

IRD Technische Spezifikation für Krankenversicherungsträger (API-Version 1.0)

Dokumenteninformation

Version	Datum	Grund der Änderung, besondere Hinweise	Bearbeiter
1.0	18.04.2024	Initialversion	RKI Referat VIG (Vertrauensstelle-ird@rki.de)
1.1	07.05.2024	<ul style="list-style-type: none">• Korrektur Angaben zum Vitalstatus (Enum)• Änderungen bzgl. nur Rückgabe von Fehlern• Ergänzung Authentisierung und Authentifizierung• redaktionelle Änderungen	RKI Referat VIG (Vertrauensstelle-ird@rki.de)
1.2	23.05.2024	<ul style="list-style-type: none">• bei den Meldungen Vitalstatus und Versicherungswechsel wird nur grundsätzliche Statuscode zurückgegeben• die Liste zu fehlerhaften Datensätzen entfällt; diese Fehler werden jetzt asynchron zur Abholung bereitgestellt	RKI Referat VIG (Vertrauensstelle-ird@rki.de)
1.3	28.06.2024	verbindliche Version für die API Version 1.0 <ul style="list-style-type: none">• Anmelde-Token spezifiziert• einmalige Registrierung bei der Vertrauensstelle angepasst• Downloadpunkt für den public key der Vertrauensstelle beschrieben• Prüfung der Gültigkeit der Schlüssel und Zertifikate der Vertrauensstelle beschrieben• Informationen zum patienten-identifizierenden Datum ergänzt• Sitzungsschlüssel spezifiziert	RKI Referat VIG (Vertrauensstelle-ird@rki.de)
1.4	19.07.2024	aktualisierte verbindliche Version für die API Version 1.0 <ul style="list-style-type: none">• Formatierungen und Links angepasst• Hinweis zu JSON Parser Konformität• Detailliertes Beispiel zum Anmelde-Token• Beispiel zu verschlüsselten Ausgangsdaten• Format Id-Nummer Heilfürsorge BW	RKI Referat VIG (Vertrauensstelle-ird@rki.de)

Inhaltsverzeichnis

- [Dokumenteninformation](#)
- [Inhaltsverzeichnis](#)
- [Einleitung](#)
- [Allgemeine Informationen](#)
 - [Umgebungen](#)
 - [Registrierung bei der Vertrauensstelle](#)
 - [Authentisierung und Authentifizierung](#)

- Downloadpunkt für den Abruf von Schlüsselmaterial der Vertrauensstelle
- Datenformate
 - Eingabeformat
 - Rückgabe
- Signierung von Daten
 - Signierung der Eingangsdaten
 - Signierung der Ausgangsdaten
- Verschlüsselung von Daten
 - Verschlüsselung der Eingangsdaten
 - Verschlüsselung der Ausgangsdaten
- HTTP-Programmierschnittstellen (APIs)
 - API für die Übertragung von Vitalstatusmeldungen für im Implantateregister erfasste Versicherte
 - Fehlerfälle
 - Synchrone Fehlermeldungen
 - Beispiel Request
 - Signaturbildung der Eingangsdaten
 - API für den Abruf von Anfragen zu Vitalstatusmeldungen für im Implantateregister erfasste Versicherte
 - Fehlerfälle
 - Synchrone Fehlermeldungen
 - Beispiel Response
 - Signierung der Ausgangsdaten
 - API für den Abruf der Benachrichtigungen über die Anonymisierung der Daten von im Implantateregister erfassten Versicherten
 - Fehlerfälle
 - Synchrone Fehlermeldungen

- [Beispiel Response](#)
- [Signierung der Ausgangsdaten](#)
- [API für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Vitalstatus-Meldungen auf Seiten der Vertrauensstelle oder der Registerstelle aufgetreten sind](#)
 - [Fehlerfälle](#)
 - [Synchrone Fehlermeldungen](#)
 - [Beispiel Response](#)
 - [Signierung der Ausgangsdaten](#)
- [API für die Übertragung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten](#)
 - [Fehlerfälle](#)
 - [Beispiel Request](#)
 - [Signaturbildung der Eingangsdaten](#)
- [API für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten in der Vertrauensstelle aufgetreten sind](#)
 - [Fehlerfälle](#)
 - [Synchrone Fehlermeldungen](#)
 - [Beispiel Response](#)
 - [Signierung der Ausgangsdaten](#)
- [Übergreifende Beispiele](#)
 - [Verschlüsselung der Eingangsdaten](#)
 - [Signierung der Eingangsdaten](#)
 - [Generelle Informationen zu den Schnittstellen des Konnektors bzw. Basis-/KTR-Consumer](#)

- [Generelle Informationen zu Parametern in SOAP-Requests](#)
- [Prüfung eines TI-Zertifikates](#)
- [Freigabe der SMC-B durch Eingabe der PIN](#)
- [Abfrage der Karten des Kartenterminals](#)
- [Abfrage des AUT-Zertifikates der SMC-B-Karte](#)
- [Operation ExternalAuthenticate](#)
- [Nichtfunktionale Festlegungen](#)
- [Test](#)
 - [Definition von Testdaten](#)
 - [Testdaten Krankenversichertennummer \(KVNR\)](#)
 - [Testdaten Identifikationsnummer der Heilfürsorge BW](#)
 - [Regeln für die TI-Umgebungen](#)
- [Kontakt](#)

Einleitung

Die Vertrauensstelle des Implantateregisters Deutschland (IRD) beim Robert Koch-Institut (RKI) stellt die Schnittstelle zur Meldung von Vitalstatus und Versicherungswechseln für die Krankenversicherungsträger bereit.

Diese technische Spezifikation beschreibt die Kommunikation eines meldenden Krankenversicherungsträgers über die REST-Schnittstelle der Vertrauensstelle IRD für die Meldungen entsprechend §17 Abs. 2 IRegG und § 18 Abs. 1 IRegBV. Das sind die halbjährlichen und anlassbezogenen Meldungen der Vitalstatus der im Implantateregister Deutschland erfassten Versicherten sowie die Information zum Wechsel des Krankenversicherungsträgers dieses Personenkreises. Als Krankenversicherungsträger sind hier die gesetzlichen Krankenkassen, die privaten Krankenversicherungsunternehmen und die sonstigen Kostenträger nach §17 Abs. 3 IRegG zusammengefasst.

Inhaltliche Pflichtangaben sind Felder, die sich mindestens aus §17 Abs. 2 IRegG (siehe https://www.gesetze-im-internet.de/iregg/_17.html) begründen sowie weitere technische Felder.

Von der Vertrauensstelle IRD werden für den Produktivbetrieb jeweils die aktuell veröffentlichte Version der Schnittstelle sowie die vorhergehende Version unterstützt.

Allgemeine Informationen

Die API Version 1.0 akzeptiert als patienten-identifizierendes Datum den unveränderbaren Teil der Krankenversichertennummer (KVNR) nach § 290 Absatz 1 Satz 2 SGB V oder eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer nach § 17 Absatz 4 Satz 3 IRegG.

Alle gesetzlichen und privaten Krankenversicherungsträger einschließlich der Heilfürsorge Bundespolizei verwenden den unveränderbaren Teil der Krankenversichertennummer (KVNR).

Die Heilfürsorge Bundeswehr verwendet für ihre eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer das Format der Steuer-Id entsprechend Identifikationsnummerngesetz (IDNrG).

Umgebungen

Die Vertrauensstelle ist innerhalb der Referenzumgebung (RU) der Telematikinfrastruktur unter <https://vst-ird-ru.rki-ti.de> zu erreichen und innerhalb der Produktivumgebung (PU) der Telematikinfrastruktur unter <https://vst-ird.rki-ti.de>.

Registrierung bei der Vertrauensstelle

Ein Krankenversicherungsträger muss sich vor dem ersten Aufruf einer API-Funktion einmalig bei der Vertrauensstelle mit dem Institutionskennzeichen (IK) nach § 293 SGB V und der Telematik-Id (TID) registrieren. Bei relevanten Änderungen dieser Registrierungsdaten ist eine erneute Registrierung bei der Vertrauensstelle erforderlich.

Krankenversicherungsträger mit mehreren Institutionskennzeichen verwenden zur Registrierung das Haupt-IK.

Für die Registrierung stellt die Vertrauensstelle ein Formular zur Verfügung, das ausgefüllt an die Vertrauensstelle zu senden ist.

Authentisierung und Authentifizierung

Beim Aufruf einer API-Funktion ist ein Anmelde-Token als HTTP Header Authorization zu senden, zusammen mit dem Payload, also der Meldung an die Vertrauensstelle. Dieses Token muss

- das Institutionskennzeichen (IK) - bei mehreren Standorten das sog. Haupt-IK (s. Spezifikation unter <https://www.gkv-datenaustausch.de/faq/faq.jsp> "Rundschreiben zum Institutionskennzeichen"),
- das C.HCI.AUT x.509-Zertifikat des übermittelnden Krankenversicherungsträgers
- sowie den Zeitstempel zum Zeitpunkt der Übermittlung enthalten (Unixzeit).

Die Telematik-ID wird aus dem übermittelten x.509 Zertifikat C.HCI.AUT entnommen (s. https://gemspec.gematik.de/docs/gemSpec/gemSpec_PKI/latest/#4.7). IK und Telematik-ID müssen mit den bei der Registrierung übermittelten Daten übereinstimmen.

Die Daten des Anmelde-Token müssen durch den Konnektor bzw. Basis-/KTR-Consumer des Krankenversicherungsträgers über die SOAP-Aktion "ExternalAuthenticate" mit dem C.HCI.AUT-Zertifikat der SMC-B signiert werden.

Die Bytes der Werte des Haupt-IK, Zertifikats und Zeitstempels werden konkateniert. Als Trennzeichen wird "|" (7C) verwendet.

