Beim vorliegenden Dokument handelt es sich um einen Entwurf der gematik in Vorbereitung auf zukünftige normative Festlegungen als Grundlage entsprechender Zulassungs- und Bestätigungsverfahren. Die gematik veröffentlicht diesen Entwurf mit dem Ziel, dass sich Interessierte bereits frühzeitig einen Überblick über die mögliche Weiterentwicklung der Telematikinfrastruktur verschaffen können. Die gematik übernimmt keine Gewähr für die Aktualität, Richtigkeit und Vollständigkeit dieses Entwurfes und behält sich das Recht vor, ohne vorherige Ankündigung Änderungen oder Ergänzungen vorzunehmen oder von den Regelungen insgesamt bzw. teilweise Abstand zu nehmen.

Elektronische Gesundheitskarte und Telematikinfrastruktur

Spezifikation
Identity Provider-Dienst

Version:	1.1.0~~.0~~ CC
Revision:	~~241924~~269879
Stand:	~~30.06~~17.08.2020
Status:	zur Abstimmung freigegeben
Klassifizierung:	öffentlich_Entwurf
Referenzierung:	gemSpec_IDP_Dienst

Dokumentinformationen

Änderungen zur Vorversion

~~Es handelt sich um die Erstversion~~Anpassungen des vorliegenden Dokumentes im Vergleich zur Vorversion können Sie der nachfolgenden Tabelle entnehmen.

Dokumentenhistorie

Version	Stand	Kap./ Seite	Grund der Änderung, besondere Hinweise	Bearbeitung
1.0.0	30.06.20		initiale Erstellung des Dokuments	gematik
1.1.0 CC	17.08.20		Einarbeitung Scope-Themen zu R4.0.1 zur Abstimmung freigegeben	gematik

Inhaltsverzeichnis

1 Einordnung des Dokumentes

1.1 Zielsetzung

Die vorliegende Spezifikation definiert die Anforderungen zu Herstellung, Test und Betrieb des Produkttyps Identity Provider (IdP)-Dienst. Der IdP-Dienst basiert auf den Standards OpenID Connect (OIDC), Open Authorization 2.0 (OAuth 2) und JSON Web Token (JWT). Die hier beschriebenen Schnittstellen werden vom Authenticator-Modul und vom Anwendungsfrontend für eine Authentifizierung eines Nutzers anhand einer Smartcard genutzt. Diese Authentifikation ist die Voraussetzung, damit ein Anwendungsfrontend Zugang zu Fachdaten eines Fachdienstes erlangen kann. Der IdP-Dienst verwaltet und steuert den Authentifizierungsprozess für das E-Rezept und perspektivisch auch für weitere Anwendungen.

1.2 Zielgruppe

Das Dokument richtet sich an Hersteller und Anbieter von Identity Providern, welche die Funktionen des IdP-Dienstes der gematik realisieren wollen.

1.3 Geltungsbereich

~~Die Lösung zielt zunächst auf die Einführung eines IdP als neuen Dienst der TI und ermöglicht die Entwicklung von Anwendungen basierend auf OpenID connect. Der IdP-Dienst basiert auf der Verwendung der bereits herrausgegebenen Identitäten der TI-Plattform und wird von der gematik zunächst für alle Nutzer der Fachanwendung E-Rezept zur Verfügung gestellt. Der IdP-Dienst bietet dieser Fachanwendung damit unabhängig vom Aufbau weiterer OpenID connect basierter Identity Provider die Möglichkeit, alle Teilnehmer der TI anhand ihrer bereits etablierten Identitätsmerkmale zu identifizieren und zu authentisieren. Der IdP-Dienst ist als eine erste möglichst breite Ausbaustufe hin zu einer Lösung mit verteilten Identity Providern anzusehen.~~

~~Für kommende Releases ist daher ein flexibleres Identity Management vorgesehen. Dieses wird es Identitätsherausgebern (z.B. Krankenkassen, LEO) ermöglichen, als Anbieter eigene IdPs mit flexibler Identitätenverwaltung in die TI einzubringen, die den IdP-Dienst für die von ihnen verwalteten Identitäten ersetzen. Die Anbieter können dabei alternative Authentisierungslösungen anbieten, sofern diese sicher genug sind.~~

~~Der Standard OpenID connect und darauf beruhende Produkte im Markt bieten zusätzliche Funktionen an, die perspektivisch interessant sind. Dies betrifft z.B. die Integration SAML2 basierter Anwendungen und den Austausch von Identitäten mit externen (föderierten) IdPs für ein sektorenübergreifendes oder EU-weites Identity Management.~~

~~1.21.1 Zielgruppe~~

~~1.3~~1.1 Geltungsbereich

Dieses Dokument enthält normative Festlegungen zur Telematikinfrastruktur (TI) des deutschen Gesundheitswesens. Der Gültigkeitszeitraum der vorliegenden Version und deren Anwendung in Zulassungs- oder Abnahmeverfahren wird durch die gematik GmbH in gesonderten Dokumenten (z.B. Dokumentenlandkarte, Produkttypsteckbrief, Leistungsbeschreibung) festgelegt und bekannt gegeben.

Schutzrechts-/Patentrechtshinweis

Die nachfolgende Spezifikation ist von der gematik allein unter technischen Gesichtspunkten erstellt worden. Im Einzelfall kann nicht ausgeschlossen werden, dass die Implementierung der Spezifikation in technische Schutzrechte Dritter eingreift. Es ist allein Sache des Anbieters oder Herstellers, durch geeignete Maßnahmen dafür Sorge zu tragen, dass von ihm aufgrund der Spezifikation angebotene Produkte und/oder Leistungen nicht gegen Schutzrechte Dritter verstoßen und sich ggf. die erforderlichen Erlaubnisse/Lizenzen von den betroffenen Schutzrechtsinhabern einzuholen. Die gematik GmbH übernimmt insofern keinerlei Gewährleistungen.

1.4 Abgrenzungen

~~Spezifiziert werden in dem Dokument die von dem Produkttyp bereitgestellten (angebotenen) Schnittstellen. Benutzte Schnittstellen werden hingegen in der Spezifikation desjenigen Produkttypen beschrieben, der diese Schnittstelle bereitstellt.~~

~~Der in diesem Dokument beschriebene IdP-Dienst bietet die Basis für eine IdP-gestützte Identifikationslösung, um unterschiedlichen Nutzergruppen einen ihrer Rolle entsprechenden Zugriff auf Dienste der Provider-Zone der TI zu ermöglichen. Es werden die Schnittstellen beschrieben, die dieser zu bieten hat und anhand welcher Rahmenbedingungen diese umzusetzen sind. Grundsätzlich sind alle angebotenen Leistungen anhand der im Abschnitt 4.1 - Übergreifende Festlegungen aufgeführten RFC (Request for Comments) und natürlich der daraus hervorgehenden BCP (Best Current Practice) umzusetzen. Als Umsetzungsleitlinie sind [OpenID Connect Core 1.0] mit der Erweiterung [HEART I] & [HEART II] (siehe6.5.2- Weitere Dokumente) heranzuziehen. Die TI-weit übergreifenden Festlegungen insbesondere aus Dokumenten wie beispielsweise [gemSpec_Krypt] bezüglich Algorithmen und Schlüsselstärken sowie [gemSpec_PKI] bezüglich zu verwendender Zertifikatstypen und deren Attributausprägung haben Bestand und sind weiterhin bindend.~~

~~Teile der IdP-gestützten Identifikation der Nutzer sind ebenfalls die Dokumente [gemSpec_IDP_FD], welches die Dienste-Anbindung von Fachdiensten an den IdP-Dienst beschreibt, sowie [gemSpec_IDP_Frontend] für Endgeräte der Nutzer. Nutzer sind Leistungserbringer inklusive ihrer Institutionen ebenso wie Versicherte und nicht zuletzt Fachdienste, die ihre gesicherten Ressourcen auf diesem Wege bereitstellen.~~

~~Der mit diesen Dokumenten in Gänze beschriebene IdP-Dienst stellt eine Basisleistung der TI dar, um die durch unterschiedliche TSP herausgegebenen Identitäten zentralisiert auf deren Gültigkeit und Nutzungsberechtigung prüfen zu können und anhand des Prüfungsresultates entsprechende "ID_TOKEN" auszustellen bzw. die Ausstellung zu untersagen.~~

~~Die vollständige Anforderungslage für den Produkttyp ergibt sich aus weiteren Konzept- und Spezifikationsdokumenten; diese sind in dem Produkttypsteckbrief des Produkttyps IdP-Dienst verzeichnet.~~

Nicht Bestandteil des vorliegenden Dokumentes sind die Verfahrensschritte zur Erstellung des notwendigen Schlüsselmaterials. Es wird angenommen, dass Fachdienste ihre innerhalb der TI zu verwendenden Zertifikate für die ~~TLS~~ Transport Layer Security (TLS)-Sicherung über zentrale Plattformdienste der TI beziehen und diese dort auch geprüft werden können.

~~Ebenso wird angenommen, dass verwendete Hard- und Software geeignet ist, um eigenes asymmetrisches Schlüsselmaterial für die Ver- und Entschlüsselung sowie Signatur gemäß den verwendeten Standards erzeugen und dieses verwalten zu können.~~

Als Umsetzungsleitlinie ist [OpenID Connect Core 1.0] heranzuziehen. Die TI-weit übergreifenden Festlegungen – insbesondere aus Dokumenten wie beispielsweise [gemSpec_Krypt] bezüglich Algorithmen und Schlüsselstärken sowie [gemSpec_PKI] bezüglich zu verwendender Zertifikatstypen und deren Attributausprägungen – haben Bestand, sind weiterhin bindend und werden nicht in diesem Dokument beschrieben.

1.5 Methodik

Anforderungen als Ausdruck normativer Festlegungen werden durch eine eindeutige ID in eckigen Klammern sowie die dem RFC 2119 [RFC2119] entsprechenden, in Großbuchstaben geschriebenen deutschen Schlüsselworte MUSS, DARF NICHT, SOLL, SOLL NICHT, KANN gekennzeichnet.

Sie werden im Dokument wie folgt dargestellt:
<AFO-ID> - <Titel der Afo>
Text / Beschreibung
[<=]

Dabei umfasst die Anforderung sämtliche zwischen Afo-ID und der Textmarke [<=] angeführten Inhalte.

Hinweise auf offene Punkte

Offene Punkten werden im Dokument in dieser Darstellung ausgewiesen.

2 Systemüberblick

Der aktuelle Stand der Spezifikation enthält noch verschiedene offene Punkte, welche im Rahmen ~~des Release 4.0.1~~von zukünftigen Releases und ggf. in weiteren Wartungsreleases (u.a. in Abstimmung mit der Industrie) ausgeräumt und ergänzt bzw. angepasst werden. Es handelt sich zusätzlich zu den folgenden drei übergreifenden Themen um konkretere Aspekte, welche in den entsprechenden Kapiteln ausgewiesen werden.

2.1 Allgemeiner Überblick

In der Telematikinfrastruktur werden zahlreiche Fachdienste angeboten. Anwendungsfrontends können über die Authentifizierung des Nutzers am IdP-Dienst Zugriff zu den von den Fachdiensten angebotenen Daten erhalten. Der IdP-Dienst stellt durch gesicherte JSON Web Token (JWT) attestierte Identitäten aus. Gegen Vorlage eines "ACCESS_TOKEN" erhalten Anwendungsfrontends – entsprechend der im Token attestierten professionOID – Zugriff auf die Inhalte der Fachdienste.

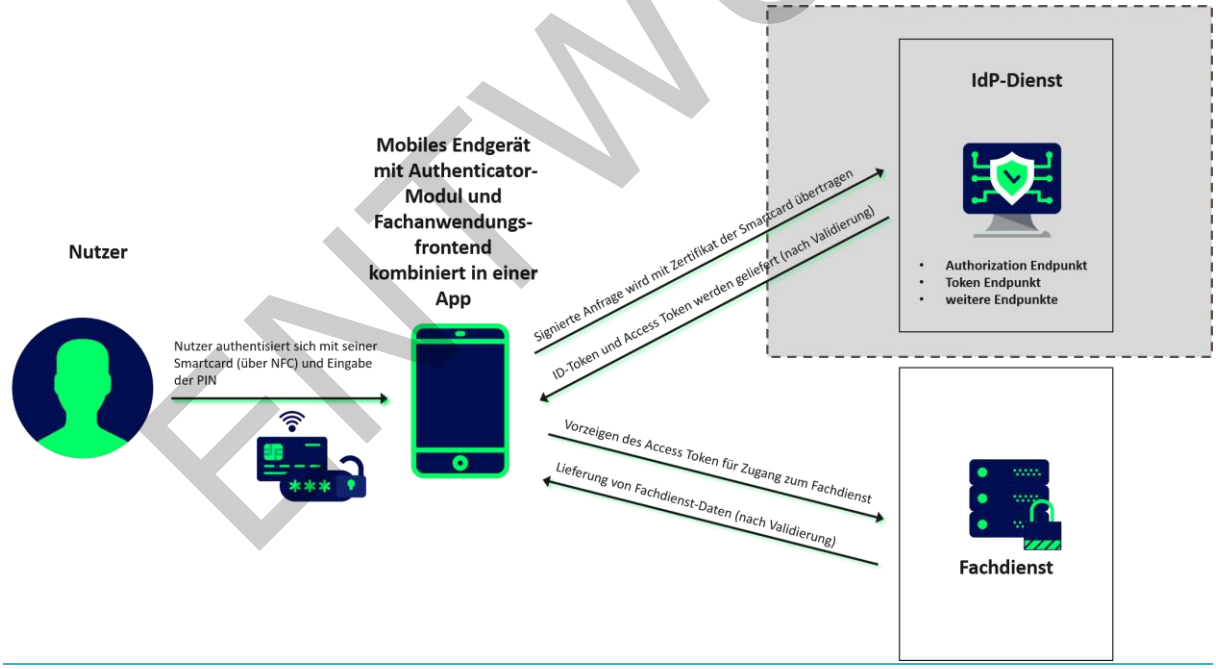


Abbildung 1: Systemüberblick (vereinfacht)

Die ~~*Im aktuellen Stand des Dokumentes fehlen Festlegungen zur Integration von Primärsystemen in der Rolle des Authenticator.*~~

Im aktuellen Stand des Dokumentes fehlen noch in Diskussion befindliche, Festlegungen zur Erweiterung des Integritätsschutzes des Discovery Documents sowie weiterer verwendeter Schlüssel. Die Standards sehen Verschlüsselungen vor, aber lassen die Methoden des Schlüsselaustausch offen und die gematik ist noch in Abstimmung zu praktikablen Methoden.

