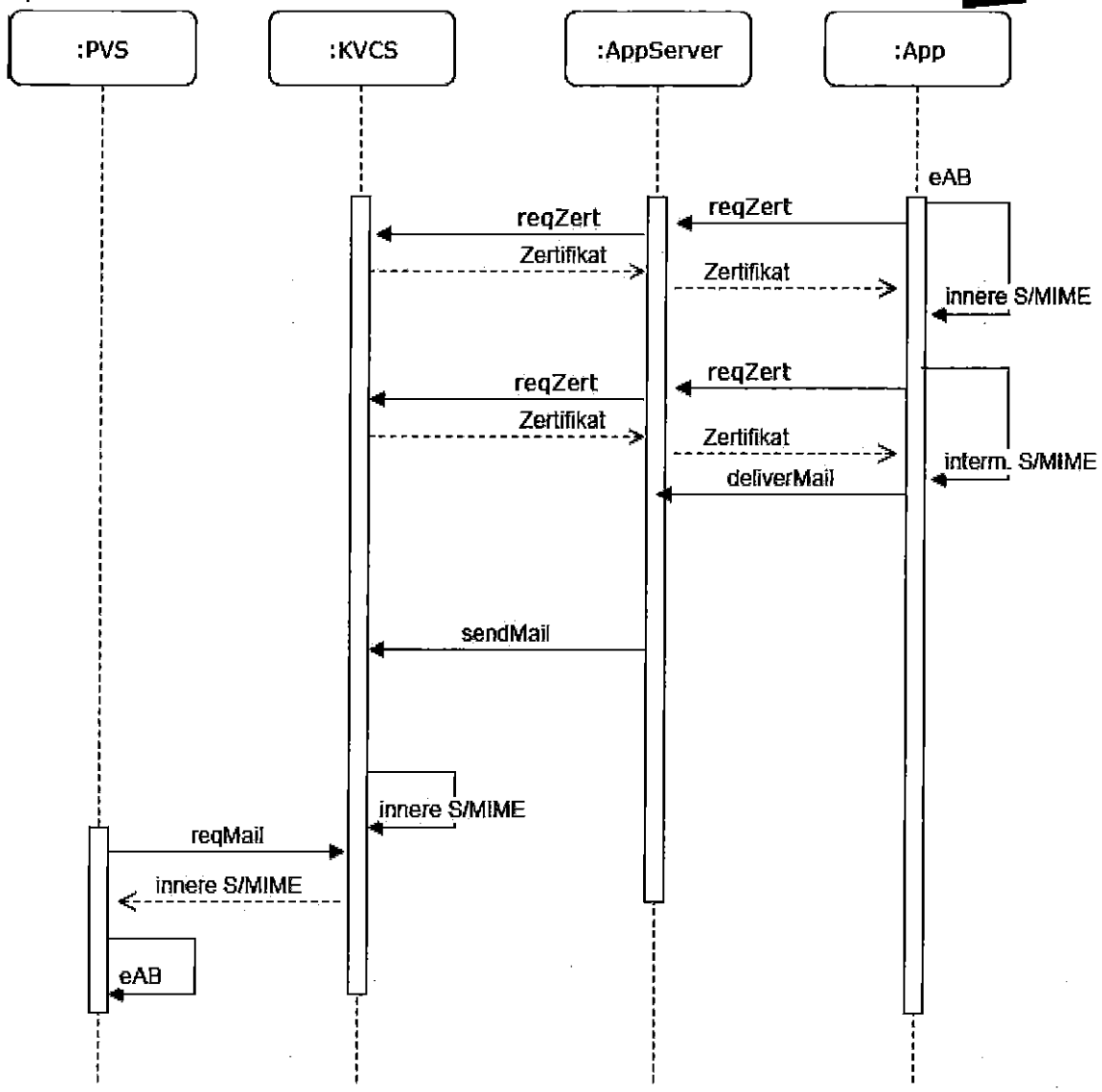
```
X-KVC-Sendersystem: KVC-Mobile;V1.0
X-KVC-Dienstkennung: eMail-Dienst;Lieferung;V1.0
Message-ID: <12345-67890-abcd@00>
Date: Fri, 23 Mar 2018 11:59:02 +0100 (CET)
To: michaela.mueller.123456.meinanbieter@kv-safenet.de
From: mailer.sy.kvtg@kv-safenet.de
Subject: eMail-Dienst;Lieferung;V1.0
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime; smime-type=enveloped-data; name="smi
Content-Transfer-Encoding: base64

MIIbEQYJKoZIhvcNAQcDoIIbAjCCGv4CAQAxggFuMIIBagIBADBSMEYxGzAZBgNV
BAMMEktWVEcgVXNlciBDQTIgUmVmMTEaMBgGA1UECgwRS1YgVGVsZW1hdGlrIEdt
...
iKRcNJW2XQVfuXNDGDxumlHn735GzvEPgysn0BpgGhPIC29ENN9V+6ecvDbhMbIX
+ifXeBEBgzIxHsQoi03GgXvy8HeLKyzo4/w/LwDYq+jees2nYfYd+J+2I5zUeOlL
WSsoM8cwwQQrd/F3VNMoQ1XBmgks
```

