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

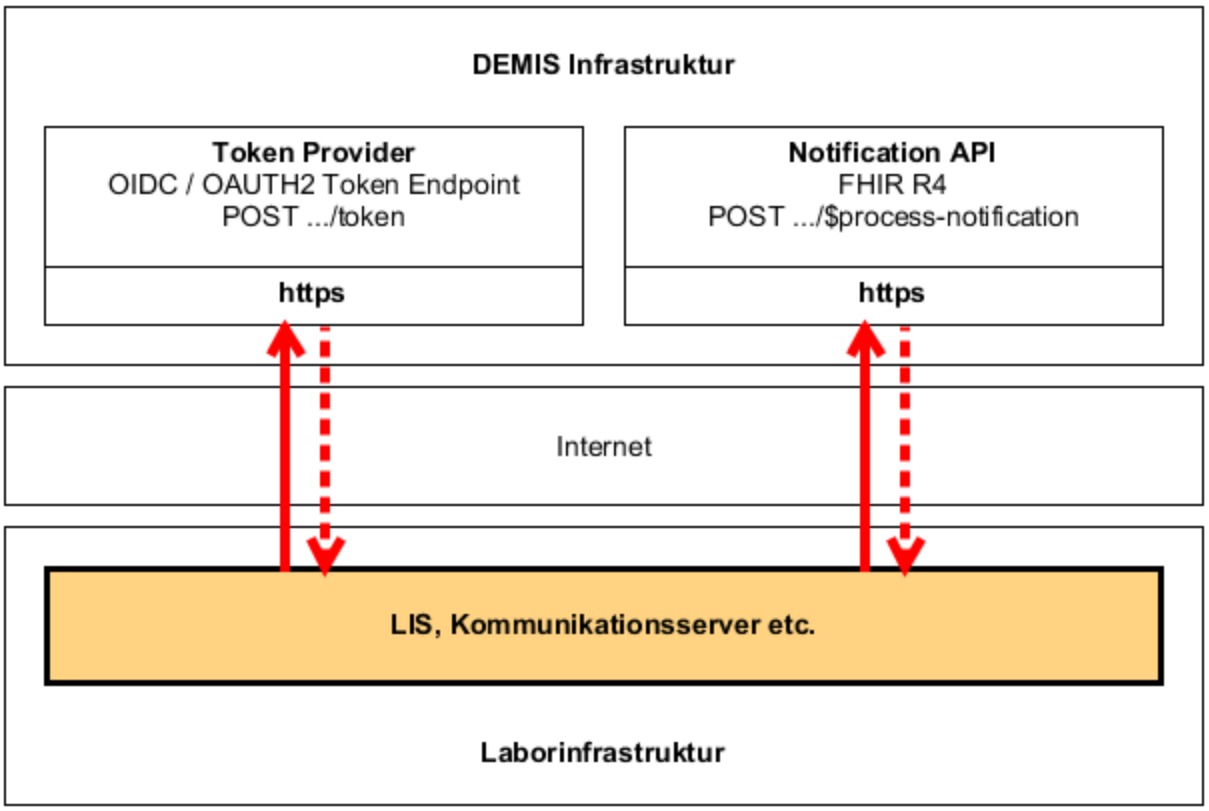
Hier finden Sie die Folien der Infoveranstaltung (Anschluss an DEMIS – Primärsystemintegration.pdf):

Inhalt

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 *D EMIS 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 entsprechende 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.