Im aktuellen Stand des Dokumentes fehlen an einigen Punkten noch Verweise auf die zugrundeliegenden Operationen der Standards OpenID Connect und OAuth2.

Die vorliegende Spezifikation definiert die Anforderungen zu Herstellung, Test und Betrieb des Produkttyps IdP-Dienst, welcher selbst Teil der IdP-gestützten Nutzerauthentifizierung zur Verwendung von openID Connect mit OAuth 2.0 ist.

Dabei wird insoweit vom Standard abgewichen, dass der Resource Owner (Fachdienst) die Identifikation nicht durch einen redirect auf den Authorization Server (IdP-Dienst) überträgt, sondern der Aufruf der Protected Resource erst stattfindet, nachdem die Identifikation erfolgt ist. Dies vereint die im Protokollfluss des Standard in Abbildung 1 RFC6749 # section-1.2] dargestellten "Resource Owner" und "Authorization Server" zu einem Dienst.

obige Abbildung stellt den Systemüberblick dar. Der Authentifizierungsprozess, welcher mit der Ausstellung und Übergabe der Token an das Anwendungsfrontend endet, wird dabei zur besseren Übersicht vereinfacht dargestellt.

Der IdP-Dienst übernimmt für den Fachdienst die Aufgabe der Identifikation des Nutzers. Der IdP-Dienst fasst die professionOID sowie weitere für den Fachdienst notwendige Attribute in signierten JSON Web Token (`"ID_TOKEN"`, `"ACCESS_TOKEN"` und `"SSO_TOKEN"`) zusammen. Fachdienste müssen keine Überprüfung des Nutzers selbst implementieren, sondern können sich darauf verlassen, dass der Besitzer des bei ihnen vorgetragenen `"ACCESS_TOKEN"` bereits identifiziert wurde. Des Weiteren stellt der IdP-Dienst sicher, dass die vom Nutzer vorgetragenen Attribute (aus dem Signaturzertifikat) gültig sind.

Der IdP-Dienst prüft, ob das vorgetragene X.509-nonQES-Signatur-Zertifikat der verwendeten Prozessor-Chipkarte (eGK, HBA oder SMC-B) für die vorgesehene Laufzeit des Tokens zeitlich gültig und ob dessen Integrität sichergestellt ist.

Der IdP-Dienst stellt nur solche `"ACCESS_TOKEN"` aus, welche auf gültigen AUT-Zertifikaten (d.h. C.CH.AUT, C.HP.AUT oder C.HCI.AUT) basieren.

2.2 Detaillierter Überblick

Der IdP-Dienst führt also nicht nur die Identifikation des Nutzers durch, sondern und stattet diesen auch selbst mit einem `"ID_TOKEN"` gemäß [RFC6749 # section-1.4[openid-connect-core 1.0 # IDToken], `einem "ACCESS_TOKEN"` gemäß [RFC6749 # section-1.4] aus.und einem `"SSO_TOKEN"` basierend auf [`RFC7519`] aus. Gewählt wird aus Sicherheitsaspekten der `"Authorization Code Grant"` gemäß [[RFC6749 # section-4.1], wobei daran erinnert sei, dass die Anfrage nicht per Redirect organisiert]. Die Verwendung von PKCE (Proof Key for Code Exchange by OAuth Public Clients) gemäß [RFC7636] wird, sondern der Authenticator die Rolle des User-Agent übernimmt. Die Verwendung eines (mobilen) Webbrowsers ist aufgrund der Nutzung von SmartCards als Authentisierungsmedium nicht umsetzbar gefordert.

Der IdP-Dienst teilt sich in mehrere Teildienste auf. Einzelne Teildienste werden zentral und bei Bedarf auf ~~unterschiedliche~~unterschiedlicher Hardware verteilt betrieben. ~~Der~~Das Authenticator-Modul wird grundsätzlich auf dezentraler Hardware zusammen mit dem Primärsystem oder auf dem mobilen Endgerät des Nutzers betrieben. ~~Die Teildienste, welche vom~~Der IdP-Dienst ~~als zentrale Plattformleistung innerhalb der TI erbracht werden, besitzen dort~~ stellt unterschiedliche Endpunkte bereit, welche eine statische IP-Adressierung und somit statische URI besitzen. Diese statisch adressierten ~~Teildienste~~Endpunkte umfassen:

- Discovery-Endpunkt ("OAuth 2.0 Authorization Server Metadata" ~~[~~[RFC8414])
- Authorization-Endpunkt (Teil des "The OAuth 2.0 Authorization Framework" ~~[RFC6749~~[RFC6749])
- ~~•~~ Token-Endpunkt ~~[~~([RFC6749 # section-3.2~~]~~
  ~~Teildienst 1 ID_TOKEN [RFC6749 # section-1.4 & RFC6749 # section-5]~~
  ~~Teildienst 2 REFRESH_TOKEN [RFC6749 # section-1.5 & RFC6749 # section-6]~~
- ~~• Token Introspection-Endpunkt [RFC7662 # section-2]~~
- ~~• Revocation-Endpunkt [RFC7009 # section-2]~~
- ~~• User Info-Endpunkt [OpenID Connect Core v1.0]~~

~~Der einzig nicht zentral betriebene Teildienst des IdP-Dienstes ist der sogenannte Authenticator, welcher auf jedem einzelnen Endgerät der Nutzer für nur dieses zuständig ist und dort für mehrere Anwendungsfrontends zeitgleich als Authenticator seinen Dienst tun kann.~~

~~Die dynamisch adressierten Teildienste des IdP-Dienstes umfassen:~~

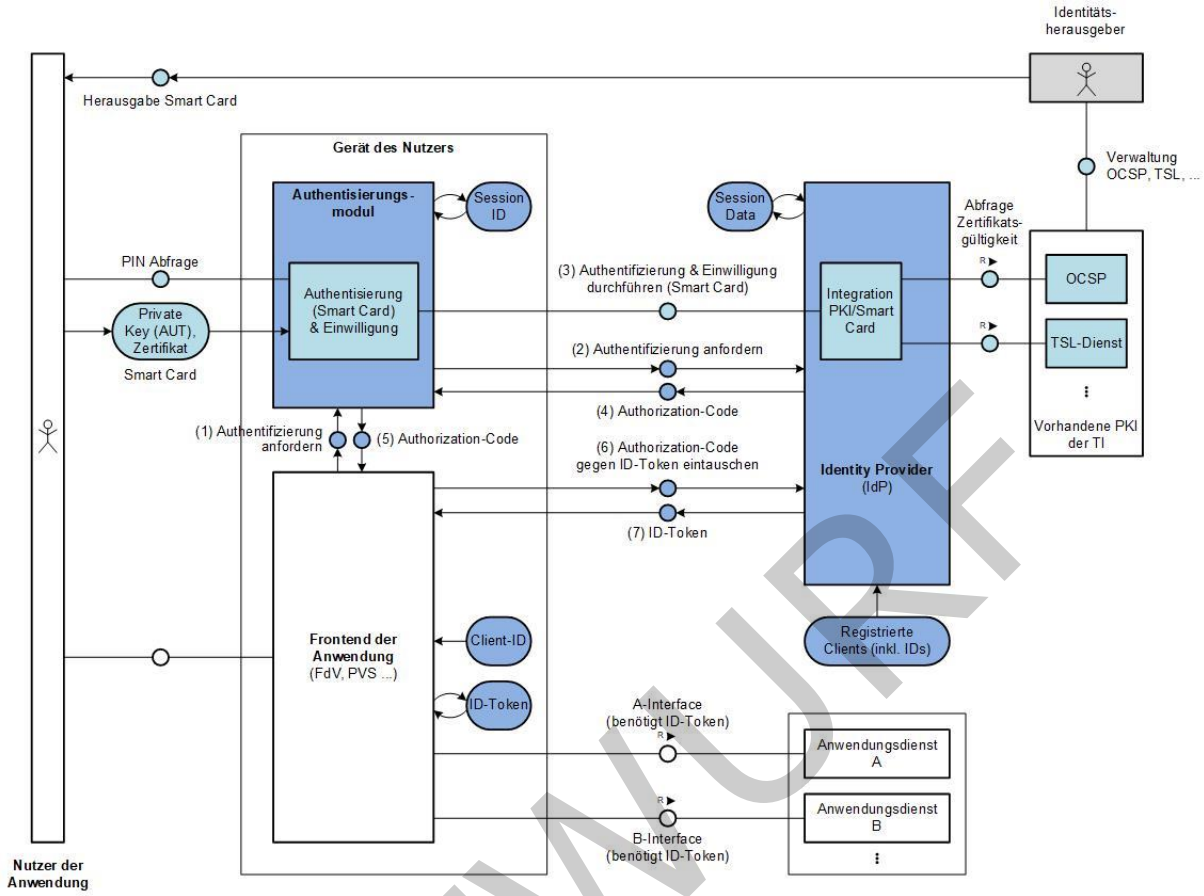
- ~~• Authenticator Applikation (der in [RFC6749 # section-4.1] beschriebene User-Agent)~~
- ~~•~~ ~~Hinweis: Beim Authenticator kann es sich sowohl um eine Applikation als eigenständiges Programm oder eine eingebundene DLL (])~~
  Teildienst 1 "ID_TOKEN" ([openid-connect-core 1.0 # IDToken])
  Teildienst 2 "ACCESS_TOKEN" ([RFC6749 # section-1.4 & RFC6749 # section-5])
  Teildienst 3 "SSO_TOKEN" ([RFC7519]).

~~Dynamic Link Library) handeln, wie auch um eine APP, welche auf einem Smartphone installiert wird. Ebenso ist es möglich, dass Teilfunktionalitäten des Authenticators durch den PTV-Konnektor übernommen werden.~~

Im folgenden Schaubild sind die vom IdP-Dienst bereitgestellten Teildienste blau hinterlegt.

Teildienste wie das Authenticator-Modul und das Anwendungsfrontend befinden sich in dem mit "Gerät des Nutzers" bezeichneten Bereich.

Fachdienste sind nicht näher bestimmt und befinden sich im Block unterhalb des Identity Providers.


Identitäts-herausgeber
Herausgabe Smart Card
Gerät des Nutzers
Verwaltung OCSP, TSL, ...
Authentisierungs-modul
Session ID
Session Data
Abfrage Zertifikats-gültigkeit
PIN Abfrage
Authentisierung (Smart Card) & Einwilligung
(3) Authentifizierung & Einwilligung durchführen (Smart Card)
Integration PKI/Smart Card
OCSP
Private Key (AUT), Zertifikat
Smart Card
(2) Authentifizierung anfordern
TSL-Dienst
(4) Authorization-Code
Vorhandene PKI der TI
(1) Authentifizierung anfordern
(5) Authorization-Code
(6) Authorization-Code gegen ID-Token eintauschen
Identity Provider (IdP)
(7) ID-Token
Frontend der Anwendung (FdV, PVS ...)
Client-ID
Registrierte Clients (inkl. IDs)
ID-Token
A-Interface (benötigt ID-Token)
Anwendungsdienst A
Anwendungsdienst B
B-Interface (benötigt ID-Token)
Nutzer der Anwendung

357

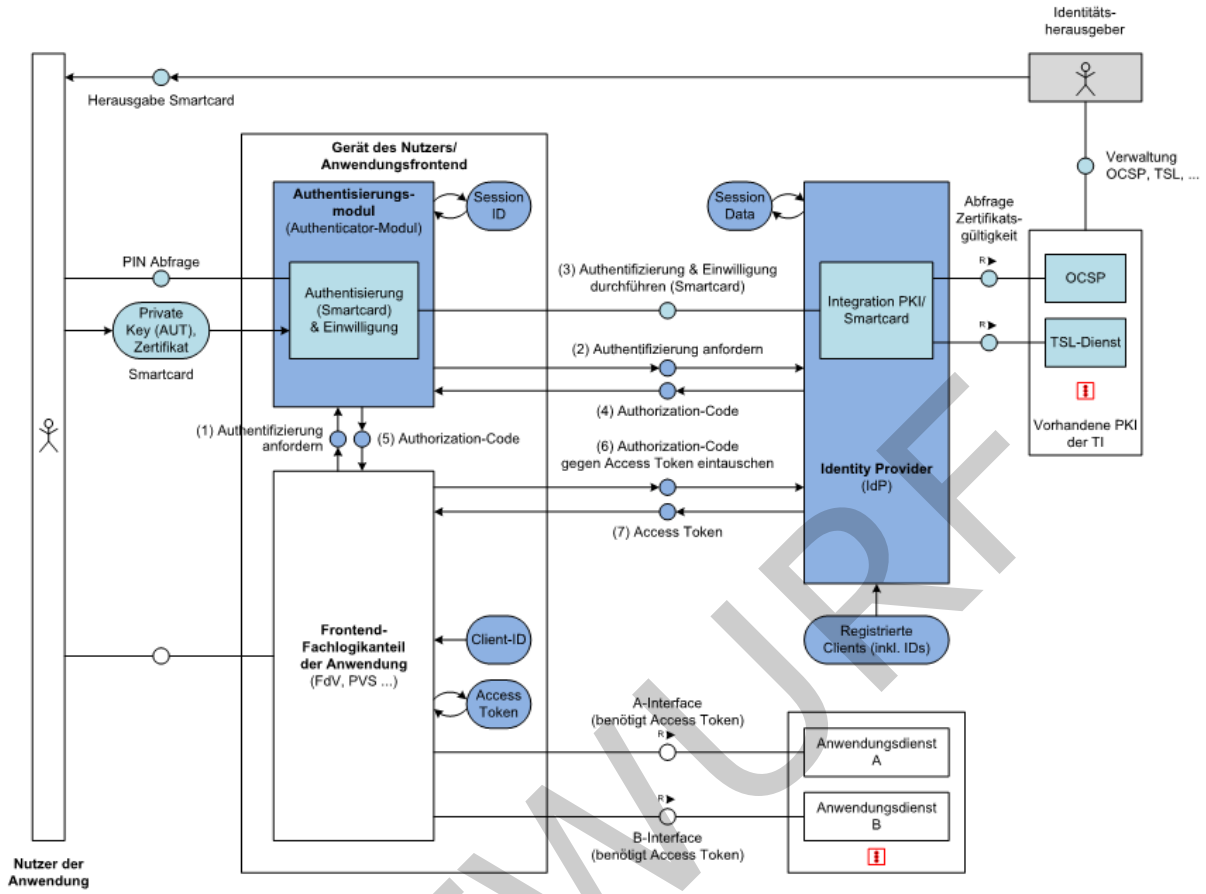


Abbildung 2: Übersichtsschaubild OAuth2.0 Smartcard-IdP-Dienst

Erläuterungen zur obigen Abbildung ~~1~~:
Die Teilschritte (1) und (5) werden bei mobilen Endgeräten (FdV) via Redirection-Endpunkt ~~[~~[RFC6749 # section-3.1.2] realisiert.
Die Teilschritte (1) und (5) können bei Primärsystemen (PVS, AVS, KVS) abweichend ~~[~~ von [RFC6749 # section-9] behandelt werden.
Die Teilschritte (2~~), (3) Challenge-Response)~~ und (4) werden durch den Authorization-Endpunkt ~~[~~gemäß [RFC6749 # section-3.1] bedient. Der Teilschritt (3) Challenge-Response wird durch den Authorizatio-Endpunkt bedient.
Die Teilschritte (6) und (7) werden durch den Token-Endpunkt ~~[RFC6749 # section-3.2~~[RFC6749 # section-3.2] bedient.

