

Nutzung der FHIR Schnittstelle für Meldende

Die folgenden Informationen sollen Hersteller von meldenden Systemen bei einer Umsetzung unterstützen. Sie können den DEMIS-Adapter bzw. die DEMIS-Adapter-API als Referenzimplementierung nutzen. Den Sourcecode fragen Sie beim DEMIS-Support via demis-support@rki.de an.

Mitschnitt der Infoveranstaltung für Softwarehersteller vom 26. November 2020: <https://www.youtube.com/watch?v=1P9JSsNSmRU&feature=youtu.be>

Hier finden Sie die Folien der Infoveranstaltung (Anschluss an DEMIS – Primärsystemintegration.pdf):



Anschluss an DEM...mintegration.pdf

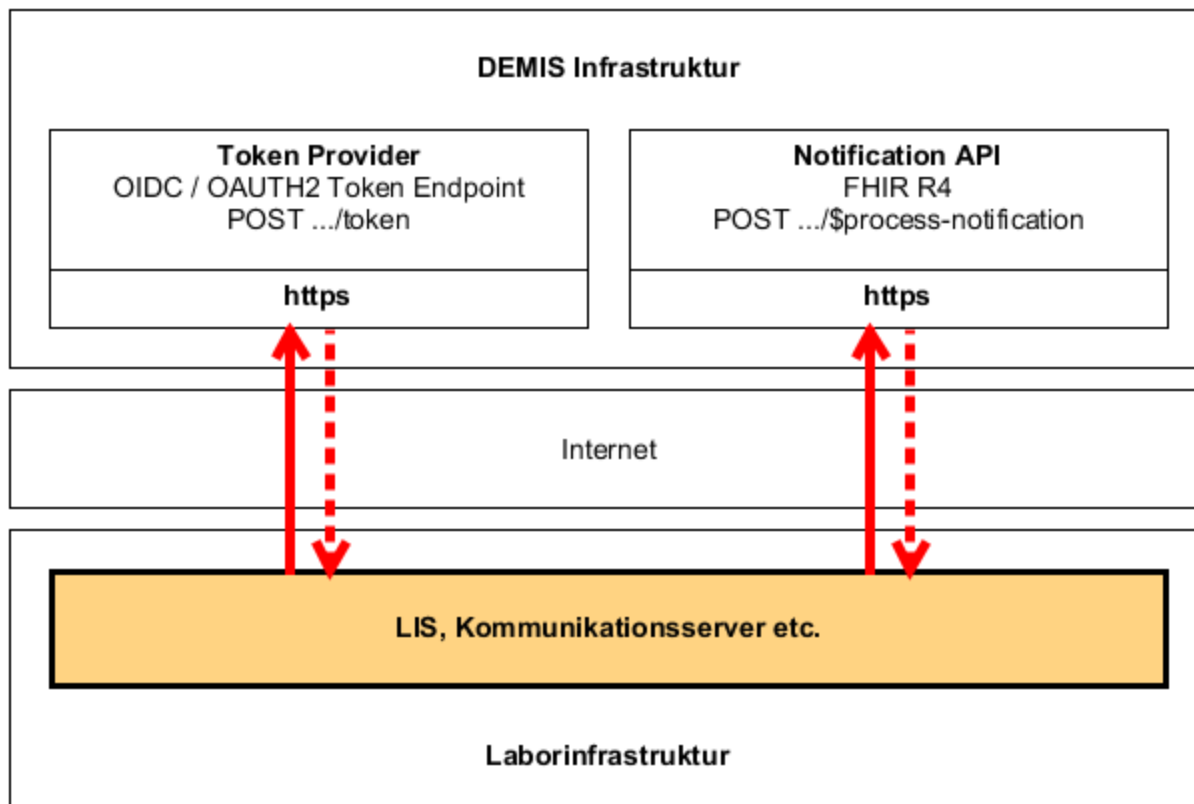
Inhalt

- [Inhalt](#)
- [Komponenten \(Überblick\)](#)
- [Abläufe \(Überblick\)](#)
- [Tokenausstellung](#)
 - [Mutual TLS](#)
 - [Zulässige Ciphersuites](#)
 - [Implementierungshinweise](#)
 - [Beispiele](#)
 - .
 - [Kommunikation mit dem Token-Endpoint](#)
 - [Endpunktadressen](#)
 - [Request-Struktur](#)
 - [Beispiele](#)
 - [Ergebnis im Erfolgsfall](#)
 - [Mögliche Fehlermeldungen](#)
 - [Struktur des Access Tokens](#)
- [Meldungsversand](#)
 - [Mutual TLS](#)
 - [Aufruf der \\$process-notification Operation](#)
 - [Endpunktadressen](#)
 - [Request-Struktur](#)
 - [Beispiele](#)
 - [Ergebnis im Erfolgsfall](#)
 - [Response-Inhalte](#)
 - [Mögliche Fehlermeldungen](#)
- [Weitere Hinweise](#)
- [Weiterführende Informationen](#)