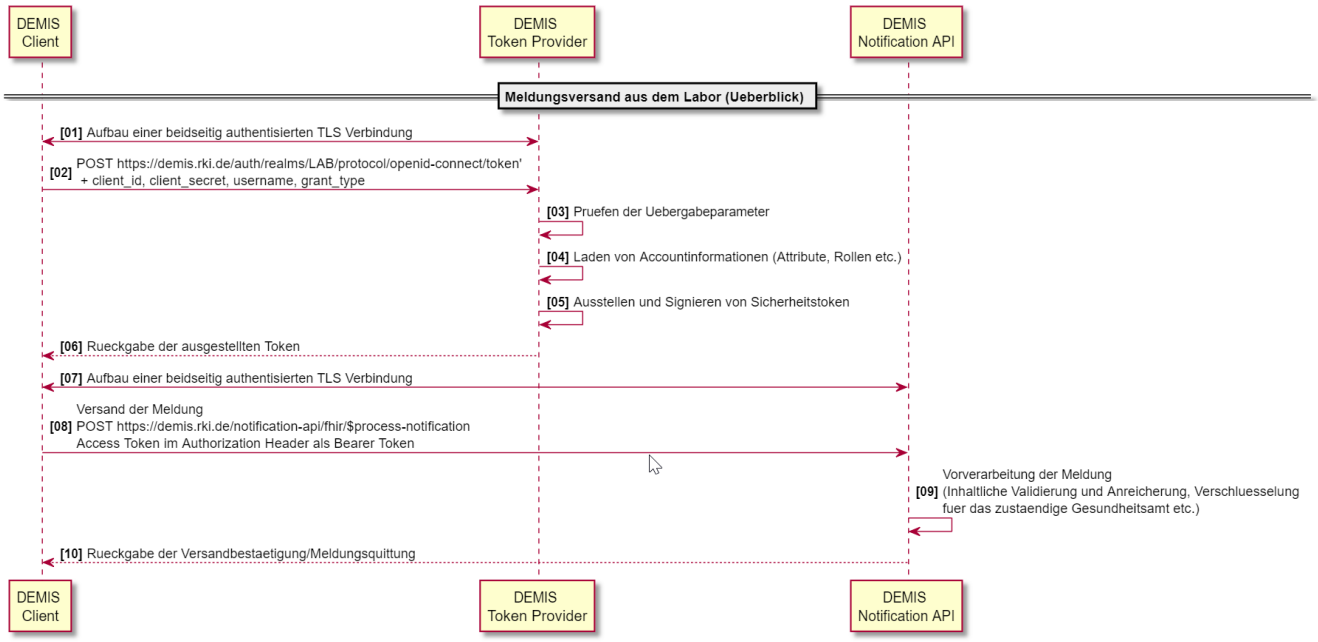
[04] Der *DEMIS Token Provider* prüft nun den Status des identifizierten Accounts. Für den Fall, dass der Account deaktiviert ist oder andere Anmeldevoraussetzungen nicht erfüllt werden, wird die Anfrage mit einer entsprechenden Fehlermeldung quittiert. Sind die Voraussetzungen für eine Anmeldung erfüllt, 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]).



[0
8
Ü

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-Produktivumgebung 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 Schlüsselmaterial muss bestmöglich vor unberechtigten Zugriffen geschützt werden. In diesem Zusammenhang sollte mit entsprechenden Mitteln des Betriebssystems der Zugriff auf die das Schlüsselmaterial enthaltene Datei eng eingegrenzt werden. Zusätzlich sollte das Schlüsselmaterial niemals unverschlüsselt im Dateisystem abgelegt werden. Entsprechende Standards für eine verschlüsselte Speicherung in speziellen Container 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 angegeben 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 Schlüsselmaterial - anders als in den Implementierungshinweisen empfohlen - unverschlüsselt gespeichert. Dies sollte nur mit dem Schlüsselmaterial der Testumgebung in dieser Form erfolgen.