Ein Beispiel dazu:

⚠ Der Payload (Content) muss in der Signatur eingebettet sein (Siehe [Signierung der Eingangsdaten](#))

```
####
# Header data:
# IK := 104127692
# Timestamp := 123456789 (unix timestamp for Do 29. Nov 21:33:09 UTC 1973)
# Zusammensetzen mit Trennzeichen |: 104127692|123456789
##

# We use ASCII representation an build a string concatenating both values
00000000 31 30 34 31 32 37 36 39 32 7c 31 32 33 34 35 36 |104127692|123456|
00000010 37 38 39 |789|

# The signing requires base64 encoding
00000000 4d 54 41 30 4d 54 49 33 4e 6a 6b 79 66 44 45 79 |MTA0MTI3NjkyfDEy|
00000010 4d 7a 51 31 4e 6a 63 34 4f 51 3d 3d |MzQ1Njc4OQ==|
```

Base64-Darstellung: (TODO - INHALT BASE64 codiert darstellen)

MTA0MTI3NjkyfDEyMzQ1Njc4OQ==

i Signierter Wert im HTTP-Header Authorization übermitteln

Beispiel Signatur:

```
hjCCAy2gAwIBAgIHAX/0Yt6yhzAKBggqhkJOPQDAjCBmjELMAkGA1UEBhMCREUxHzAdBgNVBAoMFmdlWbWF0aWsgR21i
```

Beispiel des konkreten HTTP Headers:

```
Host: vst-ird-ru.rki-ti.de
```

```
Content-Length
```

```
Authorization: custom <Base 64 kodiertes Ergebnis (PS7 / PKCS#7 Container / CMS)>
```

```
...
```

Zur Info: Der Aufbau der Telematik-ID ist hier spezifiziert: https://gmspec.gematik.de/docs/gemSpec/gemSpec_PKI/latest/#4.7.2. Die Telematik-ID wird im Feld registrationNumber der Extension Admission im X.509-Zertifikat (hier konkret des C.HCI.AUT) hinterlegt.

Weitere Informationen zur Signaturerstellung mit der SOAP-Aktion "ExternalAuthenticate" sowie dem Abruf des AUT-Zertifikates finden sich in den [übergreifenden Beispielen](#).

Hinweis: Die Clients müssen die Zeitsynchronisation gemäß den Vorgaben der Telematikinfrastruktur nutzen. Siehe dazu: https://gmspec.gematik.de/docs/gemSpec/gemSpec_Net/latest/#6.

Downloadpunkt für den Abruf von Schlüsselmaterial der Vertrauensstelle

Die Vertrauensstelle Implantateregister Deutschland besitzt eine digitale Identität in der Telematikinfrastruktur und ist Teilnehmer in der Komponenten-PKI der TI. Als weitere Anwendung für den Datenaustausch in der Telematikinfrastruktur (WANDA) ist das Schlüsselmaterial jedoch nicht allgemein im Verzeichnisdienst (VZD) auffindbar bzw. abrufbar. Daher werden öffentliche Schlüssel des Fachdienstes Vertrauensstelle Implantateregister als x.509 Zertifikate an einem Downloadpunkt bereitgestellt. Siehe hierzu Kapitel 5.9 *FD – Fachanwendungsspezifische Dienste* im Dokument *Übergreifende Spezifikation PKI* der gematik. Für die zum Zeitpunkt dieser Spezifikation aktuelle Version 2.18.0 ist unter https://gmspec.gematik.de/docs/gemSpec/gemSpec_PKI/latest/#5.9 abrufbar.

Das im Kontext Meldung Vitalstatus und Kassenwechsel zu verwendende Schlüsselmaterial der Vertrauensstelle steht daher für die PU unter https://vst-ird.rki-ti.de/pub_keys zum Download bereit. Für die RU sind die Schlüssel unter https://vst-ird-ru.rki-ti.de/pub_keys abrufbar. Die X.509-Zertifikate sind DER-kodiert. Für den Abruf der Zertifikate muss der Content-Type: application/pkix-cert [RFC-2582] angegeben werden.

Folgende Schlüssel stehen zur Verfügung:

- enc.der (C.FD.ENC)
- aut.der (C.FD.AUT)

i Beispiel mit curl für das AUT-Zertifikat.

RU

```
> curl --H "Content-Type: application/pkix-cert" --output vst-ird-ru-aut.crt https://vst-ird-ru.rki-ti.de/pub\_keys/TEST-ONLY/AUT.der
```

PU

```
> curl --H "Content-Type: application/pkix-cert" --output vst-ird-aut.crt https://vst-ird.rki-ti.de/pub\_keys/AUT.der
```

Temporär lokal gespeicherte Schlüssel der Vertrauensstelle sind mindestens nach 24 Stunden erneut vom Downloadpunkt abzurufen und die Gültigkeit der Zertifikate zu prüfen (siehe hierzu [Prüfung eines TI-Zertifikates](#)).

Hinweis: Der Downloadpunkt für den Abruf des Schlüsselmaterials ist aktuell im Aufbau.

Datenformate

Eingabeformat

Die Daten müssen in dem Format **application/json** zu der Schnittstelle gesendet werden. Die Zeichenkodierung wird als **UTF-8** gelesen. Namen der Eigenschaften werden Caselnsensitive ausgewertet.

Festlegungen zu den JSON-Daten

- Es werden keine Kommentare in den JSON-Daten akzeptiert. Die Verwendung von Kommentaren führt zu einem Fehler.
- Es werden keine doppelten Schlüssel (also Namen der Eigenschaften) in einem JSON-Knoten akzeptiert. Die Verwendung von doppelten Schlüsseln führt zu einem Fehler.
- Es werden nur Eigenschaften akzeptiert, die auch in der OpenAPI-Definition definiert sind. Die Verwendung unbekannter Eigenschaften führt zu einem Fehler.

Hinweis: RFC 8259 (<https://www.rfc-editor.org/rfc/rfc8259>) definiert Arrays als *ordered sequence of zero or more values*. In der Implementierung verwendete Parser sind auf entsprechende Konformität zu prüfen.

Rückgabe

Nach erfolgreicher Annahme und erfolgreicher Prüfung der Daten wird der HTTP-Statuscode 200 zurückgegeben. Der (Content) der Antwort ist dabei leer.

Signierung von Daten

Signierung der Eingangsdaten

Signaturen der Daten, die von den Krankenversicherungsträgern mit den jeweiligen Daten mitzusenden sind, gewährleisten die Integrität und die Authentizität der übertragenen Daten. Es wird hier das Schema Encrypt-Then-Sign angewendet, die Signatur wird also über die bereits verschlüsselten Daten gebildet.

Bei der Signatur handelt es sich um eine nonQES - Signatur (nicht qualifizierte elektronische Signatur).

Bei der Signatur werden die Eingangsdaten in einer konkatenierten Form zur Signaturberechnung und -prüfung betrachtet.

Für beispielsweise ein JSON-Fragment mit diesen Daten

```
{
  "IdDatenlieferung": "2024-01",
  "Meldungen": [
    {
```

```

    "IdDatensatz": "8-000001",
    "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBZXZJ0IGRlcyBJZFZlcnNpY2hlcuRlcnRlcg==",
    "Vitalstatus": "SWNoIGJpbiBkZXIgdGVyc2NobMO8c3NlbHRlIFZpdGFsc3RhdHVz",
    "Sterbedatum": "SWNoIGJpbiBkYXMGdmVyc2NobMO8c3NlbHRlIFN0ZXJiZWVhdHVt"
  },
  {
    "IdDatensatz": "8-000002",
    "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIElkVmVyc2l1j",
    "Vitalstatus": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIFZpdGFsc3RhdHVz",
    "Sterbedatum": "SWNoIGJpbiBlaW4gYW5kZXJlcyB2ZXJzY2hsw7xzc2VsdGVzIFN0ZXJiZWVhdHVt"
  }
]

```

entsteht folgendes Ergebnis:

<Wert der Eigenschaft IdDatenlieferung> + | + <Wert des IdDatensatzes Datensatz 1> + | + <Wert der Hexadezimale Darstellung (anhand des Beispiels oben)
 32 30 32 34 2D 30 31 7C 38 2D 30 30 30 30 30 31 7C 49 63 68 20 62 69 6E 20 65 69 6E 20 76 65 7

Die Plain Bytes werden an die entsprechende Schnittstelle des Konnektors bzw. des Basis-/KTR-Consumer gesendet. Nach der Operation der Signaturbildung wird dann die Signatur zurückgegeben. Diese ist dann in der Eigenschaft "Signatur" zu verwenden. Erweiterte Information zur Verwendung der Schnittstelle des Konnektors bzw. des Basis-/KTR-Consumer zur Signaturbildung und zu den konkreten Formaten finden sich unter dem Punkt [Übergreifende Beispiele](#).

Das jeweilige Beispiel zur Bildung der Zeichenkette, die zur Signaturbildung verwendet werden muss, findet sich unter den jeweiligen API.

Signierung der Ausgangsdaten

Signaturen der Daten, die von der Vertrauensstelle mit der jeweiligen Antwort mitgesendet werden, gewährleisten die Integrität und die Authentizität der übertragenen Daten.

Bei der Signatur handelt es sich um eine ECDSA-Signatur, die über den SHA256-Hash der Daten der Antwort gebildet wird.

Erweiterte Information zur Prüfung der Signatur und zu den konkreten Formaten finden sich unter dem Punkt [Übergreifende Beispiele](#).

 Die von der Vertrauensstelle ausgestellte Signatur ist vor der Verarbeitung zu prüfen.

Der öffentliche Schlüssel der Vertrauensstelle IRD für die Prüfung von Signaturen wird innerhalb der TI für die Krankenversicherungsträger an einem gesonderten Endpunkt zum regelmäßigen Download bereitgestellt. Siehe hierzu [Downloadpunkt für den Abruf von Schlüsselmaterial der Vertrauensstelle](#).