~~Die gematik versucht alle technisch möglichen Maßnahmen umzusetzen, um die bekannten [RFC6749 # section-10], aber auch unbekannten sicherheitskritischen Aspekte zu betrachten, um einen maximalen Schutz für die in höchstem Maße schützenswerten Informationen der Versicherten sowie der Leistungserbringer und ihrer Institutionen zu gewährleisten.~~ Der hier gezeigte Smartcard-IdP-Dienst stellt eine Basisleistung innerhalb der TI dar und soll die sichere Identifikation der Akteure anhand der ihnen bereitgestellten Identifikationsmittel (Smartcards) ermöglichen. Der Standard lässt hierbei die Einbringung weiterer Identity Provider für unterschiedlichste Identifikationsverfahren zu, ohne dass Fachdienste hierfür eine ~~grundlegende~~ Änderung der Zugangsmechanismen ~~erfahren~~realisieren müssen.

~~Aus diesem Grund werden von den in den jeweiligen Standards angeführten Optionen immer diejenigen gewählt, die das Maß an Sicherheit verbessern können. Wird~~

~~angeboten, dass etwas vor dem Transport verschlüsselt werden kann, ist diese Maßnahme zwingend umzusetzen. Wird an anderer Stelle eine Umsetzungsempfehlung ausgesprochen, um das Maß der Sicherheit zu erhöhen, so wird diese als zwingende Maßnahme vorgesehen.~~

~~Generell findet die Umsetzung der in den Standards angebotenen optionalen Möglichkeiten gemäß [OpenID Connect] unterstützt durch [OpenID Connect Core v1.0], [OpenID Connect Discovery v1.0], [OpenID Connect Registration v1.0] statt. Um den höchsten Anforderungen an Datenschutz und Datensicherheit gerecht zu werden, kommen weitere Sicherheitsaspekte zum Tragen, welche in "HEART I" [OpenID Heart - OpenID Connect v1.0] und "HEART II" [OpenID Heart - OAuth 2 v1.0] beschrieben sind bzw. gefordert werden.~~

~~Diese selbst basieren wiederum auf zahlreichen anderen Standards, wovon hier nur die erste Ebene genannt sein soll~~Die Umsetzung basiert grundsätzlich auf [OpenID Connect Core v1.0] und [OpenID Connect Discovery v1.0].

Weitere zu beachtende Standards sind die folgenden:
Request for Comments JWT (JSON Web Token) ~~[~~[RFC7519]~~,]~~, JWS (JSON Web Signature) ~~[~~[RFC7515]~~,]~~, JWE (JSON Web Encryption) ~~[~~[RFC7516]~~,]~~, JWK (JSON Web Key) ~~[~~[RFC7517]~~,]~~, JWA (JSON Web Algorithm) ~~[~~[RFC7518] und WebFinger [RFC7033] sowie OAuth 2.0 Bearer~~,~~ [RFC6750], OAuth 2.0 Assertion~~,~~ [RFC7521], OAuth 2.0 JWT Profile~~,~~ [RFC7523], OAuth 2.0 Responses~~.~~ [RFC6749].

Die Gesamtliste der referenzierten Standards finden sich im Abschnitt 6.5.2- Weitere Dokumente.

3 Systemkontext

~~Anwendungsfrontend der Versicherten (Resource Owner) wollen auf Fachdienste (Relying Party [https://openid.net/specs/openid-connect-token-bound-authentication-1_0-03.html]) zugreifen. Der Authenticator ist eine Anwendung (eigenständiges Programm, DLL oder ein entsprechendes Modul im Primärsystem oder eine APP auf einem Smartphone) auf dem Endgerät des Nutzers, welcher sich eindeutig gegenüber dem IdP-Dienst registriert und zukünftig von diesem bei Berechtigungsfreigaben, ähnlich einer Zweifaktor-Authentifizierung, in die Prozesse mit eingebunden wird. Der notwendige Authentisierungsprozess wird vom IdP-Dienst (Authorization Server) angeboten, kann jedoch vom oben genannten Anwendungsfrontend initial nicht direkt, sondern nur über den Authenticator angesprochen werden. Anwendungsfrontend und IdP-Dienst treten daher mit allen anderen Akteuren in Kontakt, Fachdienste und Authenticator jedoch nur mit IdP-Dienst und Anwendungsfrontend. Eine Verbindung zwischen Authenticator und Fachdienst findet niemals statt.~~

~~3.1~~ ~~Verfahrensbeschreibung~~

~~Vorbereitend zur folgenden~~ Die untere Abbildung beschreibt den Systemkontext aus Sicht des IdP-Dienstes. Das Authenticator-Modul liefert die Daten zur Authentifizierung des Nutzers an den IdP-Dienst. Bei positiver Validierung – gegen den OCSP/TSL-Dienst der Public Key Infrastructure (PKI) der gematik – liefert der IdP-Dienst einen "AUTHORIZATION_CODE" zurück. Der IdP-Dienst liefert ebenso einen "SSO_TOKEN", wodurch das Authenticator-Modul einen weiteren "AUTHORIZATION_CODE" ohne erneute Nutzerauthentifizierung erhalten kann.

Das Anwendungsfrontend registriert sich innerhalb eines organisatorischen Prozesses am IdP-Dienst. Das Anwendungsfrontend erlangt gegen Vorlage des "AUTHORIZATION_CODE" einen "ID_TOKEN" und einen "ACCESS_TOKEN". Das Anwendungsfrontend erhält gegen Vorlage des "ACCESS_TOKEN" Zugang zu den Fachdaten des Fachdienstes.

Der Fachdienst registriert sich am IdP-Dienst in Form eines organisatorischen Prozesses.

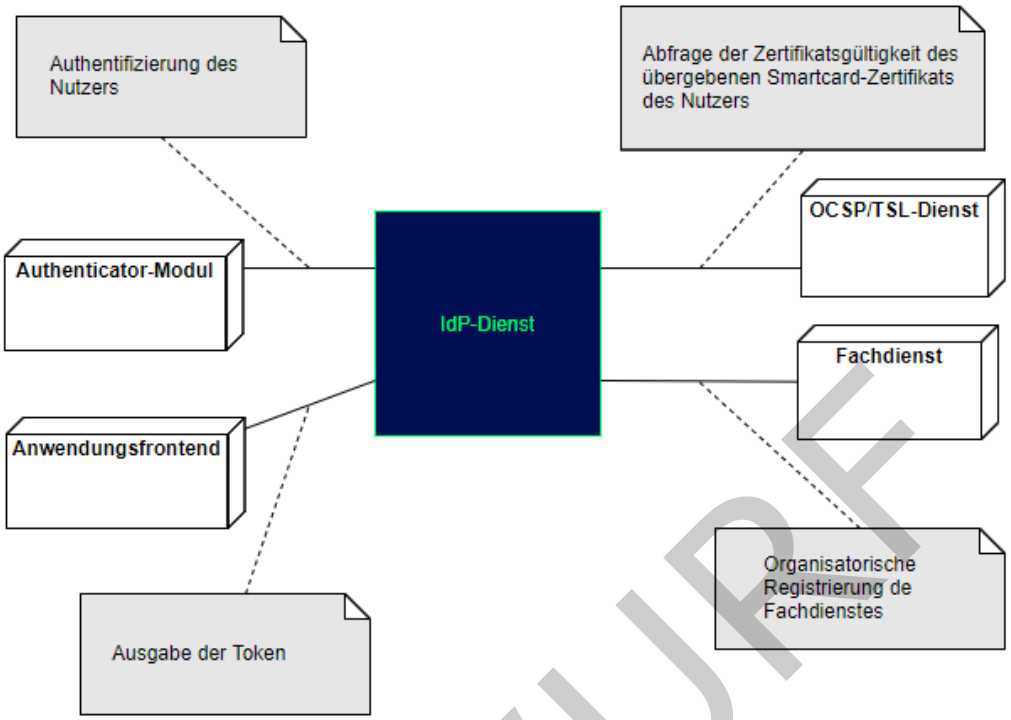


Abbildung 3: Systemkontext aus Sicht des IdP-Dienstes

~~Verfahrensbeschreibung muss der Nutzer sein mobiles Endgerät und den darauf installierten Authenticator beim IdP-Dienst (siehe [gemSpec_IDP_Frontend # Abschnitt 7.2]) registrieren.~~

~~Hinweise: Alle Akteure (Nutzer, Fachdienste ebenso wie der IdP-Dienst selbst) sind beim Authorization Server mit all ihren tokenbasierten Schnittstellen registriert und haben das Discovery Document zur Kenntnis genommen und die für sie relevanten URI und PUK der Gegenstellen im Zugriff. Die Prozesse der Registrierung von Anwendungsfrontend und Authenticator sind im Dokument [gemSpec_IDP_Frontend] beschrieben. Daher kennen alle Akteure auch die URI der Gegenstellen. Alle Vorgänge werden jeweils mit dem publicKey der endgültigen Gegenstelle verschlüsselt. Der "ACCESS_CODE" wird für das Anwendungsfrontend mit dessen "PUK_FRONT" und "ID_TOKEN" für den jeweiligen Fachdienst mit dessen "PUK_FD" verschlüsselt. Alle Instanzen adressieren sich gegenseitig über deren registrierte URI. Ist einer Instanz die URI der Gegenstelle nicht bekannt, liegt entweder ein Fehler in der Registrierung vor und diese ist zu wiederholen oder das Discovery Document des IdP-Dienstes wurde nicht regelkonform ausgewertet. Es müssen die folgenden 14 Schritte durchgeführt werden, um als Anwendungsfrontend in den Besitz eines gültigen "ID_TOKEN" zu gelangen (Voraussetzung: alle Akteure sind registriert):~~

1. ~~Anwendungsfrontend: Herunterladen und Auswerten des Discovery Documents~~
2. ~~Anwendungsfrontend: Erstellen eines "SECRET" und Bilden eines Hashwertes über das "SECRET"~~
3. ~~Anwendungsfrontend: Beauftragen des Authenticators mit Hashwert ein Token zu beschaffen (Redirection)~~
4. ~~Authenticator: Zusammenstellen des Consent~~

5. Authenticator: Einreichen des Consent mit Hashwert beim Authorization-Endpunkt
6. Authorization-Endpunkt: Abgleich des Consent mit Claim des Fachdienstes
7. Authorization-Endpunkt: Einfordern der Challenge am Authenticator/Primärsystem
8. Authenticator: Durchführen der Challenge gegen Authentisierungsmechanismus
9. Authenticator: Rückmelden der Response an Authorization-Endpunkt
10. Authorization-Endpunkt: Herausgeben des "ACCESS_CODE" an Authenticator
11. Authenticator: Weiterleiten des "ACCESS_CODE" an das Anwendungsfrontend
12. Anwendungsfrontend: Einreichen von "ACCESS_CODE" und "SECRET" beim Token-Endpunkt
13. Token-Endpunkt: Bilden eines Hashwertes über das "SECRET" und Abgleich mit Hashwert aus Schritt 5
14. Token-Endpunkt: Herausgabe des "ID_TOKEN" an das Anwendungsfrontend

3.2 Registrierung Authenticator und Anwendungsfrontend

Um ein Anwendungsfrontend nutzen zu können, muss dieses mit einem Authenticator verbunden und beide (Authenticator und Anwendungsfrontend) müssen beim IdP-Dienst registriert sein.

Die Registrierung von Authenticator und Anwendungsfrontend ist im Dokument [gemSpec_IDP_Frontend] beschrieben. Beim IdP-Dienst ist eine eindeutige Verknüpfung des registrierten Authenticators mit dem auf dem Anwendungsfrontend ausgewählten Profil gespeichert.

Startet der Nutzer sein Endgerät, auf welchem der Authenticator installiert ist, und das Gerät ist online, verbindet sich der Authenticator automatisch mit dem Authorization Server und meldet dort die URI, unter welcher er aktuell erreichbar ist, und gegebenenfalls einen neuen öffentlichen Schlüssel "PUK_APP".

Der IdP-Dienst muss die vom Endgerät des Nutzers aus mit dem Authorization Server interagierende Authenticator-Modul bereitstellen und in den jeweiligen Stores für die Betriebssysteme Android (Google Play Store) und Apple (Apple App Store) registrieren und zum kostenlosen Download anbieten.

3.31.1 Anwendungsfrontend vorbereitende Maßnahmen

Das Anwendungsfrontend muss vor dem Aufruf des Authenticators ein asymmetrisches Schlüsselpaar bestehend aus öffentlichem und privaten Schlüssel gemäß [gemSpec_Krypt] erzeugen. Dieses Schlüsselmaterial dient der zielgerichteten Empfänger-Verschlüsselung, bei der man den öffentlichen Teil des Schlüssels "öffentlich" preisgeben kann und das Chiffrat ausschließlich mit dem privaten Teil des Schlüsselpaares wieder entschlüsseln kann.Außerdem muss das Anwendungsfrontend ein SECRET (Zufallswert mit mindestens 128 Bit Güte) und hierüber einen Hash mit einem Algorithmus gemäß [HEART II # rfc.section.2.2.1 & RFC7518 # section-4.6] erzeugen.