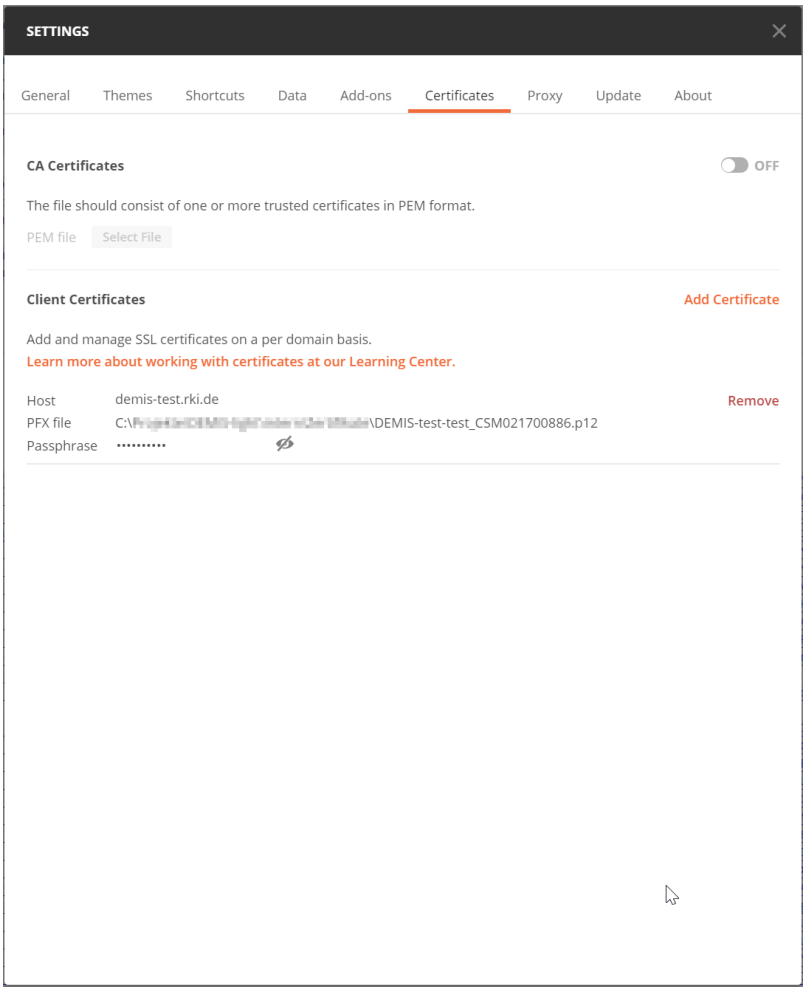
postman

Einstellungen unter “File/Settings/Certificates”

.

Kommunikation mit dem Token-Endpoint

Endpunktadressen
  ```




```
urlencode 'grant_type=password'
```

postman

Ergebnis im Erfolgsfall

Der Abruf von Token kann über die folgenden Endpunktadressen angestoßen werden: siehe Endpunkte, Zertifikate, User und Passwort

Request-Struktur

Es muss ein http-POST-Request mit spezifischen Parametern an die jeweilige Endpunktadresse gesendet werden. Diese Parameter sind `x-www-form-urlencoded` zu übertragen:

```
client_id=[placeholder]
client_secret=[placeholder]
username=[placeholder]
grant_type=password
```

Client_id / client_secret:

Siehe Endpunkte, Zertifikate, User und Passwort

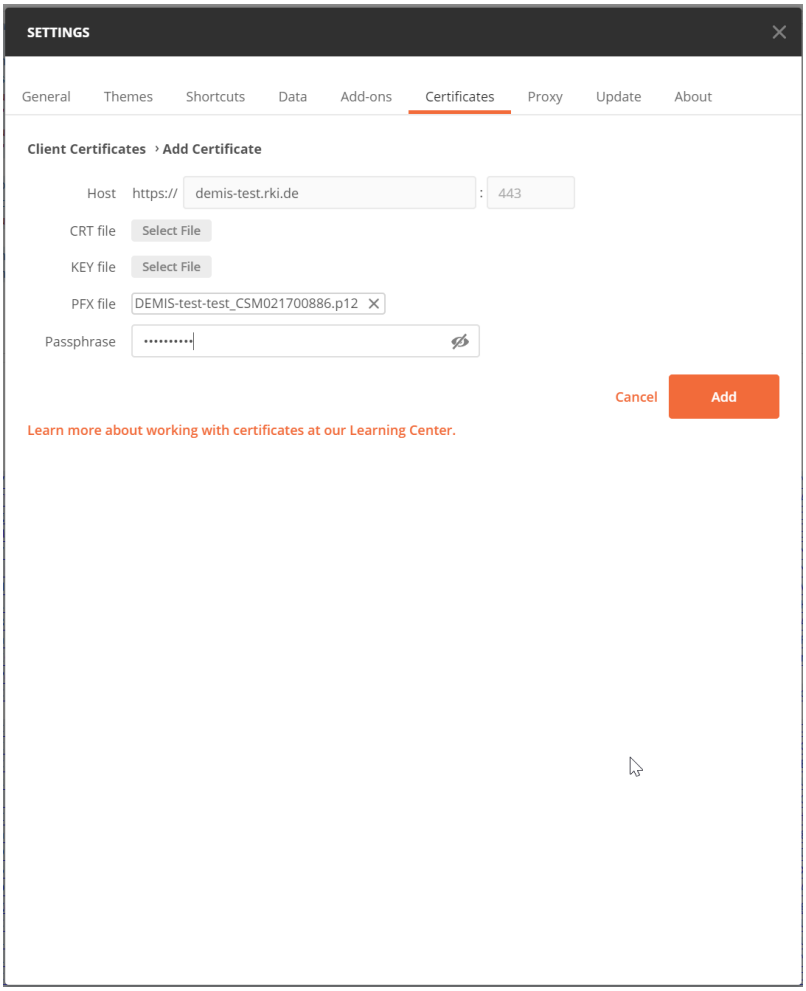
Username:

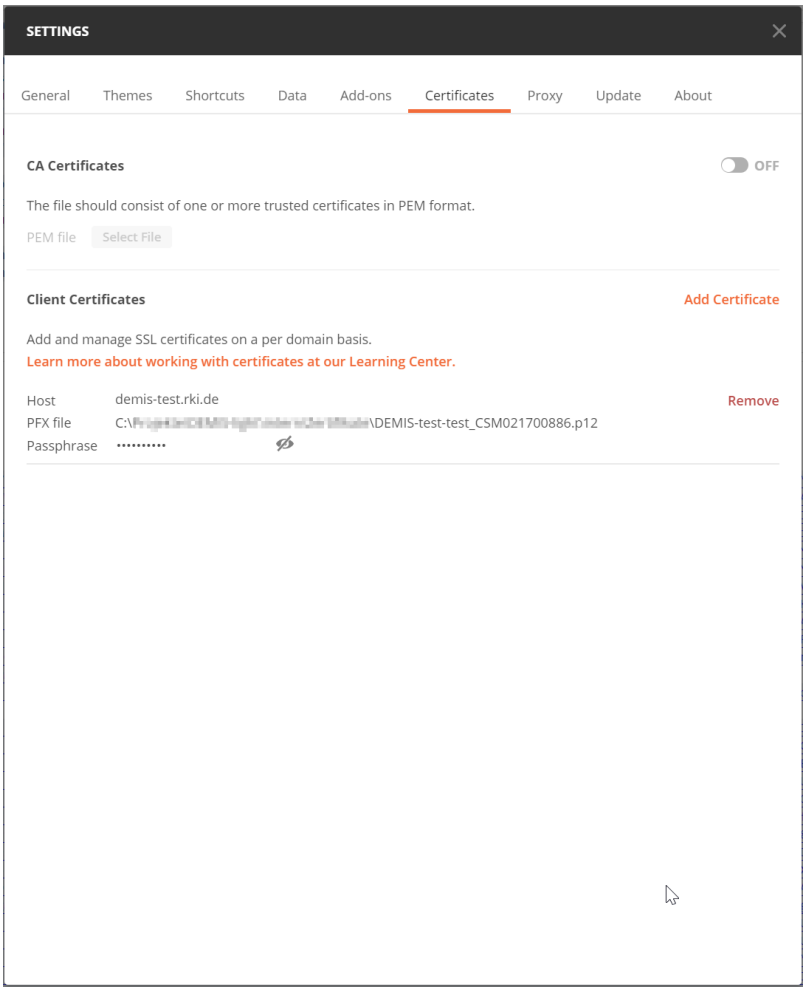
Bitte entnehmen Sie Ihre individuelle DEMIS-Kennung aus der Informationsmail des Robert Koch-Instituts zur DEMIS-Registrierung und Zertifikatsbereitstellung (z.B. DEMIS-12345). Der *username* in der Konfiguration entspricht dem CN des Zertifikats ohne das vorgestellte "DEMIS-", also z.B. bei individueller DEMIS-Kennung bzw. CN=DEMIS-12345 lautet der *username* "12345".

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.



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
client_id	demis-adapter	
client_secret	secret_client_secret	
username	test-test	
grant_type	password	
Key	Value	Description

200 OK


```
{
    "access_token": "eyJhbGciOiJ...H9A",
    "expires_in": 600,
    "refresh_expires_in": 1800,
    "refresh_token": "eyJhbG...2Os",
    "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> <head><title>400 The SSL certificate error</title></head> <body>…</body> </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", "error_description": "INVALID_CREDENTIALS: Invalid client credentials" }	Die angegebene client_id ist nicht korrekt bzw. es wurde keine client_id angegeben.
401 Unauthorized	{ "error": "unauthorized_client", "error_description": "Invalid client secret" }	Das angegebene client_secret ist nicht korrekt.
401 Unauthorized	{ "error": "invalid_grant", "error_description": "Invalid user credentials" }	Der in den Parametern angegebene username ist inkorrekt .
403 Forbidden	<html> <head><title>403 Forbidden</title></head> <body>…</body> </html>	Bitte stellen Sie sicher, dass ein gültiges Clientzertifikat für den beidseitig authentisierten Verbindungsaufbau verwendet wird.
500 Internal Server Error	{ "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. 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	**Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render.InvalidValueException'** **{** **"alg": "RS256",** **"typ": "JWT",** **"kid": "tmi4F0p2voWPmdTVESY-CdylK9Gkt2Ycny9HC_OlMF8"** **}**