Komponenten (Überblick)

Folgende DEMIS-Komponenten werden für die Integration der Meldefunktionalität in die Primärsysteme der Labore (LIS, Kommunikationsserver etc.) benötigt:

Token Provider – Sämtliche Zugriffe auf die zentralen Fachdienste der DEMIS-Infrastruktur (z.B. *DEMIS Notification API*) sind durch ein Access Control System geschützt. Um die für den jeweiligen Anwendungsfall benötigten Operationen aufrufen zu können, muss der Client ein standardisiertes Sicherheitstoken im Header des Aufrufs mitliefern. Diese Sicherheitstoken werden nach erfolgreicher Authentifizierung des jeweiligen Nutzers durch den *DEMIS Token Provider* ausgestellt. Die Authentifizierung der Nutzer erfolgt dabei zertifikatsbasiert. Der *DEMIS Token Provider* übernimmt ebenfalls die Verwaltung der Nutzer-Accounts.



Notification API
– Die Entgegennahme von Meldungen, die durch die Labore versendet werden, erfolgt durch die DEMIS Notification API. Diesem Dienst kommt eine ganz besondere Bedeutung zu, da hier ebenfalls die Vorverarbeitung der entsprechenden

n Meldung stattfindet. Die Vorverarbeitung beinhaltet u.a.:

- die Validierung der Meldungsinhalte gegen ein definiertes Regelwerk,
- die Anreicherung der Meldung um zusätzliche Attribute, wie z.B. den Zeitpunkt des Meldungseingangs oder die Id des sendenden Dienstleisters,
- die automatische Berechnung des jeweils zuständigen Gesundheitsamtes,
- die Steuerung der Generierung und Persistierung von Pseudonymen,
- die Verschlüsselung der Meldung für das initial zuständige Gesundheitsamt und
- die Steuerung der Generierung und Bereitstellung von Meldungsquittungen

Abläufe (Überblick)

Tokenausstellung

[01] Der *DEMIS Client* initiiert den Aufbau einer beidseitig authentisierten TLS-Verbindung. Die Übertragung der Informationen zwischen dem DEMIS Client und den Diensten der DEMIS Infrastruktur erfolgt somit grundsätzlich transportverschlüsselt und integritätsgeschützt. Ein Aufruf von Operationen am *DEMIS Token Provider* ist somit nur für Nutzer mit einem validen Zertifikat möglich.

[02] Der *DEMIS Client* sendet einen POST Request an den Token Endpoint des *DEMIS Token Provider*. Der Request beinhaltet für die Ausstellung des Token relevante Informationen.

[03] Der *DEMIS Token Provider* validiert die übergebenen Parameter gegen die interne Konfiguration. Inkorrekte Kombinationen werden durch eine Fehlermeldung quittiert.