Die intermediäre S/MIME wird dem Mailer-Account des Anbieters zugeordnet und vom Server des Anbieters über die entsprechenden REST-Ressourcen des KV-Connect Servers abgerufen. Der Mailer-Account des Anbieters wird hierbei für die Authentisierung verwendet, im To-Feld befindet sich jedoch die Adresse des externen Nutzers. Anschließend wird die intermediäre S/MIME auf das Endgerät des externen Nutzers übertragen, dort entschlüsselt und Signatur-geprüft und die darin enthaltene innere (originale) S/MIME entschlüsselt und Signatur-geprüft.

Beispiele für die kryptografischen Operationen sind in Beispiel-Code beschrieben.

10.11 Mail-Versand zum Freigabeempfänger

Dieser Ablauf beschreibt das Versenden einer Nachricht gemäß Inhalts-Spezifikation vom externen Nutzer an einen Arzt. Bei der zu versendenden Nachricht handelt es sich zwingend um eine von einem anderen Arzt erhaltene MIME-Nachricht mit Original-Inhalt und Original-Signatur und den unten beschriebenen Änderungen an Header-Feldern, also um eine Weiterleitung im Sinne einer Freigabe dieser Nachricht durch den externen Nutzer für einen anderen Arzt. Eine etwaige vorhandene Anforderung einer MDN muss vor der Freigabe durch Weiterleitung entfernt werden. Es kann eine MDN vom empfangenden Arzt an den externen Nutzer angefordert werden.



Das Zertifikat aus einer vom zu adressierenden Arzt empfangenen Nachricht wird extrahiert. Es wird mittels der Aktualisieren der Zertifikate aus der KV-Connect Benutzerverwaltung aktualisiert. Mit dem aktualisierten Zertifikat wird die entschlüsselte signierte Original-MIME verschlüsselt und eine KV-Connect konforme (innere) S/MIME erzeugt (siehe Kryptographische Standards). Diese wird später dem empfangenden Arzt zugestellt, der die Freigabe erhält. Header-Felder sind hierbei:

- Date: <aktuelles Datum>
- Reply-To: <externer Nutzer des Anbieters als Absender der Freigabe>
- From: <Absender der Original-S/MIME>
- To: <Freigabeempfänger>
- Subject: <Original-Subject>
- Message-ID: <neue zufällige Message-ID>
- MIME-Version: 1.0
- X-KVC-Sendersystem: <externes Sendersystem inkl. Version>
- X-KVC-Dienstkennung: <wie Original-S/MIME>

BCC darf nicht verwendet werden.

Beispiel für eine (innere) S/MIME einer freizugebenen Original-MIME:

```
X-KVC-Sendersystem: Anbieter-Mobile;V1.0
X-KVC-Dienstkennung: Arztbrief;VHitG-Versand;V1.2
```

```
Date: Fri, 23 Mar 2018 12:22:00 +0100 (CET)
Reply-To: michaela.mueller.123456.meinanbieter@kv-safenet.de
Message-ID: <12345-67890-abcd@00>
To: petra.meier@kv-safenet.de
From: frank.schmidt@kv-safenet.de
Subject: Arztbrief;VHitG-Versand;V1.2
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime; smime-type=enveloped-data; name="smi
Content-Transfer-Encoding: base64

MIIbEQYJKoZIhvcNAQcDoIIbAjCCGv4CAQAxggFuMIIBagIBADBSMEYxGzAZBgvb
BAMMEktWVEcgVXNlciBDQTIgUmVmMTEaMBgGA1UECgwRS1YgVGVsZW1hdGlrIEfg
...
iKRcNJW2XQVfuXNDGDxumlHn735GzvEPgysn0BpgGhPIC29ENN9V+6ecvDbhMbGH
+ifXeBEBgzIxHsQoi03GgXvy8HeLKyzo4/w/LwDYq+jees2nYfYd+J+2I5zUeOrE
WSsoM8cwwQQrd/F3VNMoQ1XBmgks
```

Anschließend wird die innere S/MIME, die die Original-MIME enthält, mit dem privaten Schlüssel des externen Nutzers signiert und für den KVTG Mailer-Account mailer.sy.kvtg in eine intermediäre S/MIME verschlüsselt. Das Zertifikat des KVTG Mailer-Accounts mailer.sy.kvtg wird dazu mittels der Aktualisieren der Zertifikate aus der KV-Connect Benutzerverwaltung aktualisiert. Eine initiale Version muss im Client (App) des externen Nutzers vorhanden sein. Header-Felder dieser intermediären S/MIME sind:

- Date: <aktuelles Datum>
- From: <externer Nutzer>
- To: mailer.sy.kvtg@kv-safenet.de
- Subject: eMail-Dienst;Freigabe;V1.0
- Message-ID: <identisch mit Message-ID der inneren S/MIME>
- MIME-Version: 1.0
- X-KVC-Sendersystem: <externes Sendersystem inkl. Version>
- X-KVC-Dienstkennung: eMail-Dienst;Freigabe;V1.0

Beispiel für eine intermediäre S/MIME:

```
X-KVC-Sendersystem: Anbieter-Mobile;V1.0
X-KVC-Dienstkennung: eMail-Dienst;Freigabe;V1.0
Date: Fri, 23 Mar 2018 12:23:02 +0100 (CET)
Message-ID: <12345-67890-abcd@00>
To: mailer.sy.kvtg@kv-safenet.de
From: michaela.mueller.123456.meinanbieter@kv-safenet.de
Subject: eMail-Dienst;Freigabe;V1.0
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime; smime-type=enveloped-data; name="smi
Content-Transfer-Encoding: base64

MIIbEQYJKoZIhvcNAQcDoIIbAjCCGv4CAQAxggFuMIIBagIBADBSMEYxGzAZBglP
BAMMEktWVEcgVXNlciBDQTIgUmVmMTEaMBgGA1UECgwRS1YgVGVsZW1hdGlrIEop
...
iKRcNJW2XQVfuXNDGDxumlHn735GzvEPgysn0BpgGhPIC29ENN9V+6ecvDbhMbPO
+ifXeBEBgzIxHsQoi03GgXvy8HeLKyzo4/w/LwDYq+jees2nYfYd+J+2I5zUeOaq
WSsoM8cwwQQrd/F3VNMoQ1XBmgks
```

Diese intermediäre S/MIME wird an den Server des Anbieters übertragen. Der Anbieter übermittelt sie dann über die enstprechenden REST-Ressourcen des KV-Connect Servers an das Postfach des KVTG Mailer-Accounts mailer.sy.kvtg, wobei der Mailer-Account des Anbieters zur Authentifikation verwendet wird und somit nicht dem Wert des From-Felds entspricht. Der Absender im From-Feld muss jedoch dem Adress-Schema des Anbieters entsprechen.

Anschließend wird vom KV-Connect Server die intermediäre S/MIME entschlüsselt und Signatur-geprüft und die darin enthaltene innere (freizugebene) S/MIME dem empfangenden Account zugeordnet. Das PVS des empfangenden Arztes holt die S/MIME ab, entschlüsselt sie in die ursprüngliche Nachricht gemäß Inhalts-Spezifikation und prüft die Signatur und das dafür verwendete Zertifikat.

Beispiele für die kryptografischen Operationen sind in Beispiel-Code beschrieben.

10.12 MDN vom externen Nutzer an sendenden Arzt

Das empfangende System kann - falls angefordert - gemäß der Spezifikation des Inhalts der Nachricht (siehe Inhalte) eine MDN an den sendenden Arzt zurücksenden. Die MDN wird dabei gemäß der Spezifikation MDN (anwendungsübergreifend) erstellt, signiert und verschlüsselt und als KV-Connect konforme (innere) S/MIME wie beim Mail-Versand zum Freigabeempfänger signiert und für den KVTG Mailer-Account mailer.sy.kvtg verschlüsselt und versendet.

10.13 MDN vom empfangenden Arzt an den externen Nutzer

Die Anwendung des externen Nutzers kann eine MDN vom empfangenden Arzt anfordern. Als Empfänger der MDN muss gemäß Spezifikation MDN (anwendungsübergreifend) die KV-Connect Adresse des externen Nutzers angegeben werden. Das empfangende System kann, wenn angefordert, eine MDN an den externen Nutzer zurücksenden. Diese MDN muss von der Anwendung des externen Nutzers angenommen werden und kann als Empfangsbestätigung zu der freigegebenen Nachricht angezeigt werden.Die MDN wird dabei gemäß der Spezifikation MDN (anwendungsübergreifend) erstellt, signiert und verschlüsselt und als KV-Connect konforme (innere) S/MIME wie beim Mail-Versand zum externen Nutzer behandelt.

10.14 Weitere Formate

Externe Nutzer werden über ihre KV-Connect Adresse und eine eindeutige UID identifiziert. Die Formate dieser Informationen sien in diesem Abschnitt definiert.