BODY	Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render.InvalidValueException'
	`{`
	`  "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"`
	`}`

SIGNATURE	`[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 eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJ0bWk0RjBwMn...
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


```
{
  "resourceType": "Parameters",
  "parameter": [
    {
      "name": "content",
      "resource": {
        "resourceType": "Bundle",
        "meta": {
          "lastUpdated": "2020-08-03T23:32:32.527+02:00",
          "profile": [
            "https://demis.rki.de/fhir/StructureDefinition/NotificationBundleLaboratory"
          ]
        },
        "identifier": {
          "system": "http://demis.rki.de/fhir/todo/bundleIdentifier",
          "value": "7bb100f2-ccc5-4d2c-842a-be47c8bb2cee"
        },
        "type": "document",
        "timestamp": "2020-08-03T23:32:32.525+02:00",
        "entry": [
          {
            "fullUrl": "https://demis.rki.de/fhir/Composition/37bde9c8-13a2-4249-9f16-2f8f2bb0ee0e",
            "resource": {
              "resourceType": "Composition",
              "id": "37bde9c8-13a2-4249-9f16-2f8f2bb0ee0e",
              "meta": {
                "profile": [
                  "https://demis.rki.de/fhir/StructureDefinition/NotificationLaboratorySARSCoV2"
                ]
              },
              "status": "final",
              "type": {
                "coding": [
                  {
                    "system": "http://loinc.org",
                    "code": "34782-3",
                    "display": "Infectious disease Note"
                  }
                ]
              },
```


200 OK

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

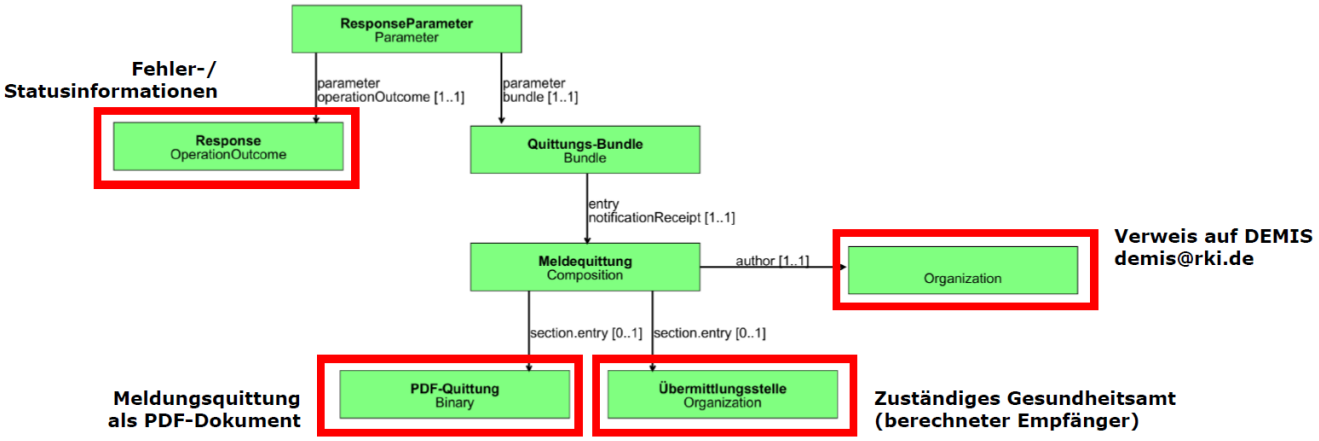

```
{
  "resourceType": "Parameters",
  "meta": {
    "profile": [
      "https://demis.rki.de/fhir/StructureDefinition/ResponseParameter"
    ]
  },
  "parameter": [
    {
      "name": "bundle",
      "resource": {
        "resourceType": "Bundle",
        "meta": {
          "profile": [
            "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-ef86-4d1f-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.



N

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) angefragt werden.

Weiterführende Informationen

- Fragen und Antworten