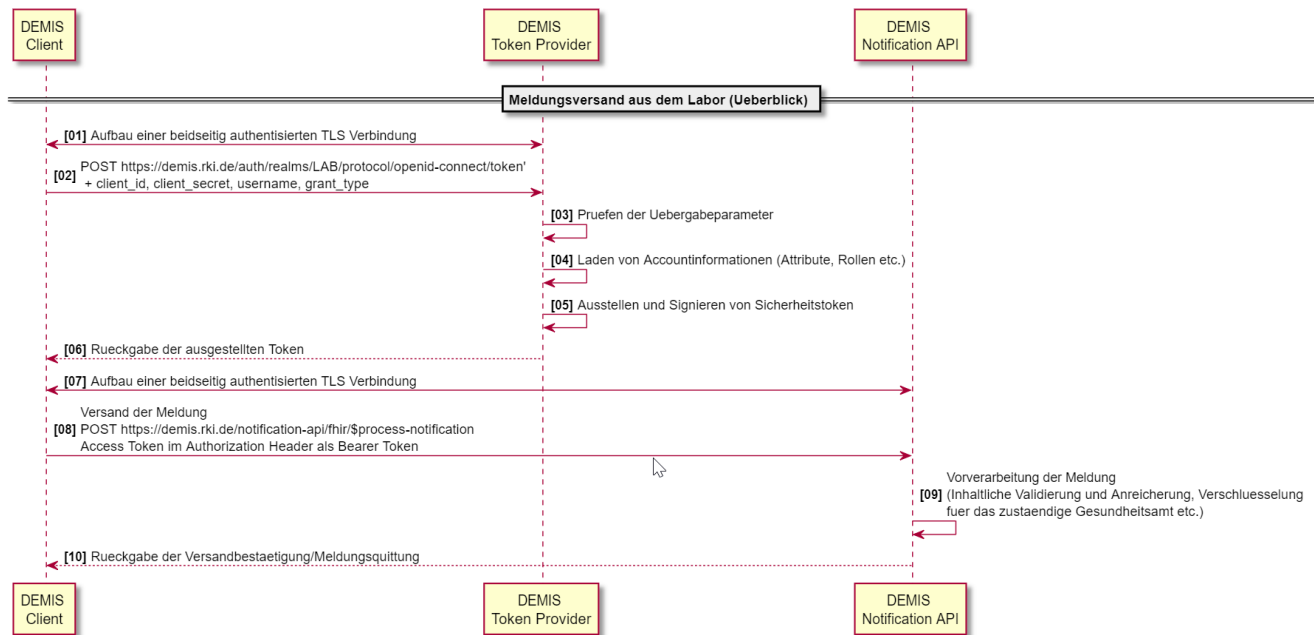
[04] Der *DEMIS Token Provider* prüft nun den Status des identifizierten Accounts. Für den Fall, dass der Account deaktiviert ist oder andere Anmeldevoraussetzungen nicht erfüllt werden, wird die Anfrage mit einer entsprechenden Fehlermeldung quittiert. Sind die Voraussetzungen für eine Anmeldung erfüllt, lädt der Token Provider alle relevanten Nutzerinformationen aus der Datenbank. Dazu gehören u.a. Identitätsattribute, wie z.B. sprechende Organisationsbezeichnungen aber auch die Zuordnung zu bestimmten Anwendungsrollen.

[05] Der *DEMIS Token Provider* bettet nun einen vorkonfigurierten Satz von Identitätsattributen (vgl. [04]) in eine JSON-Datenstruktur ein, versieht diese mit einer Gültigkeitsdauer und signiert sie.

[06] Die ausgestellten Token werden nun eingebettet in eine JSON-Datenstruktur an den aufrufenden *DEMIS Client* zurückgegeben. Das enthaltene Access Token kann für den Zeitraum seiner Gültigkeit für die Autorisierung von Zugriffen auf Dienste der zentralen DEMIS Infrastruktur genutzt werden.

Meldungsversand

[07] Grundlage des Meldungsversands ist der Aufbau einer MTLS-Verbindung zum entsprechenden Endpunkt (vgl. [01]).



DEMIS Client eine Meldung an die *DEMIS Notification API*. Im `http-Authorization-Header` des Requests befindet sich ein vom *DEMIS Token Provider* ausgestelltes Access Token (vgl. [06]). Die Meldung entspricht inhaltlich und strukturell den Vorgaben des RKI (siehe [FHIR Profile](#)).

[09] Nach einer eingehenden Prüfung des eingebetteten Sicherheitstokens erfolgt die Meldungsvorverarbeitung durch die *DEMIS Notification API*. Diese beinhaltet u.a. die Validierung, Anreicherung und Verschlüsselung der Meldung für den zuständigen Empfänger. Fehler während der Verarbeitung werden dem aufrufenden Client mitgeteilt.

[10] Eine im Rahmen der Meldungsverarbeitung erzeugte Versandbestätigung/Meldungsquittung wird an den aufrufenden *DEMIS Client* zurückgegeben.

Tokenausstellung

Der folgende Abschnitt liefert Hintergrundinformationen bezüglich der korrekten Ansteuerung des *DEMIS Token Providers*. Diese Hinweise sollen Hersteller bei der korrekten Implementierung entsprechender Funktionalität sowie der Analyse von Problemen unterstützen.