3.4 Beschaffung des ID_TOKEN

Der Nutzer ruft sein Anwendungsfrontend auf, wählt sein im Endgerät gespeichertes Profil aus, und gibt seine Zugangsdaten in das Anwendungsfrontend ein. Danach baut das Anwendungsfrontend eine Verbindung zu der im Programm hinterlegten URI des IdP-Dienstes auf, um sich von dort das aktuelle Discovery Document herunterzuladen.

Anhand des Discovery Documents erfährt das Anwendungsfrontend die URI aller Dienste (Authorization-Endpunkt, Token-Endpunkt und Token-Revocation-Endpunkt sowie diejenigen der angebotenen Fachdienste) und wo diese jeweils ihre öffentlichen Schlüssel hinterlegt haben. Über den Authorization Server erfährt es die URI des durch den Authenticator bereitgestellten öffentlichen Schlüssel "PUK_APP". Da die Verknüpfung des Anwendungsfrontend mit seinem Authenticator bereits beim IdP-Dienst registriert ist, kann es seine Anfrage für ein "ID_TOKEN" über den IdP-Dienst via Redirection-Endpunkt an seinen Authenticator schicken.

Inhalt der mit dem "PUK_APP" verschlüsselten Anfrage ist:

- die eigene URI sowie Bezeichnung des aufzurufenden Fachdienstes,
- die eigene Programmbezeichnung mit Versionsnummer,
- der über das eigene SECRET gebildete HASH mit Angabe des Algorithmus,
- der eigene öffentliche Teil des Schlüssels "PUK_FRONT".

Der Authorization-Endpunkt leitet die verschlüsselte Token-Anfrage an die zuletzt eingetragene URI des Authenticators weiter.

Ist der Authenticator offline (nicht erreichbar), wird der Nutzer darauf hingewiesen, dass der Authenticator gestartet werden muss, um dort den Request freizugeben. Wird der Authenticator gestartet und ist online, kann man auf diesem den Request bestätigen.

3.5 Bereitstellung des Authenticators

Der Betreiber des IdP-Dienstes stellt den Authenticator über die für das jeweilige Betriebssystem üblichen Verteilungsmechanismen bereit.

3.6 Aufgaben des Authenticators

Der Authenticator entschlüsselt die "ID_TOKEN"-Anfrage des mit ihm verknüpften Anwendungsfrontend und prüft diese.

3.6.1 Aufgaben bei bestehender Subject Session

Besteht eine aktive Subject Session, versucht der Authenticator am Authorization-Endpunkt, das "ID_TOKEN" für das Anwendungsfrontend zu beantragen. Die Beantragung erfolgt ebenfalls verschlüsselt mit dem "PUK_AUTH", welchen der Authenticator auf Basis des Discovery Document ermittelt hat.

3.6.2 Bei fehlender oder ungültiger Subject Session

Bei einer fehlenden oder abgelaufenen Subject Session erfordert der Authorization-Endpunkt, dass der Authenticator sich erneut authentisiert. Der Nutzer wird aufgefordert, seine Smartcard (eHBA oder eGK) mit dem Authenticator zu verbinden, woraufhin dieser die Registrierung durchführt und eine neue Subject Session etabliert wird.

3.7.1.1 Aufgaben des Authorization-Endpunktes

Der Authorization-Endpunkt nimmt die Anfrage und entschlüsselt diese mit seinem privaten Schlüssel "PRK_AUTH". Nach der Signatur- und Integritätsprüfung überprüft der Authorization-Endpunkt, ob mit den Attributen in der ID_TOKEN-Anfrage die im Claim des Fachdienstes geforderten Parameter bedient werden können.

3.7.11.1.1 Unzureichende Attribute für das Claim

Kann das Claim nicht voll bedient werden, gibt der Authorization-Endpunkt eine Fehlermeldung gemäß [RFC6749 # section-5.2] und fordert den Nutzer zur erneuten Authentisierung und Freigabe der erforderlichen Attribute auf. Auf Grund der systemnahen Ausrichtung der Anwendungsfrontends am zugrundeliegenden E-Rezept und den damit verbundenen Fachdiensten ist eine Abweichung des bestätigten Consent mit den im Claim des Fachdienstes erwarteten Attributen nicht zu erwarten.

3.7.2 Erstellung des ACCESS_CODE

Sind alle im Claim geforderten Attribute vorhanden und deren aktuelle Gültigkeit geprüft, erstellt der Authorization-Endpunkt einen "ACCESS_CODE" und verschlüsselt diesen für das Anwendungsfrontend. Der Authorization Server veranlasst die Erstellung des "ID_TOKEN" und – wenn dies im Claim des Fachdienstes vorgesehen ist – zusätzlich des "REFRESH_TOKEN".

Den "ACCESS_CODE" sendet der Authorization-Endpunkt an die mit der Antragsstellung mitgeteilte URI des Anwendungsfrontend "URI_FRONT" mit dessen öffentlichen Schlüssel verschlüsselt zurück.

3.8 Einreichen des ACCESS_CODE

Das Anwendungsfrontend verschlüsselt den "ACCESS_CODE", zusammen mit dem von ihm in [gemSpec_IDP_Frontend # ML-107977] erzeugten "SECRET", mit dem öffentlichen Schlüssel des Token-Endpunktes "PUK_TOKEN" und reicht diesen dann beim Token-Endpunkt ein.

3.9 Aufgabe des Token-Endpunktes

Der Token-Endpunkt nimmt die verschlüsselten Daten des Anwendungsfrontend entgegen und prüft neben deren Integrität, ob das eingereichte "SECRET" bei Nutzung des identischen Hash-Verfahrens zum bitgleichen Hash-Wert führt. Stimmen die beiden Hash-Werte aus dem initialen Aufruf (siehe 5.4.1.1 Annahme und Prüfung von

~~"ACCESS_CODE" und "SECRET") und dem nun gebildeten Hash-Wert überein, ist sichergestellt, dass Aufrufer und Initiator identisch sind. Der Token-Endpunkt gibt daraufhin das "ID_TOKEN" mit dem öffentlichen Schlüssel des Fachdienstes "PUK_FD" verschlüsselt an das Anwendungsfrontend heraus.~~

~~3.10 Einreichen des "ID_TOKEN" beim Fachdienst~~

~~Um schlussendlich Zugriff auf den Fachdienst zu bekommen, reicht das Anwendungsfrontend das verschlüsselte "ID_TOKEN" beim Fachdienst ein.~~

~~3.11.1 Aufgabe des Fachdienstes~~

~~Der Fachdienst nimmt das verschlüsselte "ID_TOKEN" entgegen und entschlüsselt es mit seinem privaten Schlüssel "PRK_FD". Danach überprüft es die Integrität und die Übereinstimmung mit dem eigenen Claim. Enthält das "ID_TOKEN" mehr oder weniger Attribute, als im Claim vereinbart, oder sind diese fehlerhaft oder nicht befüllt, stimmt die Integrität oder Signatur des "ID_TOKEN" nicht oder ist das "ID_TOKEN" zeitlich nicht mehr gültig, bricht der Fachdienst die Kommunikation mit einer dem Abbruchgrund entsprechenden Fehlermeldung ab.~~

~~Anderenfalls gewährt der Fachdienst dem Anwendungsfrontend Zugriff gemäß eigener Spezifikation.~~

~~3.12~~3.1 Akteure und Rollen

~~3.13 Akteure~~

~~Resource Owner (Nutzer)~~

Im Systemkontext des IdP-Dienstes interagieren verschiedene Akteure (Nutzer und aktive Komponenten) in unterschiedlichen OAuth2-Rollen gemäß [RFC6749 # section-1.1].

Tabelle 1: TAB_IDP_DIENST_0001 Akteure und OAuth2-Rollen

Akteur	OAuth2-Rolle
Nutzer	Resource Owner
Fachdienst	Resource Server
Anwendungsfrontend	Teil des Clients
Authenticator-Modul	Teil des Clients

IdP-Dienst	Authorization Server
Fachdaten	Protected Resource

Nutzer (Rolle: Resource Owner)

Der Resource Owner ist der Nutzer, ~~dem die gesicherten Ressourcen (Protected Resource) auf dem gesicherten Server (Protected Server) zugeordnet sind. Im Falle der TI ist der Resource Owner der Versicherte, Arzt oder die Leistungserbringerinstitution.~~

~~Der Nutzer (Client im herkömmlichen Sinne) ist der Resource Owner, also derjenige,~~ welcher auf die ~~durch die Protected Resource bereitgestellten Informationen Zugriff bekommt.~~

beim Fachdienst (Resource Server ~~[rfc7165#section-5.2] **(Technische Einrichtung zum Bereitstellen der Protected Resource)**~~) für ihn bereitgestellten Daten (Protected Resource) zugreift.

Der Resource Owner verfügt über die folgenden Komponenten:

- Endgerät des Nutzers
- Authenticator-Modul
- Anwendungsfrontend

Fachdienst (Rolle: Resource Server)

Der Resource Server ist ~~im Kontext~~ der ~~Telematikinfrastruktur durch den Fachdienst dargestellt, welcher die geschützten Informationen (Protected Resource)~~Fachdienst, der dem Nutzer (Resource Owner) Zugriff auf ~~der sicheren Umgebung~~seine Fachdaten (Protected ~~Server) bereitstellt.~~

~~**Client (Anbieter eines Fachdienstes)**~~

Resource) gewährt. Der ~~Client ist derjenige, der den IdP-Dienst bzw. die von ihm bereitgestellten Teildienste (Endpunkte) zur Bereitstellung seiner eigenen schützenswerten Daten auf dem Resource Server verwendet.~~

~~**Protected Resource (Serverseitige Fachanwendung)**~~

~~Die Protected Resource sind die durch den Anbieter des Fachdienstes (Client) auf dem Resource Server bereitgestellten Fachanwendungen und Informationen.~~

~~4.1 Zerlegung des Produkttyps~~

~~Der IdP-Dienst ist Teil der Gesamtlösung und stellt die zentralisierte Identitätsprüfung der auf die Fachdienste zugreifenden Nutzer bereit. Als weitere Teile der Gesamtlösung sind neben dem IdP-Dienst die Clients (Fachdienste) mit den Protected Servers (technische Plattform) zu nennen, auf denen die~~ Fachdienst, der die geschützten Fachdaten (Protected Resources ~~(Fachdaten oder Fachverfahren) für den~~) anbietet, ist in der Lage, auf Basis von "ACCESS_TOKEN" Zugriff ~~durch die Nutzer (Versicherte, Bediener eines AVS, PVS oder KVS) bereitgestellt werden. Insbesondere die Gruppe der Nutzer umfasst das Gros aller Akteure, welche aktiv mit dem IdP-Dienst kommunizieren. Ein IdP-Dienst bietet Fachdiensten seine Dienste an, auf welche Millionen Nutzer auch zeitgleich zugreifen. Ein wesentlicher Bestandteil des IdP-Dienstes ist der Authenticator, welcher auf den dezentralen Komponenten in den Praxen, Kliniken, Apotheken und bei den Versicherten betrieben wird. Der Authenticator steht in direktem Kontakt mit dem IdP-Dienst und muss von daher optimal an die dortigen Erfordernisse angepasst sein~~für Clients zu gewähren. Ein solches Token repräsentiert die delegierte Identifikation des Resource Owners.

Anwendungsfrontend/Authenticator-Modul kombiniert in einer Applikation (Rolle: Client)

Der Client greift mit dem Authenticator-Modul und dem Anwendungsfrontend (OIDC Relying Party bzw. OAuth2 Client) auf Fachdienste (Resource Server) und ihre geschützten Fachdaten (Protected Resource) zu. Das Anwendungsfrontend kann auf einem Server als Webanwendung (Primärsystem als Terminalserver), auf einem Desktop-PC oder einem mobilen Gerät (z.B. Smartphone) ausgeführt werden.

IdP-Dienst (Rolle: Authorization Server)

Der Authorization Server authentifiziert den Resource Owner (Nutzer) und stellt "ID_TOKEN", "ACCESS_TOKEN" und "SSO_TOKEN" für den vom Resource Owner erlaubten Anwendungsbereich (SCOPE) aus, welche dieser wiederum beim Fachdienst einreicht.

Tabelle 2: TAB_IDP_DIENST_0002 Kurzbezeichnung der Schnisttstellen des IdP-Dienstes

Kurzzeichen	Schnittstelle
AUTH	Authorization-Endpunkt
TOKEN	Token-Endpunkt
REDIR	Redirection-Endpunkt
DD	Discovery Document-Endpunkt

Weitere Akteure im Kontext IdP-Dienst sind:

Fachdaten (Rolle: Protected Resource)

Die geschützten Fachdaten, welche vom Fachdienst (Resource Server) angeboten werden.

3.2 Begriffsdefinition

~~Der Produkttyp zerlegt sich also in zentrale Komponenten (IdP-Dienst) und dezentrale Komponenten (Primärsysteme und Smartphones), auf welchen der Authenticator als eigenständiges Modul oder zusammen mit dem Anwendungsfrontend (Frontend des Versicherten oder Primärsystem) gemeinsam betrieben wird, um die Authentisierung des Nutzers gegenüber dem IdP durchzuführen.~~

~~*Detailliertere Festlegungen analog zu [gemSpec_FD_eRp#5.6.4 Sicherheit der Netzübergänge] sind noch offen.*~~
~~*Festlegungen für eine Ergänzung des Integritätsschutzes des Discovery Documents mittels TI-PKI sind noch in Diskussion.*~~

~~4.1 Übergreifende Festlegungen~~

~~Rollenausschluss~~

~~Der Anbieter des IdP-Dienstes muss sicherstellen, dass durch Vertreterregelungen oder Krankheitsausfälle keine Verletzung der vorgesehenen Rollentrennung erfolgen kann. Insbesondere ist es den Mitarbeitern des IdP-Dienstes untersagt, gleichfalls Entwickler eines Anwendungsfrontend zu sein oder zukünftig zu werden. Ebenso ist es Mitarbeitern und Führungskräften des Anbieters für den IdP-Dienst untersagt, für andere Anbieter von Diensten für die Telematikinfrastruktur tätig zu sein oder zu werden. Es darf nicht zu einer zeitlich verzögerten Personalunion kommen.~~

~~Zugriff durch Mitarbeiter~~

~~Der Anbieter IdP-Dienst stellt Identitätsnachweise aus, mit deren Hilfe es möglich ist, auf vertrauliche, insbesondere persönliche Informationen zuzugreifen. Der Anbieter des IdP-Dienstes muss darum alles in seiner Macht Stehende unternehmen, um die Zuverlässigkeit, Integrität, Vertrauenswürdigkeit seiner Mitarbeiter sicher zu stellen. Die in den Fachdiensten durch die "ID_TOKEN" erreichbaren Daten sind in höchstem Maße schützenswert, da es sich durchweg um Daten aus dem Vertrauensverhältnis zwischen Arzt und Patient handelt.~~

~~Dokumentationspflicht~~

~~Der Anbieter des IdP-Dienstes hat eine besondere Dokumentationspflicht. So muss jeglicher Zugang bzw. Zugriff auf Systeme, die im direkten Zusammenhang mit dem IdP-Dienst stehen, dokumentiert werden. Hiermit sind insbesondere administrative Zugriffe auf die Systeme gemeint, welche ausschließlich im mindestens "Vier-Augen-Prinzip" zu erfolgen haben.~~

~~Service-Lokalisierung~~

Die Lokalisierung des vom IdP-Dienst angebotenen Service erfolgt ausschließlich über die Bekanntmachung im Fachportal der gematik GmbH.