Verschlüsselung von Daten

Verschlüsselung der Eingangsdaten

Der Schutz der Datenübertragung (Transportkanal) durch die Krankenversicherungsträger an die Vertrauensstelle IRD muss per TLS gesichert sein. Es ist mindestens die Version TLSv1.2 zu verwenden. Im Allgemeinen sind die Vorgaben der Technischen Richtlinie TR-02102-2 "Kryptographische Verfahren: Empfehlungen und Schlüssellängen Teil 2 – Verwendung von Transport Layer Security (TLS)" des Bundesamt für Sicherheit in der Informationstechnik anzuwenden.

Wegen des Schutzbedarfs "Hoch" sind für eine durchgehende Ende-zu-Ende-Verschlüsselung alle patienten- und/oder fallidentifizierenden Daten vor der Übertragung für die Vertrauensstelle auf Feldebene zu verschlüsseln.

Erweiterte Informationen zur konkreten Umsetzung finden sich unter dem Punkt [Übergreifende Beispiele](#).

Die öffentlichen Schlüssel der Vertrauensstelle IRD und der Registerstelle IRD für die Verschlüsselung werden innerhalb der TI für die Krankenversicherungsträger an gesonderten Endpunkten zum regelmäßigen Download bereitgestellt. Siehe hierzu [Downloadpunkt für den Abruf von Schlüsselmaterial der Vertrauensstelle](#).

Verschlüsselung der Ausgangsdaten

Die Vertrauensstelle liefert patientenidentifizierende Daten (das Feld IdVersicherter) an die Krankenversicherungsträger auf Feldebene verschlüsselt. Dafür erzeugt die Vertrauensstelle einen symmetrischen Schlüssel, der nur für diesen Aufruf gültig ist. Zusammen mit den verschlüsselten Daten erhält die empfangende Stelle diesen abrufgebundenen Sitzungsschlüssel als Antwort, der gegen den öffentlichen Schlüssel des Empfängers (C.HCI.ENC X.509-Zertifikat des Krankenversicherungsträgers) verschlüsselt ist.


Die Parameter zur Entschlüsselung und Prüfung für die empfangende Stelle zur Entschlüsselung und Prüfung werden zusammen mit dem Chiffre übermitteln, d.h. der zu verwendende Initialisierungsvektor (IV) und der Authentication Tag.

Gemäß der Technischen Richtlinie TR-02102 des BSI muss ein Verschlüsselungsverfahren mit Datenauthentisierung genutzt werden. Es wird die Blockchiffre AES mit 256 Bits im Betriebsmodus GCM genutzt.

Die Speicherung des verschlüsselten Wertes ist dann die Konkatenation von verschlüsseltem Wert, Authentication Tag (128 Bit) und IV (96 Bit).

HTTP-Programmierschnittstellen (APIs)

Allgemeiner Hinweis

 Innerhalb der Eigenschaften "IdDatenlieferung" und "IdDatensatz" dürfen niemals patienten- bzw. fallidentifizierende Daten übermittelt werden.

API für die Übertragung von Vitalstatusmeldungen für im Implantateregister erfasste Versicherte

[POST /notify/api/v1/vitalstatusnotification](#)

Diese Schnittstelle ist für die Übertragung von Vitalstatusmeldungen für im Implantateregister Deutschland erfasste Versicherte ("Implantatträger") zu verwenden.

Eingabedaten:

Die Daten, die von der API für die Übertragung von Vitalstatusmeldungen von im Implantateregister erfassten Versicherten erwartet und verarbeitet werden, sind in der OpenAPI-Spezifikation OpenAPI_VitalStatusNotificationService-API_V1.json unter dem Typ "VitalStatusNotificationBundle" beschrieben. Die Dokumente zum API werden zukünftig unterhalb der URL <https://xml.ir-d.de/rst/download/vst/> zum Download bereitstehen.

Eigenschaft "IdDatenlieferung"

- definiert die für den meldenden Krankenversicherungsträger eindeutige Id der Datenlieferung
- diese wird durch den Krankenversicherungsträger festgelegt
- Optional: Nein
- Typ: String
- minLength: 3
- maxLength: 40
- Schutz: Der Wert wird in Plaintext übertragen

Eigenschaft "Meldungen"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "VitalStatusNotification"
- Felder innerhalb eines Datensatzes:
 - IdDatensatz
 - definiert die eindeutige Id des Datensatzes innerhalb dieser Datenlieferung
 - diese wird durch den Krankenversicherungsträger festgelegt
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - IdVersicherter
 - definiert den unveränderlichen und eindeutigen Identifikator eines Versicherten. Dieser Identifikator entspricht in der Regel dem unveränderbaren Teil der Krankenversicherungsnummer (KVNR) nach § 290 des Fünften Buches Sozialgesetzbuch (SGB V). Alternativ besteht die Möglichkeit für eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer entsprechend §17 Abs. 4 IRegG.
 - Optional: Nein
 - Format: Byte
 - Typ: String
 - Schutz: Als personenidentifizierendes Merkmal muss dieser Wert für die Vertrauensstelle verschlüsselt werden.
 - Prüfungen:
 - bei Angabe einer Krankenversicherungsnummer:
 - es wird geprüft, dass die Angabe einer gültigen KVNR entspricht (Regex: $^{[A-Z]\{1\}[0-9]\{9\}}$, sowie Prüfung der Prüfsumme)
 - es wird geprüft, dass die Angabe nicht der für Patienten ohne deutsche Krankenversicherung reservierten KVNR entspricht ([1](#))
 - es wird geprüft, dass diese KVNR im jeweiligen Kontext benutzt werden darf ([2](#))
 - bei Angabe einer anderen eindeutigen, unveränderbaren und nach einheitlichen Kriterien gebildeten Identifikationsnummer:
 - es wird geprüft, ob die Angabe einem bekannten, festgelegten Schema entspricht, einschließlich der Prüfung einer Prüfsumme
 - Sterbedatum
 - Sterbedatum einer Person, die ein meldepflichtiges Implantat trägt. Das Format in unverschlüsselter Form: YYYY-MM-DD (vierstellige Jahreszahl, zweistellige Monatszahl und zweistellige Tageszahl getrennt durch "-". Dies entspricht dem XML-Schema eines Datums, siehe https://www.w3schools.com/XML/schema_dtypes_date.asp. Die Angabe der Zeitzone ist nicht notwendig.
 - Optional: Nein. Wenn diese Angabe nicht anwendbar ist (da der VitalStatus des Versicherten "Lebend" oder "Unbekannt" ist, so muss dieses Feld durch den Ersatzwert "---N/A----" anstelle des Todesdatums befüllt werden (als Klartext vor der Verschlüsselung). Dieses Feld muss gesetzt sein, da ansonsten indirekt auf den Vitalstatus des Eintrags geschlossen werden könnte.
 - Format: Byte
 - Typ: String
 - Schutz: Als medizinisches Datum muss dieser Wert für die Registerstelle verschlüsselt werden
 - Prüfungen (diese werden asynchron bei der Registerstelle nach der Entschlüsselung vorgenommen)
 - Die Registerstelle prüft, ob die Angabe einem gültigen Datum mit erwartetem Format yyyy-mm-dd oder dem definierten Ersatzwert "---N/A----" entspricht.
 - Vitalstatus
 - aktueller Vitalstatus des Versicherten
 - Optional: Nein
 - Format: Byte
 - Typ: String
 - Schutz: Als medizinisches Datum muss dieser Wert für die Registerstelle verschlüsselt werden
 - Prüfungen (diese werden asynchron bei der Registerstelle nach der Entschlüsselung vorgenommen)
 - es wird geprüft, ob die vorhandene Angabe einem der folgenden erlaubten Werte entspricht: "01"=Lebend, "02"=Verstorben, "03"=Unbekannt

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Rückgabe

Nach erfolgreicher Annahme und rudimentärer Prüfung der Daten wird der HTTP-Statuscode 200 zurückgegeben. Der (Content) der Antwort ist dabei leer.

Fehlerfälle

Fehler, die bei der Annahme oder Prüfung eines an das API der Vertrauensstelle IRD gesendeten Datenpaketes auftreten, werden auf die gesamte Meldung ausgewertet. Auftretende Fehler werden dem meldenden Krankenversicherungsträger sofort synchron zurückgemeldet.

Weiterhin erfolgt nach der Annahme der Daten innerhalb der Vertrauensstelle und im Implantateregister eine Verarbeitung der Daten. Auftretende Fehler bei der Verarbeitung werden dem meldenden Krankenversicherungsträger asynchron über einen gesonderten Abruf der Daten zur Verfügung gestellt. Die Abfrage von Fehlern wird durch einen Polling-Mechanismus des meldenden Krankenversicherungsträgers erreicht. Wenn zu Datensätzen keine Fehlermeldungen zum Abruf zur Verfügung stehen, so bedeutet es für den Krankenversicherungsträger, dass keine Fehler bei der Verarbeitung aufgetreten sind.

Synchrone Fehlermeldungen

Bei Fehlern, die sich auf die gesamte Meldung beziehen, findet **keine** Verarbeitung der Datensätze statt.

Fehler, die bei der Kommunikation mit der API oder bei der Verarbeitung von Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurückgemeldet.