Mutual TLS

Voraussetzung für die Anmeldung am *DEMIS Token Provider* und die anschließende Ausstellung eines Sicherheitstokens ist der Aufbau einer beidseitig authentisierten TLS-Verbindung mit dem entsprechenden Endpunkt. Informationen aus dem für den Verbindungsaufbau verwendeten Zertifikat sowie das Ergebnis der Zertifikatsprüfung bilden die Grundlage der Nutzeridentifizierung und -authentifizierung. Ein entsprechendes Zertifikat für die *DEMIS*-Produktumgebung kann über den *DEMIS*-Support via demis-support@rki.de durch die meldenden Labore und Krankenhäuser beantragt werden. Zertifikate für den Zugang/Zugriff auf die *DEMIS* Testumgebung müssen durch Hersteller gesondert angefordert werden.

Zulässige Ciphersuites

Folgende Ciphersuites können für den Aufbau der TLS-Verbindung genutzt werden:

```

ECDHE-ECDSA-AES128-GCM-SHA256
ECDHE-RSA-AES128-GCM-SHA256
ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-RSA-AES256-GCM-SHA384
ECDHE-ECDSA-CHACHA20-POLY1305
ECDHE-RSA-CHACHA20-POLY1305
DHE-RSA-AES128-GCM-SHA256
DHE-RSA-AES256-GCM-SHA384
  
```

Implementierungshinweise

Für fast alle Programmiersprachen existieren Bibliotheken, die den Aufbau einer beidseitig authentisierten TLS-Verbindung unterstützen. Im Zusammenhang mit deren Nutzung sind jedoch grundlegende Sicherheitsaspekte zu betrachten. Von besonderer Bedeutung sind in diesem Zusammenhang die folgenden Aspekte:

- **Sichere Schlüsselspeicherung** - Das mit dem Clientzertifikat assoziierte private Schlüsselmateriale muss bestmöglich vor unberechtigten Zugriffen geschützt werden. In diesem Zusammenhang sollte mit entsprechenden Mitteln des Betriebssystems der Zugriff auf die das Schlüsselmateriale enthaltene Datei eng eingegrenzt werden. Zusätzlich sollte das Schlüsselmateriale niemals unverschlüsselt im Dateisystem abgelegt werden. Entsprechende Standards für eine verschlüsselte Speicherung in speziellen Containern existieren und werden von den einschlägigen Bibliotheken umfassend unterstützt. Das für die Ableitung des Containerschlüssels gewählte Passwort sollte entsprechend sicher gewählt werden. Entsprechende Empfehlung hierzu finden sich in einschlägigen Dokumenten des BSI.
- **Truststore Management** - Es sollte darauf verzichtet werden, den Default-Truststore zu verwenden, der üblicherweise mit TLS-Implementierungen ausgeliefert wird. Die zum Einsatz kommenden Serverzertifikate für die DEMIS-Test- und Produktivumgebung stammen aus der "D-TRUST SSL Class 3 CA 1 EV 2009" CA. Ausschließlich diese CA sollte im entsprechenden Truststore als vertrauenswürdig hinterlegt werden.
- **Prüfen des Serverzertifikats** - todo (Hostname-Verification, Certificate Chain Validation ...)

Beispiele

Folgende Beispiele sollen für verschiedene Tools, die häufig im Rahmen der Entwicklung genutzt werden, zeigen, wie die zertifikatsbasierte Authentisierung konfiguriert werden kann:

curl

```
curl -verbose --key key.pem --cert cert.pem --request POST ' ...' ...
```



Aus einem p12 Zertifikat kann man die cert.pem und die key.pem Zertifikate extrahieren, damit sie mit dem angegebenen curl Befehl funktionieren:

- ```
openssl pkcs12 -in path.p12 -out cert.pem -clcerts -nokeys
openssl pkcs12 -in path.p12 -out key.pem -nocerts -nodes
```

Nach Eingabe des p12 Passworts, werden die Dateien extrahiert. Diese kann man mit dem verlinkten curl Befehl verwenden.

**Hinweis:** Im angeführten Beispiel wird das Schlüsselmateriale - anders als in den Implementierungshinweisen empfohlen - unverschlüsselt gespeichert. Dies sollte nur mit dem Schlüsselmateriale der Testumgebung in dieser Form erfolgen.