Verwendete Standards

Soweit nicht explizit etwas anderes beschrieben ist, verhalten sich alle Schnittstellen des IdP-Dienstes und der daran angeschlossenen ortsveränderlichen Komponenten (Endgerät des Nutzers mit Authenticator und Anwendungsfrontend), beschrieben in [gemSpec_IDP_Frontend], sowie Fachdienste als Teil der zentralen TI Providerzone, beschrieben in [gemSpec_IDP_FD], nach den unten aufgelisteten Standards. Verschärft durch die für Applikationen im Gesundheitswesen vorgesehene Erweiterung HEART, in welcher eine erweiterte Schärfung der Vorgaben aus den Standards beschrieben ist.

Durch die konsequent zielgerichtete Verschlüsselung von Consent, Access Code, ID- und Refresh Token der Token Introspection und der Userinfo-Anfragen ist zu jeder Zeit während der Datenübertragung gewährleistet, dass ausschließlich der tatsächlich adressierte Empfänger Zugriff auf die im Payload übertragene Information erhält. Der Schutz der Daten der Fachdienste lässt sich mit dem ID-Token jedoch nicht realisieren und muss von den Fachdiensten bei der Kommunikation zwischen Dienstanbieter und Dienst-Nutzern gesondert gesichert werden. Es bietet sich an, das verwendete Schlüsselmaterial der Anmeldung über den IdP-Dienst hier ebenfalls zu nutzen, um auch das hohe Sicherheitsniveau aufrecht zu erhalten. Dennoch liegt die Aufgabe, diese Kommunikation abzusichern, außerhalb dises Dokuments. Die Qualität des Schlüsselmaterials sowie die verwendeten Algorithmen der eingesetzten Standards sind sorgfältig ausgewählt und entsprechen dem aktuellen Stand der Technik.

Insbesondere sei bei der Vielzahl der verwendeten Standards darauf hingewiesen, dass bei deren Verwendung in jedem Fall eine optional sicherere Variante vorzuziehen ist, wenn sich hierzu die Möglichkeit bietet.

Die verwendeten Standards sind:

- RFC3986 (URI)
- RFC7009 (JSON REVOCATION)
- RFC7165 (JOSE)
- RFC7231 (HTTP)
- RFC7515 (JWS JSON SIGNATURE)
- RFC7516 (JWE JSON ENCRYPTION)
- RFC7517 (JWK JSON KEY)
- RFC7518 (JWE JSON ALGORITHM)
- RFC7519 (JWT JSON WEB TOKEN)
- RFC7520 (JOSE Protection)
- RFC7521 (Assertion Authorization)
- RFC7522 (Assertion SAML 2.0)
- RFC7523 (JSON TOKEN Profile)
- RFC6749 (Oauth2)
- RFC7591 (OAuth2 Dynamic Client Registration)
- RFC6750 (Oauth2 Bearer)
- RFC7636 (Oauth Proof Key for Public Client)

- RFC7662 (OAuth TOKEN INTROSPECTION)
- RFC8252 (OAuth 2.0 for Native Apps)
- RFC8417 (Security Event Token)

A_19881 - Gültigkeitsdauer von JSON-Schlüsselmaterial
Server (Teil-Dienste des Authorization Servers sowie Fachdienste) MÜSSEN zwei Schlüsselsätze gleichzeitig bedienen, wobei die Erneuerung jeweils alle 24 Stunden erfolgen MUSS. Der ältere Schlüssel ist somit <24 Stunden und der jüngere Schlüssel <48 Stunden gültig.**[<=]**

A_19870 - Verwendung eindeutiger URI
Der Anbieter IdP-Dienst MUSS alle verwendeten Adressen in Form von URI gemäß [RFC3986] angeben und in einem Discovery Document gemäß [RFC8414 # section-2] innerhalb der TI und im Internet veröffentlichen. **[<=]**

A_19871 - Bekanntgabe des Downloadpunktes im Fachportal der gematik
Die gematik MUSS den Downloadpunkt des Discovery-Document im Fachportal veröffentlichen.**[<=]**

A_19872 - Discovery Document interne und externe Adressierung
Der Discovery-Endpunkt MUSS die zwei Discovery Documents sowohl innerhalb der TI als auch im Internet jeweils mit voneinander abweichender Adressierung veröffentlichen.**[<=]**

Das Discovery Document innerhalb der TI adressiert hierbei die URI der Fachdienste und Schnittstellen des IdP-Dienstes innerhalb der Telematikinfrastruktur. Das im Internet bereitgestellte Discovery Document stellt die URI der angebotenen Fachdienste im Internet mit dort auflösbaren Adressen oder den statischen IP-Adressen bereit.

Achtung: Es gibt je ein internes und externes (public) "Discovery Document". Diese unterscheiden sich in den darin angebotenen URI, welche gleichlautend im Host-Anteil auf unterschiedliche Domänen bzw.Die folgende Tabelle enthält die Abkürzungen (für die privaten Schlüssel PrK und für öffentliche Schlüssel PUK) der verschiedenen Endpunkte des IdP-Dienstes und deren Verwendung.

Tabelle 3: TAB_IDP_DIENST_0003 Bezeichnungen der Schlüssel und deren URI

	PUK	URI PUK	private Key	URI Dienst
Authorization Server				URI_DD
Authorization-Endpunkt (AUTH)	PUK_AUTH - für die Verschlüsselung des "SSO_TOKEN" - für die Signaturprüfung des "AUTHORIZATION_COD	PUK_URI_AUTH	PrK_AUTH - für die Entschlüsselung des "SSO_TOKEN" - zum Signieren des "AUTHORIZATION_COD	URI_AUTH

	E" und des "SSO_TOKEN"		E" und des "SSO_TOKEN"	
Discovery-Endpunkt (DISC)	PUK_DISC - für die Signaturprüfung des Discovery Document	PUK_URI_DISC	PrK_DISC - zum Signieren des Discovery Document	URI_DISC
Token-Endpunkt (TOKEN)	PUK_TOKEN - für die Signaturprüfung des "ID_TOKEN" und des "ACCESS_TOKEN"	PUK_URI_TOKEN	PrK_TOKEN - zum Signieren des "ID_TOKEN" und des "ACCESS_TOKEN"	URI_TOKEN
Fachdienst (FD)	PUK_FD - für die Verschlüsselung des "ACCESS_TOKEN"	PUK_URI_FD	PrK_FD - für die Entschlüsselung des "ACCESS_TOKEN"	URI_FD

Hinweis: Werden alle Teildienste auf einem Server gemeinsam betrieben, so können diese dasselbe Schlüsselmaterial verwenden. Werden Teildienste auf unterschiedlichen physischen oder logischen Servern betrieben, so sind die Endpunkte mit eigenem Schlüsselmaterial auszustatten.

Die URL des Discovery Documents "URI_DD" stellt somit den zentralen Anlaufpunkt dar, anhand dessen alle weiteren „statischen" Dienste (Endpunkte des IdP-Dienstes und der Fachdienste) adressiert werden können.

3.3 Verfahrensbeschreibung

Vorbereitende Maßnahmen: Das Anwendungsfrontend und der Fachdienst haben sich im Zuge eines organisatorischen Prozesses beim IdP-Dienst registriert. Das Anwendungsfrontend und das Authenticator-Modul haben das Discovery Dokument eingelesen und kennen damit die Uniform Resource Identifier (URI) und die öffentlichen Schlüssel der vom IdP-Dienst angebotenen Endpunkte. Der Fachdienst hat bei der Registrierung am IdP-Dienst seinen öffentlichen Schlüssel hinterlegt.

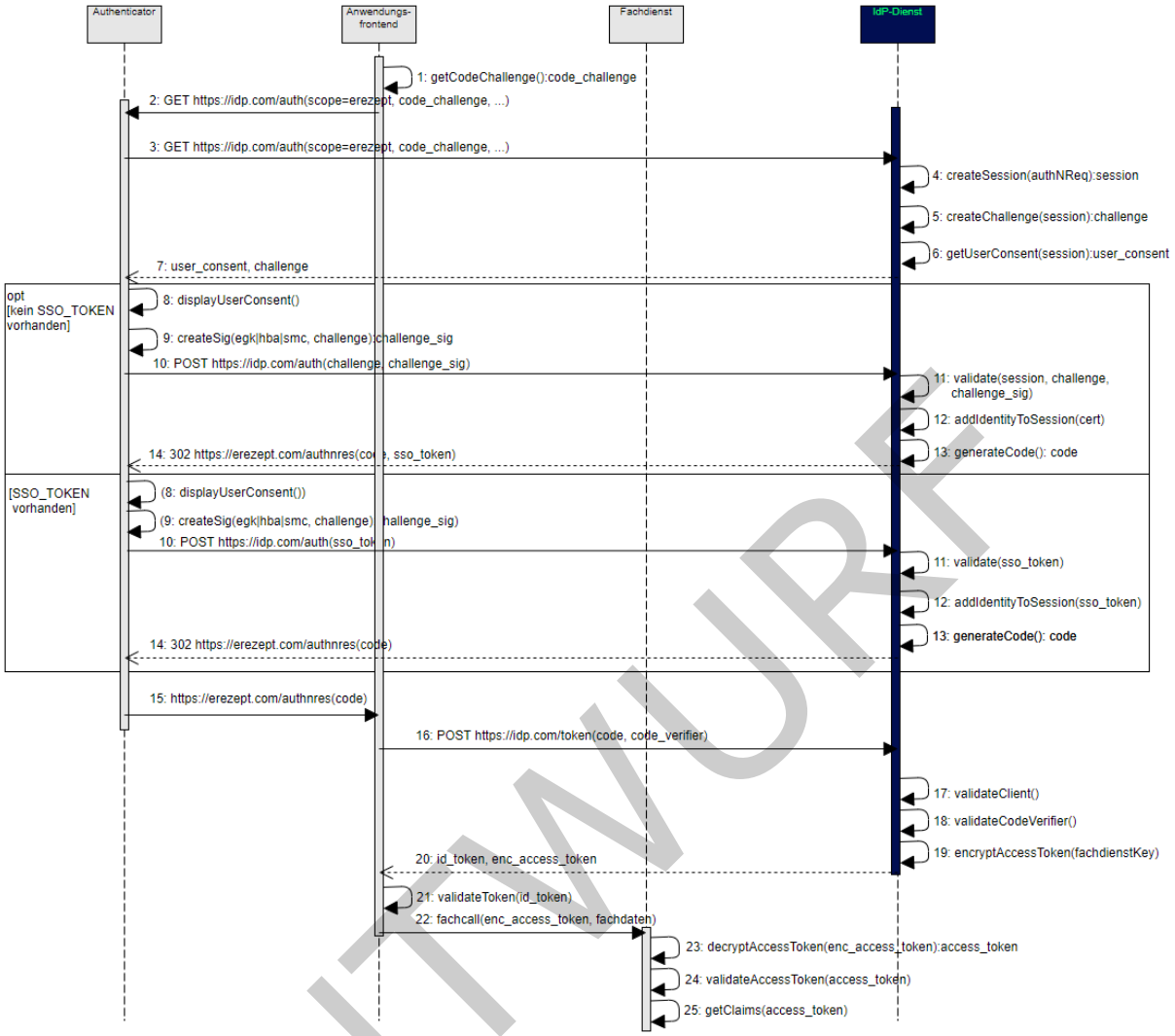


Abbildung 4: Datenfluss-Diagramm IdP-Dienst

Die Prozessschritte, welche notwendig sind, damit ein mobiles Anwendungsfrontend einen Token erhält sind:

1. Das Anwendungsfrontend erzeugt sich einen "CODE_VERFIER" [RFC7636 # section-4.1] und bildet darüber den Hash "CODE_CHALLENGE" mit dem Hash-Algorithmus S256 gemäß [RFC 7636 # section-4.2].
2. Das Anwendungsfrontend überträgt die "CODE_CHALLENGE" gemäß [RFC8252 # Anhang B] an das Authenticator-Modul.
3. Das Authenticator-Modul überträgt die "CODE_CHALLENGE" mit der genutzten "code_challenge_method" S256 weiter an den Authorization-Endpunkt des IdP-Dienst.
4. Der Authorization-Endpunkt legt eine "SESSION_ID" an und speichert alle Informationen zum Vorgang in der "CHALLENGE".
5. Der Authorization-Endpunkt stellt alle Informationen zusammen und erzeugt die "CHALLENGE".

Der Authorization-Endpunkt TLD (Top-Level-Domain) verweisen.