Liste der Fehlerfälle:

- StatusCode 400: Dieser Fehler wird zurückgemeldet, wenn die Nachricht fehlerhaft war, weil Pflichtangaben fehlen oder Angaben nicht plausibel sind (siehe dazu Prüfungen von Feldern). Es werden in diesem Fehlerfall keine erweiterten Fehlerbeschreibungen zurückgemeldet.
- StatusCode 401: Dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.
- StatusCode 403: Dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war. Dies beinhaltet auch die Verwendung von produktiven Daten in der Referenz-Umgebung.
- StatusCode 415: Dieser Fehler wird zurückgemeldet, wenn ein falscher Content-Type gesendet wurde.
- StatusCode 500: Dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Request

```
{
  "IdDatenlieferung": "2024-01",
  "Meldungen": [
    {
      "IdDatensatz": "8-0000001",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBZXZJ0IGRlcyBJZFZlcnNpY2hlcmlcRlcg==",
      "Vitalstatus": "SWNoIGJpbiBkZXIgdGVyc2NobMO8c3NlbHRlIFZpdGFsc3RhdHVz",
      "Sterbedatum": "SWNoIGJpbiBkYXNldGVyc2NobMO8c3NlbHRlIFN0ZXJiZWVhdHVt"
    },
    {
      "IdDatensatz": "8-0000002",
      "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlnQgZGVzIElkVmVyc2lj",
      "Vitalstatus": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlnQgZGVzIFZpdGFsc3RhdHVz",
      "Sterbedatum": "SWNoIGJpbiBlaW4gYW5kZXJlcyB2ZXJzY2hsw7xzc2VsdGVzIFN0ZXJiZWVhdHVt"
    }
  ],
  "Signatur": "SWNoIGJpbiBkaWUgU2lnbmF0dXJlZG1lIGdlc2FtdGVuIFZpdGFsc3RhdHVzLU1lbGR1bmdlbi"
}
```

Signaturbildung der Eingangsdaten

Die Signatur, die vom Sender an den Eingangsdaten mitzusenden ist, bezieht sich auf die (bereits verschlüsselten) Vitalstatus-Meldungen. Bei der Signatur werden die Daten in einer konkatenierten Form zur Signaturberechnung und -prüfung betrachtet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "IdDatenlieferung": "2024-01",
  "Meldungen": [
    {
      "IdDatensatz": "8-0000001",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBZXZJ0IGRlcyBJZFZlcnNpY2hlcuRlcnRlcg==",
      "Vitalstatus": "SWNoIGJpbiBkZXIgdGVyc2NobMO8c3NlbHRlIFZpdGFsc3RhdHVz",
      "Sterbedatum": "SWNoIGJpbiBkYXNpdGVyc2NobMO8c3NlbHRlIFN0ZXJiZWVhdHVt"
    },
    {
      "IdDatensatz": "8-0000002",
      "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIElkVmVyc2l1j",
      "Vitalstatus": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIFZpdGFsc3RhdHVz",
      "Sterbedatum": "SWNoIGJpbiBlaW4gYW5kZXJlcyB2ZXJzY2hsw7xzc2VsdGVzIFN0ZXJiZWVhdHVt"
    }
  ]
}
```

entsteht folgendes Ergebnis:

<Wert der Eigenschaft IdDatenlieferung> + | + <Wert des IdDatensatzes Datensatz 1> + | + <Wert de
Hexadezimale Darstellung
32 30 32 34 2D 30 31 7C 38 2D 30 30 30 30 30 30 31 7C 49 63 68 20 62 69 6E 20 65 69 6E 20 76 65 7

Weitere Informationen zu der Bildung der Signatur finden sich unter dem Punkt [Signierung von Eingangsdaten](#).

API für den Abruf von Anfragen zu Vitalstatusmeldungen für im Implantateregister erfasste Versicherte

[GET /notify/api/v1/vitalstatusnotification/requests](#)

Diese Schnittstelle ist für den Abruf von Anfragen zu Vitalstatusmeldungen für im Implantateregister erfasste Versicherte ("Implantatträger") zu verwenden.

Eingabedaten:

keine

Rückgabe

Wenn keine Daten zum Abruf vorhanden sind, wird der StatusCode 204 zurückgegeben.

Wenn Daten zum Abruf vorhanden sind, wird der StatusCode 200 zurückgegeben. Als Content der Antwort wird die Liste der Versicherten zurückgegeben, zu denen eine Vitalstatusmeldung angefragt wird. Elemente innerhalb dieser Liste sind vom Typ "RequestsForVitalStatusNotificationBundle".

Eigenschaft "Meldungen"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "RequestForVitalStatusNotification"
- Felder innerhalb eines Datensatzes:
 - IdVersicherter

- definiert den unveränderlichen und eindeutigen Identifikator eines Versicherten. Dieser Identifikator entspricht in der Regel dem unveränderbaren Teil der Krankenversicherungsnummer (KVNR) nach § 290 des Fünften Buches Sozialgesetzbuch (SGB V). Alternativ besteht die Möglichkeit für eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer entsprechend §17 Abs. 4 IRegG.
- Optional: Nein
- Format: Byte
- Typ: String
- Schutz: Als personenidentifizierendes Merkmal ist dieser Wert für den Krankenversicherungsträger mit dem für diese Übertragung erzeugten Sitzungsschlüssel (siehe [Verschlüsselung der Ausgangsdaten](#)) verschlüsselt

Eigenschaft "SessionKey"

- symmetrischer 32 Bytes langer Schlüssel gültig für diese Übertragung (Session) für die Verschlüsselung der IdVersicherter
- Optional: Nein
- Format: Byte
- Typ: String

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Fehlerfälle

Synchrone Fehlermeldungen

Fehler, die beim Abruf der Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurückgemeldet. Liste der Fehlerfälle:

- StatusCode 401: dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.
- StatusCode 403: dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war.
- StatusCode 500: dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Response

```
{
  "Anfragen" :
  [
    {
      "IdVersicherter" : "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2h1cnRlcnRlcg",
    },
    {
      "IdVersicherter" : "SWNoIGJpbiBlaW4gYW5kZkZlciB2ZXJzY2hsw7xzc2VsdGVyIFd1cnQgZGVzIElkVmVyc2"
    }
  ],
  "SessionKey" : "ZmYwNTBhOGRmZjg1Zjc3NzBlYjc5YTZjYzgwODlkYTEyNWUwZjhhN2MwYjVlNzc3NTg3ODM3Y2UyZDNkZ"
  "Signatur" : "SWNoIGJpbiBkaWUgU2lnbmF0dXIgw7xiZXIgcGllIGdlc2FtdGVuIFZpdGFsc3RhHVzLU1lbGR1bmdlbiB"
}
```

Signierung der Ausgangsdaten

Die Antworten, die durch die Aktion zurückgegeben werden, beinhalten neben der Liste der Daten auch die Signatur. Die Signatur wird dabei über die konkatenierten Werte innerhalb dieser Antwort gebildet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "Anfragen" :
  [
```

```

    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2hlcuRlcg
    },
    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlnQgZGVzIElkVmVyc2
    }
  ],
  "SessionKey": "ZmYwNTBhOGRmZjg1Zjc3NzBlYjc5YTZjYzgwODlkYTEyNWUwZjhhN2MwYjVlNzc3NTg3ODM3Y2UyZD
}

```

entsteht folgendes Ergebnis:

<SessionKey> + | + <Wert IdVersicherter1> + | + <Wert IdVersicherter2>

Hexadezimale Darstellung

FF 05 0A 8D FF 85 F7 77 0E B7 9A 6C C8 08 9D A1 25 E0 F8 A7 C0 B5 E7 77 58 78 37 CE 2D 3D D6 60 7

Weitere Informationen finden sich unter dem Punkt [Signierung der Ausgangsdaten](#).

API für den Abruf der Benachrichtigungen über die Anonymisierung der Daten von im Implantateregister erfassten Versicherten

[GET /notify/api/v1/anonymizationnotifications](#)

Entsprechend IRegG benachrichtigt das IRD die Krankenversicherungsträger bei der Anonymisierung der Daten zu einem im Implantateregister erfassten Versicherten ("Implantatträger"). Diese Schnittstelle ist für den Abruf dieser Benachrichtigungen zu verwenden.

Eingabedaten:

keine

Rückgabe

Wenn keine Daten zum Abruf vorhanden sind, wird der StatusCode 204 zurückgegeben.

Wenn Daten zum Abruf vorhanden sind, wird der StatusCode 200 zurückgegeben. Als Content der Antwort wird die Liste der Versicherten zurückgegeben, zu denen eine Anonymisierung innerhalb der Vertrauensstelle und der Registerstelle vorgenommen wurden. Elemente innerhalb dieser Liste sind vom Typ "AnonymizationNotificationsBundle".

Eigenschaft "Anonymisierungen"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "AnonymizationNotification"
- Felder innerhalb eines Datensatzes:
 - IdVersicherter
 - definiert den unveränderlichen und eindeutigen Identifikator eines Versicherten. Dieser Identifikator entspricht in der Regel dem unveränderbaren Teil der Krankenversicherungsnummer (KVNR) nach § 290 des Fünften Buches Sozialgesetzbuch (SGB V). Alternativ besteht die Möglichkeit für eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer entsprechend §17 Abs. 4 IRegG.
 - Optional: Nein
 - Format: Byte
 - Typ: String
 - Schutz: Als personenidentifizierendes Merkmal ist dieser Wert für den Krankenversicherungsträger mit dem Sitzungsschlüssel (siehe [Verschlüsselung der Ausgangsdaten](#))
 - verschlüsselt

Eigenschaft "SessionKey"

- symmetrischer 32 Bytes langer Schlüssel gültig für diese Übertragung (Session) für die Verschlüsselung der IdVersicherter

- Optional: Nein
- Format: Byte
- Typ: String

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Fehlerfälle

Synchrone Fehlermeldungen

Fehler, die beim Abruf der Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurück gemeldet. Liste der Fehlerfälle:

- StatusCode 401: dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.
- StatusCode 403: dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war.
- StatusCode 500: dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Response

```
{
  "Anonymisierungen":
  [
    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2h1cnRlcg",
    },
    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIElkVmVyc2",
    }
  ],
  "Signatur": "SWNoIGJpbiBkaWUgU2lnbmF0dXIgw7xiZXIgzG1lIGdlc2FtdGVuIFZpdGFsc3RhdHVzLU1lbGR1bmdlbiB"
}
```