### postman

Einstellungen unter "File/Settings/Certificates"

## Kommunikation mit dem Token-Endpoint

### Endpunktadressen

## Ergebnis im Erfolgsfall

## Request-Struktur

**Client\_id / client\_secret:**

**Username:**

## Beispiele

## curl

```
curl --verbose --key key.pem --cert cert.pem
--request POST 'https://demis-test.rki.de/auth
/realms/LAB/protocol/openid-connect-token' --
header 'Content-Type: application/x-www-form-
urlencoded' --data-urlencode 'client_id=demis-
adapter' --data-urlencode
'client_secret=secret_client_secret' --data-
urlencode 'username=test-test' --data-
```

SETTINGS

×

General

Themes

Shortcuts

Data

Add-ons

Certificates

Proxy

Update

About

Client Certificates

➤ Add Certificate

Host

https://

demis-test.rki.de

:

443

CRT file

Select File

KEY file

Select File

PFX file

DEMIS-test-test\_CSM021700886.p12

×

Passphrase

.....

🔑

Cancel

Add

Learn more about working with certificates at our Learning Center.

SETTINGS

General

Themes

Shortcuts

Data

Add-ons

Certificates

Proxy

Update

About

CA Certificates

OFF

The file should consist of one or more trusted certificates in PEM format.

PEM file

Select File

Client Certificates

Add Certificate

Add and manage SSL certificates on a per domain basis.

Learn more about working with certificates at our Learning Center.

Host

demis-test.rki.de

Remove

PFX file

C:\ProgramData\Microsoft\Windows Kits\10\Windows Common\10.0.17134.1\DEMIC-test-test\_CSM021700886.p12

Passphrase

\*\*\*\*\*

POST

https://demis-test.rki.de/auth/realms/LAB/protocol/openid-connect/token

Send

Save

Params

Authorization

Headers

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

|                                     | KEY           | VALUE                | DESCRIPTION | ... | Bulk Edit |
|-------------------------------------|---------------|----------------------|-------------|-----|-----------|
| <input checked="" type="checkbox"/> | client_id     | demis-adapter        |             |     |           |
| <input checked="" type="checkbox"/> | client_secret | secret_client_secret |             |     |           |
| <input checked="" type="checkbox"/> | username      | test-test            |             |     |           |
| <input checked="" type="checkbox"/> | grant_type    | password             |             |     |           |
|                                     | Key           | Value                | Description |     |           |

200 OK

```
{
 "access_token": "eyJhbGciOiJI9A",
 "expires_in": 600,
 "refresh_expires_in": 1800,
 "refresh_token": "eyJhbGciOiJI9A",
 "token_type": "bearer",
 "not-before-policy": 0,
 "session_state": "82b56439-f5ce-41d8-a29b-544f2688ee58",
 "scope": "profile"
}
```

## Mögliche Fehlermeldungen

| http response code        | response body                                                                                                      | Mögliche Ursache                                                                                                                                                                                                                                                                                                  |
|---------------------------|--------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 400 Bad Request           | <html><br><head><title>400 The SSL certificate error</title></head><br><body>...</body><br></html>                 | Das für die Authentisierung genutzte Zertifikat ist nicht korrekt oder ungültig. Bitte prüfen Sie, dass das für die jeweilige Umgebung (TEST, PROD) benutzte Zertifikat korrekt konfiguriert ist. Überprüfen Sie die zeitliche Gültigkeit des verwendeten Zertifikats. Überprüfen Sie den Status des Zertifikats. |
| 400 Bad Request           | { "error": "unauthorized_client",<br>,<br>"error_description": "INVALID_CREDENTIALS: Invalid client credentials" } | Die angegebene client_id ist nicht korrekt bzw. es wurde keine client_id angegeben.                                                                                                                                                                                                                               |
| 401 Unauthorized          | {<br>"error": "unauthorized_client",<br>"error_description": "Invalid client secret" }                             | Das angegebene client_secret ist nicht korrekt.                                                                                                                                                                                                                                                                   |
| 401 Unauthorized          | { "error": "invalid_grant",<br>"error_description": "Invalid username credentials" }                               | Der in den Parametern angegebene username ist inkorrekt .                                                                                                                                                                                                                                                         |
| 403 Forbidden             | <html><br><head><title>403 Forbidden</title></head><br><body>...</body><br></html>                                 | Bitte stellen Sie sicher, dass ein gültiges Clientzertifikat für den beidseitig authentisierten Verbindungsaufbau verwendet wird.                                                                                                                                                                                 |
| 500 Internal Server Error | {<br>"error": "unknown_error" }                                                                                    | Der in den Parametern angegebene username korrespondiert ggf nicht mit den Angaben im CN des SubjectDNs im für die Authentisierung genutzten Zertifikat.<br><br>ggf. auch andere Fehler                                                                                                                           |
| 503 Service Unavailable   |                                                                                                                    | Das System wird gewartet und ist nicht erreichbar.                                                                                                                                                                                                                                                                |

**Hinweis:** Weitere Fehlermeldungen sind möglich. Die enthaltenen Beschreibungstexte sind dann jedoch zumeist aussagekräftig genug, um das Problem zu analysieren und zu lösen.