A_19873 - Inhalte des Discovery Documents
Beide Discovery Documente MÜSSEN gemäß [RFC8414 # section-2] mindestens folgende Attribute als URI beinhalten:
iss (Hier ist der IdP-Dienst erreichbar)
jwks_uri (für Abruf der/des PUK des Authorization Server [RFC7517])
"URI_DISC" (URI, unter welcher das Discovery Document bereitgestellt ist)
"URI_AUTH" &"PUK_URI_AUTH" URI des Dienstes und öffentlichen Schlüssels des Authorization-Endpunktes gemäß [RFC6749]
"URI_TOKEN" &"PUK_URI_TOKEN" URI des Dienstes und öffentlichen Schlüssels des Token-Endpunktes gemäß [RFC6749]
"URI_INT" &"PUK_URI_INT" URI des Dienstes und öffentlichen Schlüssels des Introspection-Endpunktes gemäß [RFC7662]
"URI_REV" &"PUK_URI_REV" URI des Dienstes und öffentlichen Schlüssels des Revocation-Endpunktes gemäß [RFC7009]
"URI_INFO" &"PUK_URI_INFO" URI des Dienstes und öffentlichen Schlüssels des Userinfo-Endpunktes.**[<=]**

Hinweis: In jedem Fall muss der IdP-Dienst die URI im Discovery Document für jede einzelne Schnittstelle eintragen, um ein Hardware-Splitting (3-Tier-Lösung) zu ermöglichen. Die vom Authorization Server angebotenen Schnittstellen SOLLEN auf unterschiedlichen physischen Schnittstellen erreichbar sein.
Hinweis: "URI_APP": URI des Authenticator-Moduls wird dynamisch registriert und kann daher nicht im Discovery Document hinterlegt sein. Diese ist eineindeutig mit der "SUBJECT_SESSION" verbunden. "URI_FRONT": Ist die URI des Anwendungsfrontend. Sie wird dynamisch registriert und kann daher nicht im Discovery Document stehen. Die Anwendungs-Session ist eindeutig mit dem Anwendungsfrontend des Nutzers über dessen "URI_FRONT" verbunden und kann durch die "SUBJECT_SESSION" identifiziert werden.

Ein Beispiel eines Discovery Documents finden Sie unter: HEART II # rfc.section.3.1.5

A_19874 - Bereitstellung Internes Discovery Document innerhalb der TI
Der IdP-Dienst MUSS das internal Discovery Document, nach Änderungen und nach Schlüsselwechseln mit seinem privaten Schlüssel "PRK_DD" gemäß [gemSpec_Krypt # Abschnitt 3] signiert, an einem spezifischen Downloadpunkt TLS-gesichert innerhalb der TI bereitstellen.**[<=]**

A_19875 - Absicherung des Internen Discovery Document innerhalb der TI mit TLS
Der Authorization Server MUSS das Discovery Document mit einem Zertifikat der Komponenten-PKI vom Type "C.FD.TLS-S" gemäß [gemSpec_PKI # Abschnitt 5.9.3.2] mit der technischen Rolle "oid_idpd" signieren, um eine TI-interne Prüfung des TLS-Zertifikates zu ermöglichen. **[<=]**

A_19876 - Internes Discovery Document - Prüfung vor Veröffentlichung
Der IdP-Dienst MUSS alle von ihm imDiscovery Document angebotenen URI und URI anderer Dienste insbesondere Fachdienste vor deren Veröffentlichung im internal Discovery Document auf bloße Erreichbarkeit prüfen. **[<=]**

~~A_19877 - Bereitstellung Externes Discovery Document im Internet~~

6. ~~Der IdP-Dienst~~ stellt den mit dem entsprechenden Fachdienstes vereinbarten Consent (Zustimmung des Nutzers zur Verarbeitung der angezeigten Daten) zusammen.
7. Der Authorization-Endpunkt überträgt "CHALLENGE" und Consent-Abfrage "USER_CONSENT" zum Authenticator-Modul.
8. Das Authenticator-Modul fordert den Nutzer zu Consent-Freigabe auf mittels Smartcard und PIN-Eingabe. Falls bereits ein "SSO_TOKEN" beim Authenticator-Modul existiert, entfällt dieser Schritt.
9. Das Authenticator-Modul verwendet die PIN um die "CHALLENGE" von der Smartcard signieren zu lassen. Falls bereits ein "SSO_TOKEN" beim Authenticator-Modul existiert, entfällt dieser Schritt.
10. Das Authenticator-Modul überträgt "CHALLENGE" mit dem Smartcard-Zertifikat an den IdP-Dienst (Antwort Schritt 7). Falls ein "SSO_TOKEN" beim Authenticator-Modul existiert, wird diese Token anstatt der "Challenge" zum IdP-Dienst transportiert.
11. Der Authorization-Endpunkt validiert die "SESSION_ID", "CHALLENGE" und "SIGNATUR". Die Signatur wird anhand des im "x5c"-Header mitgelieferten Authentifizierungszertifikats der Smartcard validiert. Falls ein "SSO_TOKEN" angenommen wurde, wird dieses validiert. Entschlüsselt wird das "SSO_TOKEN" vom Authorization Endpunkt mit seinem privaten Schlüssel "PrK_AUTH". Die Überprüfung der Signatur des "SSO_TOKEN" führt der Authorization Endpunkt anhand seines öffentlichen Schlüssels "PUK_AUTH" durch.
12. Der Authorization-Endpunkt verknüpft die "SESSION_ID" mit der Identität aus der Signatur. Falls ein "SSO_TOKEN" angenommen wurde, verknüpft der Authorization Endpunkt die "SESSION_ID" mit der Identität aus dem "SSO_TOKEN".
13. Der Authorization-Endpunkt erstellt den "AUTHORIZATION_CODE".
14. Der Authorization-Endpunkt überträgt den "AUTHORIZATION_CODE" und den "SSO_TOKEN" an das Authenticator-Modul (Antwort Schritt 3). Falls das ~~public Discovery Document erneut nach Änderungen und nach einem Schlüsselwechsel mit einer spezifischen URI TLS-gesichert im Internet zum Download bereit. Das public Discovery Document MUSS~~das Authenticator-Modul ein vorhandenes "SSO_TOKEN" an den Authorization Endpunkt zur Erlangung eines "AUTHORIZATION_CODE" geschickt hat, wird kein neues "SSO_TOKEN" vom Authorization Endpunkt erstellt und verschickt. Der "AUTHORIZATION_CODE" und das "SSO_TOKEN" werden vom Authorization Endpunkt mit seinem privaten Schlüssel ~~des Discovery-Endpunktes "PRK_DD" gemäß [gemSpec_Krypt # Abschnitt 3.1]~~ "PrK_AUTH" signiert. Der Authorization Endpunkt verschlüsselt das "SSO_TOKEN" und mit seinem öffentlichen Schlüssel "PUK_AUTH" verschlüsselt.
15. Das Authenticator-Modul überträgt den "AUTHORIZATION_CODE" an das Anwendungsfrontend (Antwort Schritt 2).
16. Das Anwendungsfrontend sendet "CODE_VERFIER" und "AUTHORIZATION_CODE" zum Token-Endpunkt des IDP-Dienstes.

17. Der Token-Endpunkt validiert den Client ("SESSION_ID" in der signierten "CHALLENGE"). Der Token Endpunkt validiert den "AUTHORIZATION_CODE" anhand des öffentlichen Schlüssels "PUK_AUTH" des Authorization Endpunktes.
18. Der Token-Endpunkt validiert den "CODE_VERFIER" und gleicht diesen mit der "CODE_CHALLENGE" ab.
1. Der Token-Endpunkt erzeugt die erforderlichen Token, signiert ~~sein.~~ ~~[<=]~~
2. 19. die Token mit seinem privaten Schlüssel "PrK_TOKEN" und verschlüsselt das"ACCESS_TOKEN" mit dem öffentlichen Schlüssel "PUK_FD" des angeforderten Fachdienstes.
20. Der Token-Endpunkt überträgt die Token an das Anwendungsfrontend (Antwort Schritt 16).
21. Das Anwendungsfrontend prüft die Token-Signatur anhand des öffentlichen Schlüssels "PUK_TOKEN" des Token Endpunktes.
22. Das Anwendungsfrontend reicht das gültige "ACCESS_TOKEN" beim Fachdienst ein.
23. Der Fachdienst entschlüsselt das "ACCESS_TOKEN" mit seinem privaten Schlüssel "PrK_FD".
24. Der Fachdienst validiert das "ACCESS_TOKEN" anhand des öffentlichen Schlüssels "PUK_TOKEN" des Token Endpunktes.
25. Der Fachdienst zieht die Claims (d. h. die Key/Value-Paare im Payload eines Tokens) aus dem "ACCESS_TOKEN" und gibt bei positiver Validierung den Zugriff auf die Fachdaten frei.

Hinweis: Verwendet der Nutzer ein Primärsystem, führt der Konnektor in in Schritt 9 die Funktion "externalAuthenticate" für eine Signatur mit der SMC-B durch. Setzt der Nutzer ein mobiles Endgerät ein, ruft das Authenticator-Modul die Signaturfunktion eines HBA oder einer eGK für eine nonQES-Signatur der Smartcard auf. Die erforderlichen Token in Schritt 19 sind "ID_TOKEN", "ACCESS_TOKEN" und "SSO_TOKEN". Das Authenticator-Modul kann mit dem "SSO_TOKEN" einen neuen "AUTHORIZATION_CODE" beim IdP-Dienst ohne erneute Nutzer-Authentifizierung anfordern und damit ein neues "ACCESS_TOKEN" vom IdP-Dienst erhalten. Im"SSO_TOKEN" hinterlegt der IdP-Dienst die für ihn selbst bestimmten Informationen zum gesamten Vorgang, sodass er keine schützenswerten Informationen zentral speichern muss. Das "SSO_TOKEN" beinhaltet alle Daten, die beim IdP-Dienst benötigt werden, um auf die Vorgangshistorie zurückzugreifen und ggf. neue "ACCESS_TOKEN" herauszugeben. Die Informationen im "SSO_TOKEN" sind mit dem öffentlichen Schlüssel des Authorization-Servers für diesen selbst verschlüsselt und können ausschließlich mit dem privaten Schlüssel des Authorization Servers wieder entschlüsselt werden.

Im Schaubild Datenflussdiagramm IdP-Dienst oben ist der Datenfluss zwischen Anwendungsfrontend, Authenticator-Modul, IdP-Dienst und Fachdienst dargestellt. Der Datenfluss weicht im Falle von Primärsystemen hiervon ab, wenngleich Primärsysteme ebenfalls Nutzer-Endgeräte sind. Die Abweichung des Datenflusses wird im nächsten Kapitel erläutert.

3.4 Abweichende Verfahrensbeschreibung für Primärsysteme

Da bei Primärsystemen der Zugriff auf das Authenticator-Modul nicht in allen Fällen in Form eines Links innerhalb des Systems erfolgen kann, muss von der Vorgehensweise für

mobile Endgeräte des Nutzers abgewichen werden. Das Primärsystem hat nicht die Möglichkeit, die Anfrage zum freizugebenden Consent anzuzeigen und nicht zur Eingabe der PIN aufzufordern. Zum Betrieb des Primärsystems ist es notwendig, dass sich die SMC-B im freigeschalteten Modus befindet. Damit muss die Freischaltung der SMC-B genutzt werden, um die Consent-Freigabe dauerhaft zu bestätigen und die vom IdP-Dienst in Schritt 6 geforderte Challenge ohne PIN-Eingabe zu realisieren.

Für die Signatur der Challenge wird die Funktion "externalAuthenticate" des Konnektors verwendet, welcher diesen Funktionsaufruf nur von den Primärsystemen entgegennimmt, an welchen ein Mitarbeiter der Praxis aktiv eingeloggt ist.

3.5 Registrierung Anwendungsfrontend und Fachdienst

Um ein Anwendungsfrontend nutzen zu können, muss dieses gemeinsam mit einem Authenticator-Modul in einer Applikation kombiniert und am IdP-Dienst registriert sein. Die Registrierung des Anwendungsfrontends ist im Dokument [gemSpec_IDP_Frontend] beschrieben.

Anbieter von Fachdiensten müssen Ihre Fachdienste über einen organisatorischen Prozess am IdP-Dienst durchführen.

A_20737 - Ermöglichung einer organisatorischen Registrierung für Anwendungsfrontends und Fachdienste
Der Anbieter des IdP-Dienstes MUSS eine organisatorische Registrierung von Anwendungsfrontends und Fachdiensten ermöglichen.**[<=]**

3.6 Anwendungsfrontend vorbereitende Maßnahmen

Das Anwendungsfrontend muss ein "CODE_VERIFIER" (Zufallswert) gemäß [RFC7636 # section-4.1] und hierüber einen Hash, die "CODE_CHALLENGE", gemäß [RFC7636 # section-4.2] mit dem Algorithmus S256 gemäß [RFC7636 # section-4.2] erzeugen.

3.7 Anfrage eines ACCESS_TOKEN

Die folgende Anfrage an den Authorization Endpunkt umfasst die Schritte 1-3 aus dem Gesamtablauf des Kapitels 3.2. Der Nutzer ruft sein Anwendungsfrontend auf. Die Addressierung des IdP-Dienstes ist im Anwendungsfrontend als Parameter in einer Konfigurationsdatei oder direkt im Quellcode hinterlegt.

Das Anwendungsfrontend liefert seine Anfrage auf ein "ACCESS_TOKEN" über das Authenticator-Modul an den Authorization Endpunkt.

Inhalt der Anfrage ist:

- die "REDIRECT_URI" sowie Bezeichnung des aufzurufenden Fachdienstes,
- die eigene Hersteller-ID, Programmkürzel und Versionsnummer,
- der über das eigene "CODE_VERIFIER" [RFC7636 # section-4.1] gebildete HASH "code_challenge" [RFC7636 # section-4.2] mit Angabe des Algorithmus "code_challenge_method" [RFC7636 # section-4.3],

- der "STATE"-Parameter [RFC8252 # section-8.9] wird genutzt, um CSRF (Cross-Site-Request-Forgery) zu verhindern.

3.8 Aufgaben des Authorization-Endpunktes

Der Authorization-Endpunkt nimmt die Anfrage an und entschlüsselt diese mit seinem privaten Schlüssel "PRK_AUTH". Nach der Signatur- und Integritätsprüfung überprüft der Authorization-Endpunkt, ob mit den Attributen in der"ACCESS_TOKEN"-Anfrage die im Claim des Fachdienstes geforderten Parameter bedient werden können.

3.8.1 Unzureichende Attribute für das Claim

Kann das Claim nicht voll bedient werden, gibt der Authorization-Endpunkt eine Fehlermeldung gemäß [RFC6749 # section-5.2] und fordert den Nutzer zur erneuten Authentisierung und Freigabe der erforderlichen Attribute auf.