Signierung der Ausgangsdaten

Die Antworten, die durch die Aktion zurückgegeben werden, beinhalten neben der Liste der Daten auch die Signatur. Die Signatur wird dabei über die konkatenierten Werte innerhalb dieser Antwort gebildet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "Anonymisierungen":
  [
    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2h1cnRlcg",
    },
    {
      "IdVersicherter": "SWNoIGJpbiBlaW4gYW5kZXJlciB2ZXJzY2hsw7xzc2VsdGVyIFdlcnQgZGVzIElkVmVyc2",
    }
  ]
}
```

entsteht folgendes Ergebnis:

<Wert IdVersicherter1> + | + <Wert IdVersicherter2>

Hexadezimale Darstellung

49 63 68 20 62 69 6E 20 65 69 6E 20 76 65 72 73 63 68 6C C3 BC 73 73 65 6C 74 65 72 20 57 65 72 7

Weitere Informationen finden sich unter dem Punkt [Signierung der Ausgangsdaten](#).

API für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Vitalstatus-Meldungen auf Seiten der Vertrauensstelle oder der Registerstelle aufgetreten sind

[GET /notify/api/v1/vitalstatusnotification/processingerrors](#)

Diese Schnittstelle ist für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Vitalstatus-Meldungen auf Seiten der Vertrauensstelle oder der Registerstelle aufgetreten sind.

Eingabedaten:

keine

Rückgabe

Wenn keine Daten zum Abruf vorhanden sind, wird der StatusCode 204 zurückgegeben.

Wenn Daten zum Abruf vorhanden sind, wird der StatusCode 200 zurückgegeben. Als Content der Antwort wird die Liste der Datensätze zurückgegeben, bei denen es bei der asynchronen Verarbeitung von Vitalstatus-Meldungen auf Seiten der Vertrauensstelle oder der Registerstelle zu Fehlern kam. Elemente innerhalb dieser Liste sind vom Typ "VitalStatusNotificationProcessingErrorsBundle".

Eigenschaft "Fehler"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "VitalStatusNotificationProcessingErrorResult"
- Felder innerhalb eines Datensatzes:
 - IdDatenlieferung
 - definiert die für den meldenden Krankenversicherungsträger eindeutige Id der Datenlieferung des Datensatzes
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - IdDatensatz
 - definiert die eindeutige Id des Datensatzes (innerhalb der Datenlieferung)
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - Code
 - definiert den Fehler, der aufgetreten ist, und wodurch der Datensatz nicht verarbeitet werden konnte
 - Optional: Nein
 - Typ: String
 - Schutz: Der Wert wird in Plaintext übertragen

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Fehlerfälle

Synchrone Fehlermeldungen

Fehler, die beim Abruf der Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurückgemeldet. Liste der Fehlerfälle:

- StatusCode 401: dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.
- StatusCode 403: dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war.
- StatusCode 500: dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Response

```
{
  "Fehler":
  [
    {
      "IdDatenlieferung": "2024-01",
      "IdDatensatz": "8-0000001",
      "Code": "DecryptionError"
    },
    {
      "IdDatenlieferung": "2024-02",
      "IdDatensatz": "8-0000002",
      "Code": "WrongFormatIdVersicherter"
    }
  ],
  "Signatur": "SWNoIGJpbiBkaWUgU2lnbmF0dXIgw7xiZXIgzGllIGd1c2FtdGVuIFZpdGFsc3RhdHVzLU1lbGR1bmdlbiB"
}
```

Signierung der Ausgangsdaten

Die Antworten, die durch die Aktion zurückgegeben werden, beinhalten neben der Liste der Daten auch die Signatur. Die Signatur wird dabei über die konkatenierten Werte innerhalb dieser Antwort gebildet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "Fehler":
  [
    {
      "IdDatenlieferung": "2024-01",
      "IdDatensatz": "8-0000001",
      "Code": "DecryptionError"
    },
    {
      "IdDatenlieferung": "2024-02",
      "IdDatensatz": "8-0000002",
      "Code": "WrongFormatIdVersicherter"
    }
  ]
}
```

entsteht folgendes Ergebnis:

<Wert IdDatenlieferung1> + | + <Wert IdDatensatz1> + | + <Wert Fehlercode1> + <Wert IdDatenliegf

Hexadezimale Darstellung

32 30 32 34 2D 30 31 7C 38 2D 30 30 30 30 30 30 31 7C 44 65 63 72 79 70 74 69 6F 6E 45 72 72 6F 7

Weitere Informationen finden sich unter dem Punkt [Signierung der Ausgangsdaten](#).

API für die Übertragung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten

[POST /notify/api/v1/insuranceupdatenotification](#)

Diese Schnittstelle ist für die Übertragung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten zu verwenden.

Eingabedaten:

Die Daten, die von der API für die Übertragung von Vitalstatusmeldungen von im Implantateregister erfassten Versicherten erwartet und verarbeitet werden, sind in der OpenApi-Spezifikation unter der Adresse https://xml.ir-d.de/rst/download/vst/OpenAPI_InsuranceUpdateService-API_V1.json unter dem Typ "VitalStatusNotificationBundle" beschrieben. Die Dokumente zum API werden zukünftig unterhalb der URL <https://xml.ir-d.de/rst/download/vst/> zum Download bereitstehen.

Eigenschaft "IdDatenlieferung"

- definiert die für den meldenden Krankenversicherungsträger eindeutige Id der Datenlieferung
- diese wird durch den Krankenversicherungsträger festgelegt
- Optional: Nein
- Typ: String
- minLength: 3
- maxLength: 40
- Schutz: Der Wert wird in Plaintext übertragen

Eigenschaft "Meldungen"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "InsuranceUpdateNotification"
- Felder innerhalb eines Datensatzes:
 - IdDatensatz
 - definiert die eindeutige Id des Datensatzes innerhalb dieser Datenlieferung
 - diese wird durch den Krankenversicherungsträger festgelegt
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - IdVersicherter
 - definiert den eindeutigen Identifizierer des Versicherten. Dieser Identifizierer ist der unveränderbare Teil der Krankenversicherungsnummer (KVNR) nach § 290 des Fünften Buches Sozialgesetzbuch (SGB V) bzw. eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer entsprechend §17 Abs. 4 IRegG.
 - Optional: Nein
 - Format: Byte
 - Typ: String
 - Schutz: Der Wert muss für die Vertrauensstelle verschlüsselt werden
 - Prüfungen:
 - bei Angabe einer Krankenversicherungsnummer:
 - es wird geprüft, dass die Angabe einer gültigen KVNR entspricht (Regex: `^[A-Z]{1}[0-9]{9}$` sowie Prüfung der Prüfsumme)
 - es wird geprüft, dass die Angabe nicht der für Patienten ohne deutsche Krankenversicherung reservierten KVNR entspricht (¹)
 - es wird geprüft, dass diese KVNR im jeweiligen Context benutzt werden darf (²)
 - bei Angabe einer anderen eindeutigen, unveränderbaren und nach einheitlichen Kriterien gebildete Identifikationsnummer:
 - es wird geprüft, ob die Angabe einem bekannten, festgelegten Schema entspricht sowie eine Prüfung der Prüfsumme
 - IdVersicherterAlt
 - definiert den eindeutigen Identifizierer des Versicherten bei der vorherigen Versicherung. Dieser Identifizierer ist der unveränderbare Teil der Krankenversicherungsnummer (KVNR) nach § 290 des Fünften Buches Sozialgesetzbuch

(SGB V) bzw. eine andere eindeutige, unveränderbare und nach einheitlichen Kriterien gebildete Identifikationsnummer entsprechend §17 Abs. 4 IRegG.

- Optional: Ja - Muss nur gesetzt werden, wenn der Versicherte bei seiner Vorversicherung unter einer anderen Nummer versichert war
- Format: Byte
- Typ: String
- Schutz: Der Wert muss für die Vertrauensstelle verschlüsselt werden
- Prüfungen:
 - bei Angabe einer Krankenversicherungsnummer:
 - es wird geprüft, dass die Angabe einer gültigen KVNR entspricht (Regex: `^[A-Z]{1}[0-9]{9}$`, sowie Prüfung der Prüfsumme)
 - es wird geprüft, dass die Angabe nicht die der definierten KVNR eines Patienten ohne deutsche Krankenversicherung ist ⁽¹⁾
 - es wird geprüft, dass diese KVNR im jeweiligen Context benutzt werden darf ⁽²⁾
 - bei Angabe einer anderen eindeutigen, unveränderbaren und nach einheitlichen Kriterien gebildeten Identifikationsnummer:
 - es wird geprüft, ob die Angabe einem bekannten, festgelegten Schema entspricht sowie eine Prüfung der Prüfsumme
- IkAlt
 - definiert die Haupt-IK (Institutionskennzeichen) der Vorversicherung
 - Optional: Nein
 - Format: Byte
 - Typ: String
 - Schutz: Der Wert muss für die Vertrauensstelle verschlüsselt werden
 - Prüfungen:
 - es wird geprüft, dass die Angabe einem validen Institutionskennzeichen entspricht (Regex: `^[0-9]{9}$` sowie Prüfung der Prüfsumme)

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Rückgabe

Nach erfolgreicher Verarbeitung der Daten wird der StatusCode 200 zurückgegeben. Als Content der Antwort wird die signierte Transfernummer zurückgegeben (Typ: "SignedTransferNumber").

Fehlerfälle

Fehler, die bei der Annahme oder Prüfung eines Datenpaketes an das API der Vertrauensstelle IRD auftreten, werden auf die gesamte Meldung ausgewertet. Auftretende Fehler werden dem meldenden Krankenversicherungsträger sofort synchron zurückgemeldet.