## Struktur des Access Tokens

Für die weiteren Verarbeitungsschritte ist der in die Response-Datenstruktur eingebettete `access_token` von besonderem Interesse. Hierbei handelt es sich um einen JSON Web Token (JWT) gemäß [RFC7519](#), der folgende Inhalte transportiert:

|        |                                                                                                                                                                                                                                                    |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| HEADER | <b>Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render. InvalidValueException'</b><br>{<br>"alg": "RS256",<br>"typ": "JWT",<br>"kid": "tmi4F0p2voWPmdTVESY-Cdy1K9Gkt2Ycny9HC_01MF8"<br>} |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|



|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>BODY</b>      | Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render.InvalidValueException'<br><pre> {   "exp": 1597840018,   "iat": 1597839418,   "jti": "e21f4269-b4bd-437d-b1ae-392d616cdb78",   "iss": "https://demis-test.rki.de/auth/realms/LAB",   "aud": "notification-api",   "sub": "99ca7a5f-89d0-4451-8319-662fcfa3dab2",   "typ": "Bearer",   "azp": "demis-adapter",   "session_state": "82b56439-f5ce-41d8-a29b-544f2688ee58",   "acr": "1",   "resource_access": {     "notification-api": {       "roles": [         "lab-notification-sender"       ]     }   },   "scope": "profile",   "organization": "Test Test",   "preferred_username": "test-lab01" } </pre> |
| <b>SIGNATURE</b> | [Signature Bytes]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

Das Access Token ist als in seiner enkodierten Form als Bestandteil des Requests eingebettet im `Authorization` header (Bearer Token) an die Notification API zu senden.

## Meldungsversand

Der folgende Abschnitt liefert Hintergrundinformationen bezüglich der korrekten Ansteuerung der *DEMIS Notification API*. Diese Hinweise sollen Hersteller bei der korrekten Implementierung entsprechender Funktionalität sowie der Analyse von Problemen unterstützen.

DEMIS Meldungen nutzen als zugrundeliegenden Standard [HL7 FHIR \(R4\)](#). Die Umsetzung ist eine [custom FHIR Operation](#).

## Mutual TLS

Es gelten die [hier](#) getroffenen Festlegungen und Hinweise. **Beachten Sie, dass Sie auch bei der Kommunikation mit `$process-notification` das Zertifikat wie oben beschrieben übergeben müssen.**

## Aufruf der `$process-notification` Operation

### Endpunktadressen

Der Versand von Meldungen erfolgt über die folgenden Endpunktadressen: siehe [Endpunkte](#), [Zertifikate](#), [User](#) und [Passwort](#)

### Request-Struktur

Die zu senden Meldungen müssen diesen Profilen entsprechen: [FHIR Profile](#).

### Beispiele

Beispielrequest finden sich auf der folgenden Seite: [Beispiele](#) und können entsprechend als Referenz für die Gestaltung eigener Strukturen genutzt werden.

Im Zuge der Entwicklung kann es hilfreich sein, generierte Requests über Tools wie z.B. [postman](#) zu senden. Die folgenden Beispiele sollen hier die entsprechend erforderliche Konfiguration der Werkzeuge zeigen:

#### postman

## Ergebnis im Erfolgsfall

POST

https://demis-test.rki.de/notification-api/fhir/\$process-notification

Send

Save

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Code

▼ Headers (0)

| KEY | VALUE | DESCRIPTION | ... | Bulk Edit | Presets ▼ |
|-----|-------|-------------|-----|-----------|-----------|
| Key | Value | Description |     |           |           |

▼ Temporary Headers (10) ⓘ

| KEY             | VALUE                                                                    |
|-----------------|--------------------------------------------------------------------------|
| Authorization   | Bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUiIiwia2lkIA6ICJ0bWk0RjBwMn... |
| Content-Type    | application/json                                                         |
| User-Agent      | PostmanRuntime/7.22.0                                                    |
| Accept          | */*                                                                      |
| Cache-Control   | no-cache                                                                 |
| Postman-Token   | 43744c1f-e74c-4a32-a9cb-065896870aaf                                     |
| Host            | demis-test.rki.de                                                        |
| Accept-Encoding | gzip, deflate, br                                                        |
| Content-Length  | 14361                                                                    |
| Connection      | keep-alive                                                               |