3.8.2 Erstellung des AUTHORIZATION_CODE

Sind alle im Claim geforderten Attribute vorhanden und die Gültigkeit der Attribute geprüft, erstellt der Authorization-Endpunkt einen "AUTHORIZATION_CODE" und sendet diesen an das Anwendungsfrontend. Der Authorization Endpunkt prüft die Signatur der "CHALLENGE" und das mitgelieferte Zertifikat der Smartcard des Nutzers gegen den OCSP/TSL-Dienst der PKI der gematik.

3.9 Einreichen des AUTHORIZATION_CODE

Das Anwendungsfrontend reicht den "AUTHORIZATION_CODE" zusammen mit dem "CODE_VERIFIER" beim Token-Endpunkt ein.

3.10 Aufgabe des Token-Endpunktes

Der Token-Endpunkt des IdP-Dienstes nimmt die Daten des Anwendungsfrontends entgegen und prüft neben deren Integrität, ob der eingereichte "CODE_VERIFIER" bei Nutzung des Hash-Verfahrens S256 (nach [RFC7636 # section-4.2]) zum bitgleichen Hash-Wert führt. Stimmt der Hash-Werte aus dem initialen Aufruf des Authenticator-Moduls - die "CODE_CHALLENGE" - mit dem gebildeten Hash-Wert überein, ist sichergestellt, dass Aufrufer und Initiator identisch sind. Der Token-Endpunkt gibt daraufhin das"ID_TOKEN" und das "ACCESS_TOKEN" an das Anwendungsfrontend heraus.

3.11 Einreichen des "ACCESS_TOKEN" beim Fachdienst

Um schlussendlich Zugriff auf den Fachdienst zu bekommen, reicht das Anwendungsfrontend das "ACCESS_TOKEN" beim Fachdienst ein.

3.12 Aufgabe des Fachdienstes

Der Fachdienst nimmt das "ACCESS_TOKEN" entgegen. Der Fachdienst muss das "ACCESS_TOKEN" mit seinem privaten Schlüssel "PrK_FD" entschlüsseln. Danach überprüft er die Integrität und die Übereinstimmung mit dem eigenen Claim. Enthält das "ACCESS_TOKEN" mehr oder weniger Attribute, als im Claim vereinbart, oder sind diese fehlerhaft oder nicht befüllt, stimmt die Integrität oder Signatur des "ACCESS_TOKEN" nicht oder ist das "ACCESS_TOKEN" zeitlich nicht mehr gültig, bricht der Fachdienst die Kommunikation mit einer dem Abbruchgrund entsprechenden Fehlermeldung ab.

Bei positiver Validierung gewährt der Fachdienst Zugriff auf seine Fachdaten.

4 Zerlegung des Produkttyps

Der Produkttyp besteht aus einer zentralen Komponente (IdP-Dienst). Diese wird bei der Durchführung des Authentifizierungsprozesses vom Authenticator-Modul unterstützt. Das Authenticator-Modul übernimmt die Ausführung der Nutzerauthentisierung. Bei Verwendung eines stationären Endgerätes mit installiertem Primärsystem, realisiert das Primärsystem die Funktionalität des Authenticator-Moduls. Das Anwendungsfrontend ist ebenso als Teil des Primärsystems realisiert. Bei Verwendung eines mobilen Endgeräts, ist dort sowohl das Authenticator-Modul, als auch das Anwendungsfrontend gemeinsam in einer Applikation installiert.

~~**A_19878 - Absicherung des Externen Discovery Document im Internet mit TLS**~~
~~Der IdP-Dienst MUSS, für die HTTPS-Schnittstellen im Internet, Extended Validation TLS-Zertifikate eines Herausgebers gemäß [CAB-Forum] verwenden.**[<=]**~~

~~**A_19879 - Externes Discovery Document - Prüfung vor Veröffentlichung**~~
~~Der IdP-Dienst MUSS alle von ihm angebotenen URI betreiben und URI anderer Dienste, insbesondere Fachdienste, vor deren Veröffentlichung im public Discovery Document auf bloße Erreichbarkeit prüfen.**[<=]**~~

Der IdP-Dienst stellt die zentralisierte Identitätsprüfung der auf die Fachdienste zugreifenden Nutzer bereit. Als weitere Teile der Gesamtlösung sind neben dem IdP-Dienst die Clients (Anwendungsfrontend/Primärsystem) und die Fachdienste zu nennen, auf denen Fachdaten für den Zugriff durch die Nutzer (z. B. Versicherte oder Bediener eines AVS, PVS oder KVS) bereitgestellt werden.Ein IdP-Dienst bietet Fachdiensten seine Dienste an, auf welche Millionen Nutzer zeitgleich zugreifen. Eine wesentliche Ergänzung des IdP-Dienstes ist das Authenticator-Modul, welches auf den dezentralen Komponenten in den Praxen, Kliniken, Apotheken und bei den Versicherten betrieben wird.

A_20687~~A_19880~~ - Bereitstellung der PUK
Der Authorization Server MUSS zu allen verwendeten privaten Schlüsseln ~~"*_PRK~~"PrK_AUTH", "PrK_TOKEN" und "PrK_DISC" das öffentliche Pendant ~~"*_~~"PUK_AUTH", "PUK_TOKEN" und "PUK_DISC" zum Download ~~bereitstehen~~bereitstellen. Dies ermöglicht die Prüfung der von den einzelnen Schnittstellen vorgenommenen Signaturen ebenso wie die zielgerichtete Verschlüsselung des Payloads für den bestimmten Empfänger. **[<=]**

A_20732 - Aufnahme der öffentlichen Schlüssel in das Discovery Document
Der Authorization Server MUSS zu jedem privaten Schlüssel dessen öffentlichen Teil mit einer eigenen absoluten URI in das Discovery Document aufnehmen.**[<=]**

Hinweis: Die Bereitstellung von öffentlichem Schlüsselmaterial bezieht sich auf die Schlüssel zum Signieren und ggf. Verschlüsseln der JSON Web Token. Hiermit sind nicht die öffentlichen Schlüssel der TLS-Verschlüsselung gemeint.

A_20686~~A_19895~~ - Erweiterte Nutzung von Schlüsseln
Der Authorization Server MUSS die einzelnen Schnittstellen (AUTH, DISC, TOKEN~~, INT, REV, INFO~~) mit getrennten Interfaces bedienen.~~[<= Die Verwendung des identischen Schlüsselmaterials für mehrere Schnittstellen ist nicht zulässig. **[<=]**~~

4.1.1 Allgemeine Sicherheitsanforderungen

A_20582 - IdP-Dienst - Berücksichtigung OWASP-Top-10-Risiken
Der IdP-Dienst MUSS Maßnahmen zum Schutz vor den zum Zulassungszeitpunkt aktuellen OWASP-Top-10-Risiken umsetzen.[<=]

4.1.2 Sicherheit der Netzübergänge

Der IdP-Dienst wird für Versicherte über das Internet erreichbar gemacht und für Leistungserbringer über das Netz der TI. Die folgenden Anforderungen beschreiben die für diese Netzübergänge erforderlichen Sicherheitsmechanismen. Für den Netzübergang aus dem Internet als Transportnetz zum IdP-Dienst ist ein Paketfilter erforderlich.

A_20583 - IdP-Dienst – Sicherung zum Transportnetz Internet durch Paketfilter
Der Anbieter des IdP-Dienstes MUSS dafür sorgen, dass das Transportnetz Internet durch einen Paketfilter (ACL) gesichert wird und ausschließlich die erforderlichen Protokolle weiterleitet. Der Anbieter des IdP-Dienstes MUSS dafür sorgen, dass der Paketfilter des IdP-Dienstes frei konfigurierbar auf der Grundlage von Informationen aus OSI-Layer 3 und 4 ist, das heißt Quell- und Zieladresse, IP-Protokoll sowie Quell- und Zielport.[<=]

A_20584 - IdP-Dienst – Platzierung des Paketfilters Internet
Der Anbieter des IdP-Dienstes DARF den Paketfilter des IdP-Dienstes zum Schutz in Richtung Transportnetz Internet NICHT physisch auf dem vorgeschalteten TLS-terminierenden Load Balancer implementieren.[<=]

A_20585 - IdP-Dienst – Richtlinien für den Paketfilter zum Internet
Der Paketfilter des IdP-Dienstes MUSS die Weiterleitung von IP-Paketen an der Schnittstelle zum Internet auf das HTTPS- Protokoll beschränken.[<=]

A_20586 - IdP-Dienst – Verhalten bei Vollauslastung
Der Paketfilter des IdP-Dienstes MUSS so konfiguriert sein, dass bei Vollauslastung der Systemressourcen im IdP-Dienst keine weiteren Verbindungen angenommen werden.[<=]

Hinweis: Durch die Zurückweisung von Verbindungen wird sichergestellt, dass Clients einen Verbindungsaufbau mit einer anderen Instanz des Fachdienstes versuchen, bei dem die erforderlichen Ressourcen zur Verfügung stehen.

A_20587 - IdP-Dienst – Richtlinien zum TLS-Verbindungsaufbau
Der Anbieter des IdP-Dienstes MUSS dafür sorgen, dass der Eingangspunkt des IdP-Dienstes sich beim TLS-Verbindungsaufbau über das Transportnetz gegenüber dem Client mit einem Extended Validation TLS-Zertifikat eines Herausgebers gemäß [CAB-Forum] authentisiert. Der Anbieter MUSS dafür sorgen, dass das Zertifikat sich an die jeweilige Schnittstelle des Eingangspunkts für Primärsysteme, Authenticator-Module und Frontends der Versicherten des IdP-Dienstes bindet, damit Clientsysteme beim TLS-Verbindungsaufbau eine vereinfachte Zertifikatsprüfung mit TLS-Standardbibliotheken durchführen können.[<=]

4.2 Fehlermeldungen

~~4.21.1 Fehlermeldungen~~

~~Entstehen während eines Verarbeitungsprozesses Fehler, muss der jeweilige Teil-Dienst diese sofort protokollieren und melden.~~

A_20680~~A_19896~~ - Format der Fehlermeldungen
Der IdP-Dienst MUSS für die verschiedenen Teilfunktionen geeignete Fehlermeldungen erzeugen und diese an den jeweiligen Aufrufer übergeben.**[<=]**

A_20681 - Nutzung von eindeutigen Error-Codes bei der Erstellung von Fehlermeldungen
~~diese ohne zeitlichen Verzug in Form eines UDP-Multicast an die von der gematik bestimmten Ziele übermitteln.~~
Der IdP-Dienst MUSS Fehler durch eine eindeutige Nummer erkennbar machen und der gematik eine Liste der Error-Codes zur Verfügung stellen, damit die Ursachenklärung vereinfacht möglich wird.**[<=]**

~~Es folgt ein Beispiel einer möglichen Fehlermeldung, welche vom IdP-Dienst für alle durch ihn bereitgestellten Funktionen in ähnlicher Weise und gleicher Qualität umgesetzt werden MUSS.~~

A_20682 - Verwendung eines einheitlichen Schemas für die Aufbereitung von Fehlermeldungen
~~A_19899 - Fehlermeldungen sind nutzerfreundlich und basieren einheitlich auf UTC~~Der IdP-Dienst MUSS alle ~~vom verwendeten Standard~~ ausgeworfenen Fehlermeldungen zur Weiterverarbeitung in einem einheitlichen Schema aufbereiten und bereitstellen. Zeitstempel MÜSSEN auf der UTC basieren.
Beispiel: Ist eine Signatur nicht vorhanden~~,~~ oder defekt ~~oder stimmt die URI des Absenders nicht mit der vom Authenticator registrierten URI überein~~, bricht der Authorization-Endpunkt die Bearbeitung mit dem registrierten Fehlercode und einer für den Nutzer verständlichen Fehlermeldung ab.

Tabelle 4 TAB_IDP_DIENST_0004 Schema der Fehlermeldungen

Fehlermeldungscode	Fehlermeldungstext
IDPD_1001.1: 1583844803	*Signature Consent: fehlende Signatur. [10.03.2020 13:53:23]*
IDPD_1001.2: 1583844803	*Signature Consent: falsche URI. [10.03.2020 13:53:23]*
IDPD_1001.3: 1583844803	*Signature Consent: falscher Algorithmus. [10.03.2020 13:53:23]*

[<=]

A_20683 - Formulierung der Fehlermeldungen
Der IdP-~~Dient~~Dienst MUSS Fehlermeldungen, welche dem Nutzer angezeigt werden, in der Art ausformulieren, dass es dem Nutzer möglich ist, eigenes Fehlverhalten anhand der Fehlermeldung abzustellen.**[<=]**

A_20684 - Nutzung einer eindeutigen Beschreibung beim Aufbau von Fehlermeldungen
Der IdP-Dienst MUSS jedem Fehler eine eindeutige eigene Beschreibung zukommen lassen, sodass eine Fehlermeldung nicht für unterschiedliche Fehlerursachen zur Anwendung kommt.[<=]

A_20685 - Ausgabe der Fehlermeldungen in umgekehrter Reihenfolge des Auftretens
Der IdP-Dienst MUSS aufeinander aufbauende Fehlermeldungen in der umgekehrten Reihenfolge ihres Auftretens "Traceback (most recent call last)" ausgeben.
[<=]

4.3 Schnittstellenbeschreibung des IdP-Dienstes

Der IdP-Dienst ~~muss~~bietet zahlreiche Schnittstellen gegenüber unterschiedlichen Akteuren inner- und außerhalb der TI ~~anbieten~~an, weswegen es notwendig ist, die einzelnen Schnittstellen so zu beschreiben, dass andere Akteure deren Funktionsweise leichter verstehen können. Nachfolgende Abbildung skizziert die Schnittstellen des IdP-Dienstes. Komponenten und Schnittstellen, welche nicht direkt vom IdP-Dienst genutzt werden, sind in der Abbildung grau hinterlegt.