Weiterhin erfolgt nach der Annahme der Daten innerhalb der Vertrauensstelle eine Verarbeitung der Daten. Auftretende Fehler bei der Verarbeitung werden dem meldenden Krankenversicherungsträger asynchron über einen gesonderten Abruf der Daten zur Verfügung gestellt. Die Abfrage von Fehlern wird durch ein Polling-Mechanismus des meldenden Krankenversicherungsträgers erreicht. Wenn zu Datensätzen keine Fehlermeldungen zum Abruf zur Verfügung stehen, so bedeutet es für den Krankenversicherungsträger, dass keine Fehler bei der Verarbeitung aufgetreten sind.

Fehler, die bei der Kommunikation mit der API oder bei der Verarbeitung von Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurückgemeldet.

Liste der Fehlerfälle:

- StatusCode 400: dieser Fehler wird zurückgemeldet, wenn die Nachricht fehlerhaft war, weil Pflichtangaben fehlen oder Angaben nicht plausibel sind (siehe dazu Prüfungen von Feldern). Es werden in diesem Fehlerfall keine erweiterten Fehlerbeschreibungen zurückgemeldet.
- StatusCode 401: dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.

- StatusCode 403: dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war. Dies beinhaltet auch die Verwendung von produktiven Daten in der Referenz-Umgebung.
- StatusCode 415: dieser Fehler wird zurückgemeldet, wenn ein falscher Content-Type gesendet wurde.
- StatusCode 500: dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Request

```
{
  "IdDatenlieferung": "2024-001",
  "Meldungen": [
    {
      "IdDatensatz": "8-0000001",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2hlc nRl cg==",
      "IkAlt": "SWNoIGJpbiBlaW4gdmVyc2ljaGVydGVyIFdlnQgZGVyIElrQWx0"
    },
    {
      "IdDatensatz": "8-0000002",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElkVmVyc2lj",
      "IdVersicherterAlt": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElkVmVyc",
      "IkAlt": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElrQWx0"
    }
  ],
  "Signatur": "SWNoIGJpbiBkaWUgU2lnbmF0dXlwaW7xiZlXlGZGllIGdlc2FtdGVuIFZpdGFsc3Rh dHVzLU1lbGRlbmdlbi"
}
```

Signaturbildung der Eingangsdaten

Die Signatur, die vom Sender an den Eingangsdaten mitzusenden ist, bezieht sich auf die (bereits verschlüsselten) Meldungen zu den Wechseln der Versicherungen. Bei der Signatur werden die Daten in einer konkatenierten Form zur Signaturberechnung und -prüfung betrachtet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "IdDatenlieferung": "2024-001",
  "Meldungen": [
    {
      "IdDatensatz": "8-0000001",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyBJZFZlcnNpY2hlc nRl cg==",
      "IkAlt": "SWNoIGJpbiBlaW4gdmVyc2ljaGVydGVyIFdlnQgZGVyIElrQWx0"
    },
    {
      "IdDatensatz": "8-0000002",
      "IdVersicherter": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElkVmVyc2lj",
      "IdVersicherterAlt": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElkVmVyc",
      "IkAlt": "SWNoIGJpbiBlaW4gdmVyc2NobMO8c3NlbHRlciBXZXJ0IGRlcyB6d2VpdGVuIElrQWx0"
    }
  ]
}
```

entsteht folgendes Ergebnis:

<Wert der Eigenschaft IdDatenlieferung> + | + <Wert des IdVersicherten Datensatz 1> + | + <Wert d
 Hexadezimale Darstellung
 32 30 32 34 2D 30 30 31 7C 38 2D 30 30 30 30 30 30 31 7C 49 63 68 20 62 69 6E 20 65 69 6E 20 76 6

Weitere Informationen zu der Bildung der Signatur finden sich unter dem Punkt [Signierung von Eingangsdaten](#).

API für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten in der Vertrauensstelle aufgetreten sind

[GET /notify/api/v1/insuranceupdatenotification/processingerrors](#)

Diese Schnittstelle ist für den Abruf von Fehlern, die bei der asynchronen Verarbeitung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten in der Vertrauensstelle aufgetreten sind.

Eingabedaten:

keine

Rückgabe

Wenn keine Daten zum Abruf vorhanden sind, wird der StatusCode 204 zurückgegeben.

Wenn Daten zum Abruf vorhanden sind, wird der StatusCode 200 zurückgegeben. Als Content der Antwort wird die Liste der Datensätze zurückgegeben, bei denen es bei der asynchronen Verarbeitung von Meldungen bzgl. dem Versicherungswechsel von im Implantateregister erfassten Versicherten auf Seiten der Vertrauensstelle zu Fehlern kam. Elemente innerhalb dieser Liste sind vom Typ "InsuranceUpdateProcessingErrorsBundle".

Eigenschaft "Fehler"

- Definiert die Datensätze innerhalb dieser Datenlieferung
- Optional: Nein
- Typ: Array
- Typ der Elemente innerhalb des Arrays: "InsuranceUpdateProcessingErrorResult"
- Felder innerhalb eines Datensatzes:
 - IdDatenlieferung
 - definiert die für den meldenden Krankenversicherungsträger eindeutige Id der Datenlieferung des Datensatzes
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - IdDatensatz
 - definiert die eindeutige Id des Datensatzes (innerhalb der Datenlieferung)
 - Optional: Nein
 - Typ: String
 - minLength: 3
 - maxLength: 40
 - Schutz: Der Wert wird in Plaintext übertragen
 - Code
 - definiert den Fehler, der aufgetreten ist, und wodurch der Datensatz nicht verarbeitet werden konnte
 - Optional: Nein
 - Typ: String
 - Schutz: Der Wert wird in Plaintext übertragen

Eigenschaft "Signatur"

- definiert die Signatur über die Daten dieser Datenlieferung
- Optional: Nein
- Format: Byte
- Typ: String

Fehlerfälle

Synchrone Fehlermeldungen

Fehler, die beim Abruf der Daten auftreten können, werden der meldenden Einrichtung in Form von HTTP-Statuscodes zurückgemeldet. Liste der Fehlerfälle:

- StatusCode 401: dieser Fehler wird zurückgemeldet, wenn kein Authentisierungstoken vorhanden ist (s. [Authentisierung und Authentifizierung](#)) oder die Authentifizierung an der Schnittstelle fehlgeschlagen ist.
- StatusCode 403: dieser Fehler wird zurückgemeldet, wenn die Anfrage nicht autorisiert war.
- StatusCode 500: dieser Fehler wird zurückgemeldet, wenn ein interner Fehler bei der Verarbeitung aufgetreten ist.

Beispiel Response

```
{
  "Fehler":
  [
    {
      "IdDatenlieferung": "2024-01",
      "IdDatensatz": "8-0000001",
      "Code": "DecryptionError"
    },
    {
      "IdDatenlieferung": "2024-02",
      "IdDatensatz": "8-0000002",
      "Code": "WrongFormatIdVersicherter"
    }
  ],
  "Signatur": "SWNoIGJpbIbkaWUgU2lnbmF0dXIgw7xiZXIgzGllIGdlc2FtdGVuIFZpdGFsc3RhHVzLU1lbGR1bmdlbiB"
}
```

Signierung der Ausgangsdaten

Die Antworten, die durch die Aktion zurückgegeben werden, beinhalten neben der Liste der Daten auch die Signatur. Die Signatur wird dabei über die konkatenierten Werte innerhalb dieser Antwort gebildet.

Für beispielsweise ein JSON-Fragment mit diesen Daten:

```
{
  "Fehler":
  [
    {
      "IdDatenlieferung": "2024-01",
      "IdDatensatz": "8-0000001",
      "Code": "DecryptionError"
    },
    {
      "IdDatenlieferung": "2024-02",
      "IdDatensatz": "8-0000002",
      "Code": "WrongFormatIdVersicherter"
    }
  ]
}
```

entsteht folgendes Ergebnis:

<Wert IdDatenlieferung1> + | + <Wert IdDatensatz1> + | + <Wert Fehlercode1> + <Wert IdDatenliegf

Hexadezimale Darstellung

32 30 32 34 2D 30 31 7C 38 2D 30 30 30 30 30 30 31 7C 44 65 63 72 79 70 74 69 6F 6E 45 72 72 6F 7

Weitere Informationen finden sich unter dem Punkt [Signierung der Ausgangsdaten](#).

Übergreifende Beispiele

In diesem Abschnitt wird anhand von Beispielen und Beispielaufrufen gezeigt, wie Daten, die an die Vertrauensstelle gesendet werden, verschlüsselt und signiert werden müssen. Auch wird hier beschrieben, wie die Registrierung an dem Dienst der Vertrauensstelle durchgeführt werden muss sowie die Authentisierung und Authentifizierung an dem Dienst. Weiter wird hier auch gezeigt, wie Daten, die verschlüsselt von der Vertrauensstelle an einen Krankenversicherungsträger gesendet werden, entschlüsselt werden können sowie die Signatur eines Datenpaketes geprüft werden kann.

Verschlüsselung der Eingangsdaten

Bei der Verschlüsselung der Eingangsdaten wird sich an dem Konzept der gematik des E-Rezeptes angelehnt. Eine Beispiel-Implementierung zur Erstellung der verschlüsselten Datenfelder findet sich unter der Adresse <https://github.com/gematik/api-erp/blob/master/samples/VAUSimpleExample/VAU.cs> (C#) bzw. unter <https://github.com/gematik/api-erp/blob/master/samples/snippets/VAUClientCrypto.java> (Java).

Bei der Verschlüsselung wird dabei für jedes zu verschlüsselnde Datenfeld ein eigenes temp. ECC-Schlüsselpaar gebildet. Dieses Schlüsselpaar dient dann als Eingangsparameter des ECDHBasicAgreement, mit dessen Hilfe gegen das öffentliche Zertifikat des Empfängers (also die Vertrauensstelle oder die Registerstelle) das temp. SharedSecret gebildet wird. Dieses temporäre gemeinsame Geheimnis (shared secret) wird als Eingangsparameter für die Schlüsselableitung (HKDF) verwendet. Als Info für die HKDF wird der fixe Wert "VST-IRD-Transport" verwendet. Der Wert, der durch die Schlüsselableitung gebildet wird, dient als Schlüssel für die Verschlüsselung des Eingangswertes mit dem Cipher AES-256-GCM.

Das Schlüsselmaterial der Vertrauensstelle lässt sich über einen gesonderten Downloadpunkt abrufen. Siehe hierzu [Downloadpunkt für den Abruf von Schlüsselmaterial der Vertrauensstelle](#).

Nach Durchführung der Verschlüsselungsoperation wird der durch den Cipher gebildete Wert zusammen mit den öffentlichen Kurvenpunkten des temp. Schlüsselpaares als Wert innerhalb des JSON-Datenobjektes verwendet.

Ein Beispiel eines Datenkonstruktes nach Durchführung der Verschlüsselungsoperation und Bildung des zu übertragenen Wertes sieht wie folgt aus (hexadezimale Darstellung):