10.14.1 KV-Connect Adresse

Der externe Anbieter vergibt eine KV-Connect Adresse im Format

<PersonIdentifier>.<6Digits>.<Anbieter>@kv-safenet.de

<PersonIdentifier> ist frei wählbar, sollte aber Bezug zum Nutzer haben, z.B. <Vorname>.<Nachname>. Erlaubte Zeichen sind [a..zA..Z0..9_-.].

<6Digits> eine beliebige 6-stellige Zahlenfolge aus [0..9]; muss für gleiche oder ähnliche <PersonIdentifier> bei einem Anbieter unterschiedlich sein.

<Anbieter> eine eindeutige Identifikation des Anbieters aus [a..zA..Z0..9_-], die bei der Beantragung angegeben wird. Es können in begründeten Fällen mehrere unterschiedliche Werte für einen Anbieter registriert werden.

Die KV-Connect Adresse ist case-insensitiv. Die Länge des local-parts (vor dem "@") darf insgesamt maximal 64 Zeichen sein.

Der local-part muss im gesamten System des Anbieters eindeutig sein.

Beispiel:

```
petra.mueller-meier.123456.mein-anbieter@kv-safenet.de
```

10.14.2 UID

Ein Anbieter muss zu jeder vergebenen KV-Connect Adresse eine eindeutige UID speichern. Die UID besteht dabei aus 32 Hexadezimalstellen in lower-case und entspricht der MD5-Checksum des local-part der KV-Connect Adresse. Ein Anbieter muss sicherstellen, dass die UID systemweit eindeutig ist.

Beispiel:

```
lowercase(MD5-Checksum("petra.mueller-meier.123456.mein-anbieter"))

UID: e06fbe6b7717ed79dde692535ebe0fe2
```

10.15 Verbindung zwischen den Servern der KVTG und denen des Anbieters

Die übertragung von Nachrichten zwischen KV-Connect Nutzern und externen Nutzern sowie der Austausch entsprechender Zertifikate erfolgt über eine Verbindung zwischen KV-Connect Server und einem Server des externen Anbieters. Für die Verbindung zwischen den Servern der KVTG und denen des Anbieters ist eine TLS1.2-verschlüsselte Verbindung vorgeschrieben. Es muss eine Vertrauensstellung (keystore-truststore) hergestellt werden. Zulässig sind nur bestimmte Cipher Suites.

Sofern selbstsignierte Zertifikate verwendet werden, müssen diese zwischen der KVTG und dem Anbieter ausgetauscht werden und in den truststore des jeweils anderen Servers eingepflegt werden. Der Anbieter muss in diesem Fall der KVTG das Rootzertifikat zur Verfügung stellen, damit die KVTG im Falle einer Endzertifikatserneuerung beim Anbieter nicht aktiv werden muss.

Von vertrauenswürdigen CAs ausgestellte Zertifikate sind ebenfalls zulässig.

10.16 Beispiel-Code

10.16.1 Aktualisieren der Zertifikate aus der KV-Connect Benutzerverwaltung

Beispiel für das Signieren und die Verschlüsselung

```
openssl smime -sign -signer nuzter.pub.pem -inkey nutzer.priv.key -in
alt.cert.xml -certfile anbieterca.pem -out alt.cert.xml.signed
openssl smime -encrypt -aes-256-cbc -binary -in alt.cert.xml.signed -out
alt.cert.xml.smime kvtg.mailserver.pem
```

Beispiel für die Entschlüsselung und Prüfung

```
openssl smime -decrypt -aes-256-cbc -binary -recip nutzer.priv.key -in
new.cert.xml.smime -out new.cert.xml.signed
openssl smime -verify -binary -CAfile rootAndSub.pem -in new.cert.xml.
signed -out new.cert.xml
```

10.16.2 Mail-Versand zum externen Nutzer

Beispiel für die Erzeugung einer intermediären S/MIME

```
openssl smime -sign -in original.smime -text -out original.smime.signed -sig
openssl smime -encrypt -in original.smime.signed -from mailer.sy.kvtg@kv-saf
echo -e "X-KVC-Sendersystem: KVC-Mobile;V1.0\r\nX-KVC-Dienstkennung: eMail-D
```

10.16.3 Mail-Versand zum Freigabeempfänger

Beispiel für die Erzeugung der inneren S/MIME

```
openssl smime -encrypt -in original.mime -from frank.schmidt@kv-safenet.de
echo -e "X-KVC-Sendersystem: Anbieter-Mobile;V1.0\r\nX-KVC-Dienstkennung: An
```

Beispiel für die Erzeugung einer intermediären S/MIME

```
openssl smime -sign -in freigabe.smime -text -out freigabe.smime.signiert -s
openssl smime -encrypt -in freigabe.smime.signiert -from michaela.mueller.12
echo -e "X-KVC-Sendersystem: Anbieter-Mobile;V1.0\r\nX-KVC-Dienstkennung: eM
```