POST

https://demis-test.rki.de/notification-api/fhir/\$process-notification

Send

Save

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

```
1 {
2 "resourceType": "Parameters",
3 "parameter": [
4 {
5 "name": "content",
6 "resource": {
7 "resourceType": "Bundle",
8 "meta": {
9 "lastUpdated": "2020-08-03T23:32:32.527+02:00",
10 "profile": [
11 "https://demis.rki.de/fhir/StructureDefinition/NotificationBundleLaboratory"
12]
13 },
14 "identifier": {
15 "system": "http://demis.rki.de/fhir/todo/bundleIdentifier",
16 "value": "7bb100f2-ccc5-4d2c-842a-be47c8bb2cee"
17 },
18 "type": "document",
19 "timestamp": "2020-08-03T23:32:32.525+02:00",
20 "entry": [
21 {
22 "fullUrl": "https://demis.rki.de/fhir/Composition/37bde9c8-13a2-4249-9f16-2f8f2bb0ee0e",
23 "resource": {
24 "resourceType": "Composition",
25 "id": "37bde9c8-13a2-4249-9f16-2f8f2bb0ee0e",
26 "meta": {
27 "profile": [
28 "https://demis.rki.de/fhir/StructureDefinition/NotificationLaboratorySARSCoV2"
29]
30 },
31 "status": "final",
32 "type": {
33 "coding": [
34 {
35 "system": "http://loinc.org",
36 "code": "34782-3",
37 "display": "Infectious disease Note"
38 }
39]
40 }
41 }
42 }
43]
44 }
45]
46 }
```

200  
OK

Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render.InvalidValueException'

```
{
 "resourceType": "Parameters",
 "meta": {
 "profile": [
 "https://demis.rki.de/fhir/StructureDefinition/ResponseParameter"
]
 },
 "parameter": [
 {
 "name": "bundle",
 "resource": {
 "resourceType": "Bundle",
 "meta": {
 "profile": [
 "https://demis.rki.de/fhir/StructureDefinition/ReceiptBundle"
]
 },
 "type": "collection",
 "entry": [
 {
 "fullUrl": "https://demis.rki.de/fhir/Composition/0e683665-800f-4991-b2e3-6109fb0cbe86",
 "resource": {
 "resourceType": "Composition",
 "id": "0e683665-800f-4991-b2e3-6109fb0cbe86",
 "meta": {
 "profile": [
 "https://demis.rki.de/fhir/StructureDefinition/NotificationReceipt"
]
 },
 "extension": [
 {
 "url": "https://demis.rki.de/fhir/StructureDefinition/ReceivedNotification",

```

```

 "valueIdentifier": {
 "system": "https://demis.rki.de/fhir/NamingSystem/NotificationBundleId",
 "value": "d7c33261-ef86-4dlf-9aec-0fd5a5b16a7c"
 }
 },
 "status": "final",
 "type": {
 "coding": [
 {
 "system": "http://loinc.org",
 "code": "77999-1"
 }
]
 },
 "date": "2020-08-20T13:47:11+02:00",
 "author": [
 {
 "reference": "Organization/DEMIS"
 }
],
 "title": "Meldevorgangsquittung",
 "section": [
 {
 "title": "Zuständiges Gesundheitsamt",
 "code": {
 "text": "Zuständiges Gesundheitsamt"
 },
 "entry": [
 {
 "reference": "Organization/test-oegd01"
 }
]
 }
]
 }
],
 {
 "fullUrl": "https://demis.rki.de/fhir/Organization/test-oegd01",
 "resource": {
 "resourceType": "Organization",
 "id": "test-oegd01",
 "identifier": [
 {
 "system": "https://demis.rki.de/fhir/CodeSystem/reportingSite",
 "value": "test-oegd01"
 }
],
 "name": "Test-Gesundheitsamt 01 / Abteilung Gesundheitsschutz",
 "address": [
 {
 "line": [
 "Teststr. 1"
],
 "city": "Berlin",
 "postalCode": "10115"
 }
],
 "contact": [
 {
 "telecom": [
 {
 "system": "phone",
 "value": "030 - 123 456-789"
 },
 {
 "system": "fax",
 "value": "030 - 123 456-780"
 },
 {
 "system": "email",
 "value": "oegd01@dummy.de"
 }
]
 }
]
 }
 },
 {
 "fullUrl": "https://demis.rki.de/fhir/Organization/DEMIS",
 "resource": {
 "resourceType": "Organization",
 "id": "DEMIS",
 "name": "DEMIS",