```
01 1E F6 00 88 48 EF 2D 4E AE 34 C3 5A 8A 16 8D 76 59 C0 84 F0 E1 EA DF 17 A5 33 3C B5 CF 4B B6 6
```

Das erste Byte bezeichnet dabei die Version. Diese ist nach der derzeitigen Implementierung "1". Die nächsten 32 Bytes definieren die X-Koordinate des temp. gebildeten Schlüsselpaares. Die nächsten 32 Bytes definieren die Y-Koordinate des temp. gebildeten Schlüsselpaares. Die nächsten 12 Bytes definieren den IV, der bei der Verschlüsselung AES-GCM verwendet wurde. Die folgenden, letzten Bytes in dem Konstrukt definieren den AES-GCM-verschlüsselten Wert inklusi

⚠ Für jeden verschlüsselten Wert muss ein eigenes, temporäres Schlüsselpaar gebildet und verwendet werden. Nach der Verschlüsselung eines Wertes kann dieses Schlüsselpaar wieder verworfen werden.

⚠ Für jede AES-GCM-Verschlüsselungsoperation muss ein eigener, zufällig gewürfelter IV gebildet werden.

Signierung der Eingangsdaten

Die Signierung von Daten mit dem Schlüsselmaterial der TI ist nur mithilfe einer Aktion über den Konnektor bzw. des Basis-/KTR-Consumer möglich. Dazu wird ein entsprechender Endpunkt des Konnektors bzw. des Basis-/KTR-Consumer angesprochen, bei dem dann die aufgerufene SOAP-Aktion die eigentliche Arbeit übernimmt.

Request

URI	https://192.168.x.y/Konnektorservice
Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/SignatureService/v7.4#SignDocument"
Payload	<pre><soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns4="http://ws.gematik.de/conn/SignatureService/v7.4" xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"> <soap:Body> <ns4:SignDocument> <ns2:CardHandle>CARDHANDLE</ns2:CardHandle> <ns3:Context> <ns2:MandantId>MANDANT_ID</ns2:MandantId> <ns2:ClientSystemId>CLIENTSYSTEM_ID</ns2:ClientSystemId> <ns2:WorkplaceId>WORKPLACE_ID</ns2:WorkplaceId> </ns3:Context> </ns4:SignDocument> </soap:Body> </soap:Envelope></pre>

```

        <ns2:UserId>USER_ID</ns2:UserId>
    </ns3:Context>
    <ns4:TvMode>NONE</ns4:TvMode>
    <ns4:JobNumber>SIG-001</ns4:JobNumber>
    <ns4:SignRequest RequestID='RequestID'>
        <ns4:OptionalInputs>
            <dss:SignatureType>urn:ietf:rfc:5652</dss:SignatureType>
            <ns4:IncludeEContent>true</ns4:IncludeEContent>
        </ns4:OptionalInputs>
        <ns4:Document ShortText='Dies ist eine zu signierende Nachricht' ID='Req'
            <dss:Base64Data>VGZvdAo=</dss:Base64Data>
        </ns4:Document>
        <ns4:IncludeRevocationInfo>true</ns4:IncludeRevocationInfo>
    </ns4:SignRequest>
</ns4:SignDocument>
</soap:Body>
</soap:Envelope>

```

i In `<dss:Base64Data></dss:Base64Data>` befinden sich die zu signierenden Daten in Base64-Kodierung. Die Signatur ist eine detached nonQES - Signatur (nicht qualifizierte elektronische Signatur), die mithilfe des Signatur-Zertifikats auf der SMC-B-Karte gebildet wird. Da der Content separat in dem Datenpaket gesendet wird, wird dieser nicht in die Signatur eingebettet (`<ns4:IncludeEContent>>false</ns4:IncludeEContent>`). Der Signatortyp muss CMS sein (urn:ietf:rfc:5652). Es wird eine ECDSA-Signatur benötigt.

Für den Zugriff auf die SMC-B für die Signaturerstellung muss die SMC-B durch die PIN freigeschaltet werden. Siehe hierzu [Freigabe der SMC-B durch Eingabe der PIN](#).

Generelle Informationen zu den Schnittstellen des Konnektors bzw. Basis-/KTR-Consumer

Die korrekten Adressen der Endpunkte der Services des Konnektors bzw. Basis/KTR-Consumer sowie die zu verwendenden Versionen der Services lassen sich mithilfe der Herstellerinformationen bestimmen.

Die gematik-Spezifikation mit erweiterten Informationen lassen sich unter den Dokumenten "gemSpec_Basis_KTR_Consumer" bzw. "gemSpec_Kon" und der "gemSpec_Krypt" unter der Seite <https://fachportal.gematik.de/dokumentensuche> abrufen.

Generelle Informationen zu Parametern in SOAP-Requests

- Die Mandantenkonfiguration in den Parametern `<ns3:Context></ns3:Context>` zu Mandant, ClientSystem, Workplace und User muss entsprechend der festgelegten Werte des jeweiligen Krankenversicherungsträgers angepasst werden.
- Das Handle der zu verwendenden Karte in `<ns2:CardHandle></ns2:CardHandle>` referenziert die zu verwendene Karte im Kartenterminal. Siehe hierzu [Abfrage der Karten des Kartenterminals](#).

Prüfung eines TI-Zertifikates

Request

Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/CertificateService/v6.0#VerifyCertificate"
Payload	<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0"


```

xmlns:ns4="http://ws.gematik.de/conn/CertificateServiceCommon/v2.0"
xmlns:ns5="http://ws.gematik.de/conn/CertificateService/v6.0">
<soap:Body>
  <ns5:VerifyCertificate>
    <ns3:Context>
      <ns2:MandantId>_</ns2:MandantId>
      <ns2:ClientSystemId>_</ns2:ClientSystemId>
      <ns2:WorkplaceId>_</ns2:WorkplaceId>
    </ns3:Context>
    <ns4:X509Certificate>MIIFEjCCA/qgAwIBAgIHAXaCPYBSozANBgkqhkiG9w0BAQsFADCBmjEI
  </ns5:VerifyCertificate>
</soap:Body>
</soap:Envelope>

```

Response

Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" <soap:Body> <ns4:VerifyCertificateResponse xmlns:ns9="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:ns8="urn:oasis:names:tc:dss:1.0:core:schema" xmlns:ns7="http://www.w3.org/2000/09/xmldsig#" xmlns:ns6="http://ws.gematik.de/conn/CertificateServiceCommon/v2.0" xmlns:ns5="http://ws.gematik.de/tel/error/v2.0" xmlns:ns4="http://ws.gematik.de/conn/CertificateService/v6.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0"> <ns2:Status> <ns2:Result>OK</ns2:Result> </ns2:Status> <ns4:VerificationStatus> <ns4:VerificationResult>VALID</ns4:VerificationResult> </ns4:VerificationStatus> <ns4:RoleList> <ns4:Role>1.2.276.0.76.4.50</ns4:Role> </ns4:RoleList> </ns4:VerifyCertificateResponse> </soap:Body> </soap:Envelope> </pre>
---------	--

Freigabe der SMC-B durch Eingabe der PIN

Request

Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/CardService/v8.1#VerifyPin"
Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns4="http://ws.gematik.de/conn/CardServiceCommon/v2.0" xmlns:ns5="http://ws.gematik.de/conn/CardService/v8.1"> <soap:Body> <ns5:VerifyPin> <ns3:Context> <ns2:MandantId>MANDANT_ID</ns2:MandantId> <ns2:ClientSystemId>CLIENTSYSTEM_ID</ns2:ClientSystemId> <ns2:WorkplaceId>WORKPLACE_ID</ns2:WorkplaceId> </ns3:Context> </ns5:VerifyPin> </soap:Body> </soap:Envelope> </pre>