```

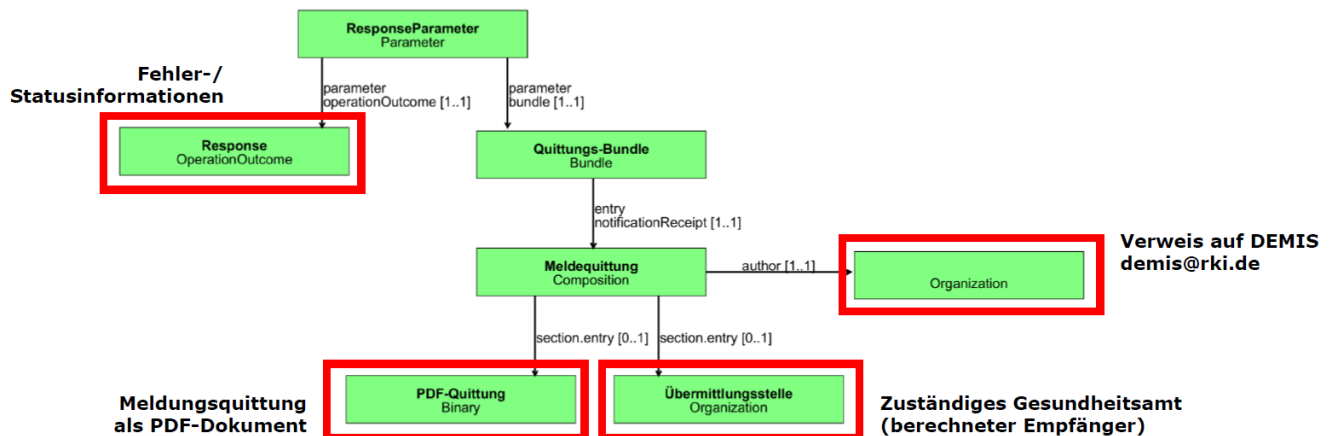
```

 "contact": [
 {
 "telecom": [
 {
 "system": "email",
 "value": "demis@rki.de"
 }
]
 }
]
 },
 {
 "name": "operationOutcome",
 "resource": {
 "resourceType": "OperationOutcome",
 "meta": {
 "profile": [
 "https://demis.rki.de/fhir/StructureDefinition/ProcessNotificationResponse"
]
 },
 "text": {
 "status": "generated",
 "div": "<div xmlns='http://www.w3.org/1999/xhtml'></div>"
 },
 "issue": [
 {
 "severity": "information",
 "code": "informational",
 "details": {
 "text": "All OK"
 }
 }
]
 }
 }
]
 }
}

```

## Response-Inhalte

Die genaue Aufschlüsselung der Response-Inhalte finden Sie im oben genannten PDF bzw. der Informationsveranstaltung für Softwarehersteller. Die zurückgegebenen Informationen können verarbeitet werden. Wichtig ist, dass der Responsecode 200 zurückkommt.



| Fehlercode    | Bemerkung                                                                    |
|---------------|------------------------------------------------------------------------------|
| 403 Forbidden | Das Bearer Token ist nicht valide oder das Zertifikat wurde nicht übergeben. |

## Weitere Hinweise

### Systemverhalten im Wartungsfall

- Sofern sich die DEMIS-Backendinfrastruktur im Wartungsmodus befindet, wird ein „503 Service Unavailable“ zurückgegeben. Der Meldungsversand sollte für eine konfigurierbare Dauer eingestellt und danach wieder aufgenommen werden.

### Mitsenden des User-Agent

- Ein sendender Client soll immer einen User-Agent der Form

```
User-Agent: <product> / <product-version> <comment>
```

mitschicken, siehe <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent>

### Implementierungsbeispiele /Bibliotheken

- Der DEMIS Adapter kann sowohl als Implementierungsbeispiel als auch als Bibliothek für die Meldungserstellung und den –versand genutzt werden.
- Der Quelltext kann derzeit direkt über das RKI ([demis@rki.de](mailto:demis@rki.de)) angefragt werden.

## Weiterführende Informationen

- [Fragen und Antworten](#)