```

        <ns2:CardHandle>CARDHANDLE</ns2:CardHandle>
        <ns4:PinTyp>PIN.SMC</ns4:PinTyp>
    </ns5:VerifyPin>
</soap:Body>
</soap:Envelope>

```

Abfrage der Karten des Kartenterminals

Request

URI	https://192.168.x.y/Konnektorservice
Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/EventService/v7.2#GetCards"
Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns4="http://ws.gematik.de/conn/EventService/v7.2"> <soap:Body> <ns4:GetCards> <ns3:Context> <ns2:MandantId>MANDANT_ID</ns2:MandantId> <ns2:ClientSystemId>CLIENTSYSTEM_ID</ns2:ClientSystemId> <ns2:WorkplaceId>WORKPLACE_ID</ns2:WorkplaceId> </ns3:Context> </ns4:GetCards> </soap:Body> </soap:Envelope> </pre>

Response

Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns4:GetCardsResponse xmlns:ns12="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:ns11="urn:oasis:names:tc:dss:1.0:core:schema" xmlns:ns10="http://www.w3.org/2000/09/xmldsig#" xmlns:ns9="http://ws.gematik.de/conn/CardTerminalInfo/v8.0" xmlns:ns8="http://ws.gematik.de/int/version/ProductInformation/v1.1" xmlns:ns7="http://ws.gematik.de/conn/CardService/v8.1" xmlns:ns6="http://ws.gematik.de/conn/CardServiceCommon/v2.0" xmlns:ns5="http://ws.gematik.de/tel/error/v2.0" xmlns:ns4="http://ws.gematik.de/conn/EventService/v7.2" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0"> <ns2:Status> <ns2:Result>OK</ns2:Result> </ns2:Status> <ns7:Cards> <ns7:Card> <ns2:CardHandle>CARDHANDLE</ns2:CardHandle> <ns6:CardType>SMC-B</ns6:CardType> <ns7:CardVersion> <ns7:COVersion> <ns7:Major>4</ns7:Major> <ns7:Minor>6</ns7:Minor> <ns7:Revision>0</ns7:Revision> </ns7:COVersion> <ns7:ObjectSystemVersion> <ns7:Major>4</ns7:Major> <ns7:Minor>8</ns7:Minor> <ns7:Revision>0</ns7:Revision> </pre>
---------	---

```

        </ns7:ObjectSystemVersion>
    </ns7:CardVersion>
    <ns6:Iccsn>xxx</ns6:Iccsn>
    <ns6:CtId>xxx</ns6:CtId>
    <ns6:SlotId>xx</ns6:SlotId>
    <ns7:InsertTime>xxxx</ns7:InsertTime>
    <ns7:CardHolderName>Praxis Dr. Mustermann TEST-ONLY</ns7:CardHolderName>
    <ns7:CertificateExpirationDate>xxx</ns7:CertificateExpirationDate>
</ns7:Card>
<ns7:Card>
    ...
</ns7:Card>
</ns7:Cards>
</ns4:GetCardsResponse>
</soap:Body>
</soap:Envelope>

```

Abfrage des AUT-Zertifikates der SMC-B-Karte

Request

URI	https://192.168.x.y/Konnektorservice
Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/CertificateService/v6.0#ReadCardCertificate"
Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns4="http://ws.gematik.de/conn/CertificateService/v6.0"> <soap:Body> <ns4:ReadCardCertificate> <ns2:CardHandle>CARDHANDLE</ns2:CardHandle> <ns3:Context> <ns2:MandantId>MANDANT_ID</ns2:MandantId> <ns2:ClientSystemId>CLIENTSYSTEM_ID</ns2:ClientSystemId> <ns2:WorkplaceId>WORKPLACE_ID</ns2:WorkplaceId> </ns3:Context> <ns4:CertRefList> <ns4:CertRef>C.AUT</ns4:CertRef> </ns4:CertRefList> <ns4:Crypt>ECC</ns4:Crypt> </ns4:ReadCardCertificate> </soap:Body> </soap:Envelope> </pre>

Response

Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns4:ReadCardCertificateResponse xmlns:ns9="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:ns8="urn:oasis:names:tc:dss:1.0:core:schema" xmlns:ns7="http://www.w3.org/2000/09/xmldsig#" xmlns:ns6="http://ws.gematik.de/conn/CertificateServiceCommon/v2.0" xmlns:ns5="http://ws.gematik.de/tel/error/v2.0" xmlns:ns4="http://ws.gematik.de/conn/CertificateService/v6.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0"> <ns2:Status> <ns2:Result>OK</ns2:Result> </ns2:Status> <ns6:X509DataInfoList> </pre>
---------	--

```

        <ns6:X509DataInfo>
          <ns6:CertRef>C.AUT</ns6:CertRef>
          <ns6:X509Data>
            <ns6:X509IssuerSerial>
              <ns6:X509IssuerName>CN=GEM.SMCB-CA51 TEST-ONLY,OU=Institution
              <ns6:X509SerialNumber>Seriennummer des Zertifikates</ns6:X509
            </ns6:X509IssuerSerial>
            <ns6:X509SubjectName>CN=xxx,O=xxx,L=xxx,C=DE</ns6:X509SubjectName
            <ns6:X509Certificate>Zertifikat_Base64_codiert</ns6:X509Certifica
          </ns6:X509Data>
        </ns6:X509DataInfo>
      </ns6:X509DataInfoList>
    </ns4:ReadCardCertificateResponse>
  </soap:Body>
</soap:Envelope>

```

Operation ExternalAuthenticate

Request

URI	https://192.168.x.y/Konnektorservice
Method	POST
HTTP Header	Content-Type: text/xml; charset=UTF-8 SOAPAction: "http://ws.gematik.de/conn/SignatureService/v7.4#ExternalAuthenticate"
Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0" xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0" xmlns:ns4="http://ws.gematik.de/conn/SignatureService/v7.4" xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"> <soap:Body> <ns4:ExternalAuthenticate> <ns2:CardHandle>CARDHANDLE</ns2:CardHandle> <ns3:Context> <ns2:MandantId>MANDANT_ID</ns2:MandantId> <ns2:ClientSystemId>CLIENTSYSTEM_ID</ns2:ClientSystemId> <ns2:WorkplaceId>WORKPLACE_ID</ns2:WorkplaceId> </ns3:Context> <ns4:OptionalInputs> <dss:SignatureType>urn:bsi:tr:03111:ecdsa</dss:SignatureType> </ns4:OptionalInputs> <ns4:BinaryString> <dss:Base64Data MimeType="application/octet-stream">l04FkzXlh+UBzEv5BhPg </ns4:BinaryString> </ns4:ExternalAuthenticate> </soap:Body> </soap:Envelope> </pre>

Response

Payload	<pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns5:ExternalAuthenticateResponse xmlns:ns15="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:ns14="http://ws.gematik.de/conn/CertificateServiceCommon/v2.0" xmlns:ns13="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:ns12="http://www.w3.org/2001/04/xmlenc#" xmlns:ns11="urn:oasis:names:tc:dss-x:1.0:profiles:SignaturePolicy:schema#" xmlns:ns10="http://uri.etsi.org/02231/v2#" xmlns:ns9="urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#" xmlns:ns8="http://uri.etsi.org/01903/v1.3.2#" xmlns:ns7="http://ws.gematik.de/tel/error/v2.0" </pre>
---------	---

```
xmlns:ns6="http://www.w3.org/2000/09/xmldsig#"
xmlns:ns5="http://ws.gematik.de/conn/SignatureService/v7.4"
xmlns:ns4="urn:oasis:names:tc:dss:1.0:core:schema"
xmlns:ns3="http://ws.gematik.de/conn/ConnectorContext/v2.0"
xmlns:ns2="http://ws.gematik.de/conn/ConnectorCommon/v5.0">
<ns2:Status>
  <ns2:Result>OK</ns2:Result>
</ns2:Status>
<ns4:SignatureObject>
  <ns4:Base64Signature Type="urn:bsi:tr:03111:ecdsa">MEQCIBZU1+5f0iFAoX558/
</ns4:SignatureObject>
</ns5:ExternalAuthenticateResponse>
</soap:Body>
</soap:Envelope>
```

Nichtfunktionale Festlegungen

aktuell keine

Test

Definition von Testdaten

Anwendungstests sind entsprechend der Vorgaben der gematik nur in der dafür bereitgestellten Referenzumgebung (RU) der TI erlaubt. Die Produktivumgebung (PU) der TI darf ausschließlich mit produktiven Daten genutzt werden.

Die Regeln für die Testdaten stellen sicher, dass patienten-identifizierende Daten nicht in Testumgebungen (für IRD die Referenzumgebung der TI) verwendet werden. Hierbei sind die speziellen Regeln für die Referenzumgebung der TI und die Produktivumgebung der TI zu beachten.

Testdaten Krankenversicherthenummer (KVNR)

Der GKV-SV hat einen KVNR-Nummernkreis bereitgestellt, der ausschließlich für Tests im Rahmen des Implantatregisters Deutschland (IRD) genutzt werden darf. Die KVNRs dieses Nummernkreises werden gesichert keiner Person zugeordnet.

Definition: Nummernkreis A1111xxxxP, wobei x für eine Ziffer von 0 bis 9 und P für die Prüfziffer steht.

Testdaten Identifikationsnummer der Heilfürsorge BW

Für die Identifikationsnummer der Heilfürsorge BW stehen aktuell die [Formatinformationen zur Identifikationsnummer](#) zur Verfügung. Als Testnummer ist auf Seite 7 die TestID 02476291358 angegeben.

Regeln für die TI-Umgebungen

Referenzumgebung der TI: In der RU dürfen für das patienten-identifizierende Datum *IdVersicherter* nur Werte aus den oben beschriebenen Testdaten verwendet werden. Die Schnittstelle lehnt bei Nichtbeachtung dieser Regel die Annahme der Daten ab und gibt einen Fehlercode an den Sender zurück.

Produktivumgebung der TI: In der PU werden für das patienten-identifizierende Datum *IdVersicherter* produktive, d.h. einem Patienten zugeordnete Daten erwartet.

Aufrufe mit Werten für *IdVersicherter* aus den oben beschriebenen Testdaten werden aktuell als Ausnahme ausschließlich zum Test der korrekten Anbindung des meldenden Krankenversicherungsträgers an die Produktivumgebung der TI akzeptiert und von der Vertrauensstelle nicht verarbeitet.

Kontakt

Robert Koch-Institut
Referat VIG - Vertrauensstellen
Nordufer 20
13353 Berlin

E-Mail: Vertrauensstelle-IRD@rki.de