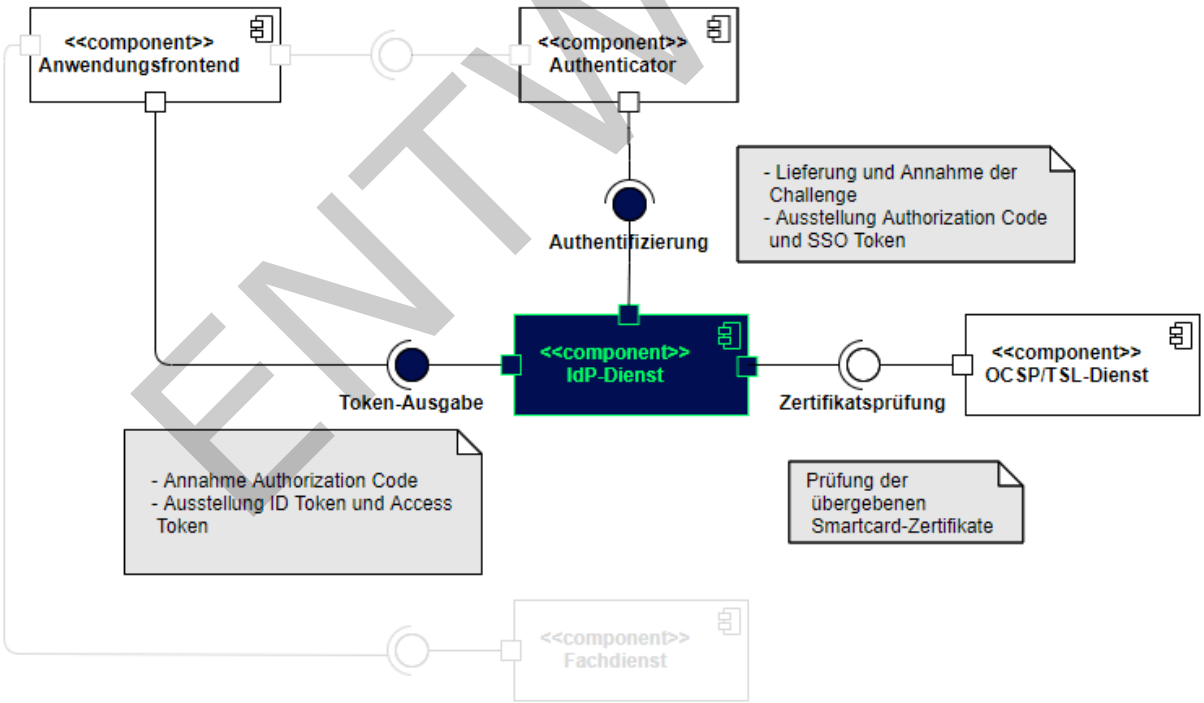


Abbildung 5 Schnittstellen des IdP-Dienstes

Die erste tokenbezogene Anfrage an den Authorization Server des IdP-Dienstes geht am Authorization-Endpunkt [RFC6749 # section-3.1] ein.

~~Hier reicht der~~ Das Authenticator-Modul reicht dort am Endpunkt den `"CONSENT"` mit der `"CHALLENGE"` ein, mit welchem ~~das `"ID`~~ die `"TOKEN"` erstellt werden ~~soll~~sollen, und

erhält den "~~ACCESS_~~AUTHORIZATION_CODE" zurück~~.~~, falls die Prüfung der signierten "CHALLENGE" und die Prüfung des übergebenen Smartcard-Zertifikats am OCSP/TSL-Dienst positiv ausfallen. Das Anwendungsfrontend reicht den "AUTHORIZATION_CODE" am Token Endpunkt [RFC6749 # section-3.2] des IdP-Dienstes ein. Der IdP-Dienst überprüft den "AUTHORIZATION_CODE" und stellt bei positiver Validierung einen "ID_TOKEN" und einen "ACCESS_TOKEN" aus.

Bei der ersten Kontaktaufnahme erzeugt der Authorization Server die "SUBJECT_SESSION", welche im weiteren Verlauf als Zeitpunkt der letzten Authentisierung gegen die eGK oder den ~~eHBA~~HBA gewertet wird. Basierend darauf dürfen weitere "~~ID~~ACCESS_TOKEN" und "~~REFRESH~~SSO_TOKEN" für andere ~~Anwendungsfrontend~~Anwendungsfrontends und Fachdienste ausgegeben werden, wenn das jeweils vorliegende Claim durch die dem Authorization Server vorliegenden Informationen bedient werden kann~~ und die PIN-Eingabe der letzten Authentifizierung zeitlich noch brauchbar ist.~~. Ist der Zeitpunkt der letzten Authentisierung zu lange her oder wird ~~der~~das Authenticator-Modul zum ersten Mal gestartet, muss eine Authentisierung erfolgen.

~~*Hinweis:* Ob und wenn ja wie lange eine Überprüfung des Zusammentreffens von Besitz (Smartcard) und dazugehöriger PIN (Wissen) nachgenutzt werden kann, muss der Anbieter eines Fachdienstes unter Bezugnahme seiner Anforderungen bezüglich Sicherheit beurteilen.~~

Hinweise für die Implementierung der Authentifizierung für Primärsystemen werden in [gemILF_PS_eRp] beschrieben.

Der Vorgang der Authentifizierung gegen die eGK oder den ~~eHBA~~HBA ist nicht Bestandteil dieser Spezifikation, sondern ist im gesonderten Dokument [gemSpec_IDP_Frontend] beschrieben~~, da sich die daraus hervorgehenden Anforderungen an den mobilen Teil auf dem Endgerät des Nutzers richten~~.

4.4 Identifikation des Clientsystems

Der IdP-Dienst verwaltet und steuert den Authentisierungsprozess für das E-Rezept und perspektivisch auch weitere Anwendungen. Damit kommt ihm eine Relevanz in der Gesundheitsversorgung zu, die sich zum einen in einer hohen Verfügbarkeit und zum anderen in einem hohen Angriffspotential widerspiegelt. Zur Unterstützung der betrieblichen Überwachung des IdP-Dienstes wird die Nutzung der im Feld befindlichen Clientsysteme protokolliert. Dabei ist der Zugriff auf die Schnittstellen des IdP-Dienstes nur durch Primärsysteme der Leistungserbringer, sein eigenes Authenticator-Modul und zugelassene E-Rezept-FdVs zulässig. Der E-Rezept-Fachdienst erkennt die Clientsysteme anhand des User-Agent-Header eingehender HTTP-Requests und protokolliert diesen Wert.

A_20588 - IdP-Dienst - Erkennung Clientsystem User-Agent
Der IdP-Dienst MUSS das vom aufrufenden Nutzer verwendete Clientsystem (Authenticator-Modul, E-Rezept-FdV oder Primärsystem) anhand des im HTTP-Request enthaltenen Header-Feld "User-Agent" gemäß [RFC7231] erkennen und in den Einträgen zur Performance-Rohdatenerfassung gemäß [gemSpec_Perf] protokollieren. Der IdP-Dienst MUSS bei fehlendem User-Agent-Header den Request mit dem HTTP-Status-Code 400 beantworten, damit in der Betriebsüberwachung des IdP-Dienstes die Nutzung unzulässiger Clientsysteme erkannt werden kann.**[<=]**

A_20589 - IdP-Dienst – Ausschluss bestimmter Clientsystem-Versionsnummern von der Kommunikation
Der IdP-Dienst MUSS die aus dem Internet vom Clientsystem mitgeteilte Versionsnummer aus dem HTTP-Header User-Agent, erkennen und festgelegte Versionsnummern über ein Blacklisting von einer Kommunikation mit dem IdP-Dienst ausschließen können. Der IdP-Dienst MUSS in diesen Fällen eine entsprechende Fehlermeldung an das Clientsystem geben. [<=]

A_20590 - IdP-Dienst – Ausschluss von Clientsystem-Versionen
Der Anbieter des IdP-Dienstes MUSS ausschließlich auf Anweisung der gematik Clientsysteme mit bestimmten Versionsnummern von einer Kommunikation mit dem IdP-Dienst ausschließen.[<=]

A_20742 - Vergabe der "client_id" durch den IdP-Dienst
Der IdP-Dienst MUSS bei der organisatorischen Registrierung des Anwendungsfrontends diesem eine eindeutige "`client_id`" zur Nutzung des IdP-Dienstes zuweisen.
[<=]

~~4.41.1 Begriffsdefinition~~

~~Die folgende Tabelle enthält die Abkürzungen (für die privaten Schlüssel PrK und für öffentliche Schlüssel PUK) der verschiedenen Akteure und deren Verwendung, wobei diese Informationen für den Authorization Server nicht angegeben sind, da hier ausschließlich die URI des Downloadpunktes des Discovery Documents hinterlegt wäre.~~

> ~~*Speziell die Verschlüsselung der einzelnen Artefakte befindet sich noch in Diskussion. Einige der aktuell vorgesehenen Verschlüsselungen können womöglich entfallen und damit auch eine Reihe der jetzt benannten Schlüssel.*~~

~~-~~	~~PUK~~	~~URI PUK~~	~~privateKey~~	~~URI Dienst~~
~~Authenticator (AUTH_APP)~~	~~PUK_APP~~	~~PUK_URI_APP~~	~~PrK_APP~~	~~URI_APP~~
~~Anwendungsfrontend (FRONT)~~	~~PUK_FRONT~~	~~PUK_URI_FRONT~~	~~PrK_FRONT~~	~~URI_FRONT~~
~~Authorization Server~~	~~-~~	~~-~~	~~-~~	~~URI_DD~~
~~Authorization-Endpunkt (AUTH)~~	~~PUK_AUTH~~	~~PUK_URI_AUTH~~	~~PrK_AUTH~~	~~URI_AUTH~~
~~Discovery-Endpunkt (DISC)~~	~~PUK_DISC~~	~~PUK_URI_DISK~~	~~RrK_DISC~~	~~URI_DISC~~
~~Token-Endpunkt (TOKEN)~~	~~PUK_TOKEN~~	~~PUK_URI_TOKEN~~	~~PrK_TOKEN~~	~~URI_TOKEN~~
~~Introspection-Endpunkt (INT)~~	~~PUK_INT~~	~~PUK_URI_INT~~	~~PrK_INT~~	~~URI_INT~~

Revocation-Endpunkt (REV)	PUK_REV	PUK_URI_REV	PrK_REV	URI_REV
Userinfo-Endpunkt (INFO)	PUK_INFO	PUK_URI_INFO	PrK_INFO	URI_INFO

Die URI des Discovery Document "URI_DD" stellt somit den zentralen Anlaufpunkt dar, anhand dessen alle weiteren „statischen" Dienste (Endpunkte des IdP-Dienstes und Fachdienste) adressiert werden können. Alle dynamisch registrierten Akteure (Primärsysteme, Endgerät des Nutzers, Authenticator und Frontend des Versicherten) werden bei der ersten Registrierung am Authorization-Endpunkt gespeichert. Im späteren Verlauf sind deren URI immer Bestandteil der Redirection des signierten Tokens bzw. Access Codes, wodurch eine Adressierung über einen Hilfsdienst nicht notwendig ist.

4.5 Registrierung von Primärsystem, Endgerät und Anwendungsfrontend

A_19882 - Endgeräte und Anwendungsfrontend Registrierung
Alle Endgeräte (Smartphones, Tablets), Anwendungen und Endgeräte, Anwendungsfrontends (Authenticator oder E-Rezept-Applikation auf einem Smartphone, Tablet oder Primärsystem) des Nutzers MÜSSEN sich beim Authorization Server am Authorization-Endpunkt gemäß [openid-heart-oauth2-1_0.# rfc.section.3.1.3 und RFC7591] registrieren.**[<=]**

A_19883 - Anwendungsfrontend gibt sich zu erkennen (Anforderung der gematik)
Das Anwendungsfrontend und Primärsysteme, welche über den Authenticator am Authorization-Endpunkt die Beantragung eines Tokens anstoßen, MÜSSEN dabei die <Hersteller-ID>, die interne Programmkennung <Produktkürzel> und die aktuell installierte Version in Form einer nachvollziehbaren Versionsnummer <Version.Major.Minor.Build> mitteilen.**[<=]**

Beispiel:

```
"<Programmbezeichnung>-1.3.15.125634"
```

A_19884 - Voraussetzung der dynamischen Registrierung (Service Discovery)
Anwendungsfrontends, Primärsysteme und Fachdienste MÜSSEN im Zuge der Registrierung das Discovery Document (DD) [RFC8414] einlesen und auswerten und danach die darin aufgeführten URI zu den benötigten PUK's und Diensten verwendet werden. Dem Anwendungsfrontend, Primärsystem und Fachdiensten MUSS der im Fachportal der gematik veröffentlichte Downloadpunkt des Discovery Document hardgecoded als Parameter für die Anwendung vorgegeben werden. **[<=]**

A_19892 - Dynamische Registrierung (Authenticator)

Der Authenticator MUSS bei der dynamischen Registrierung neben seiner absoluten URI "URI_APP" auch die URI seines öffentlichen Schlüssels "URI_PUK_APP" sowie die Versionsnummer und Bezeichnung des anzumeldenden Authenticators einreichen.**[<=]**

A_19893 - Dynamische Registrierung (Anwendungsfrontend)
Das Anwendungsfrontend und Primärsysteme MÜSSEN bei der dynamischen Registrierung neben der absoluten URI "URI_FRONT" auch die URI des öffentlichen Schlüssels "URI_PUK_FRONT" sowie die Versionsnummer und Bezeichnung des anzumeldenden Anwendungsfrontend einreichen.[<=]

A_19894 - Dynamische Registrierung (Absicherung durch TLS)
Anwendungsfrontend und Authenticator MÜSSEN bei der Übertragung von Daten im Zuge der dynamischen Registrierung diese mit dem öffentlichen Schlüssel des Authorization-Endpunktes "PUK_AUTH" verschlüsselt und zusätzlich serverseitig TLS-gesichert senden. [<=]

Der "PUK_AUTH" ist über die entsprechende URI "URI_PUK_AUTH" aus dem Discovery Document zu beziehen.

A_19854 - Zwischenspeichern des Discovery Documents
Der Authenticator und das Anwendungsfrontend sowie Fachdienste SOLLEN das Discovery Document zwischenspeichern.[<=]

Die URI der Downloadpunkte der PUK "URI_PUK_FD" der angebotenen Fachdienste ändern sich generell nicht. Die Inhalte der Discovery Documents ändern sich nur bei der Registrierung neuer Fachdienste oder bei Änderung derer URI's. Durch das Zwischenspeichern soll verhindert werden, dass das Discovery Document von den mehrere Millionen Endgeräten unnötig oft heruntergeladen wird.

Hinweis:

Ändert sich die URI eines Fachdienstes "URI_FD" oder die URI seines öffentlichen Schlüssels "URI_PUK_FD" dennoch, muss der Fachdienst die Änderung dem IdP-Dienst mitteilen. Die Mitteilung erfolgt durch erneute Registrierung des Fachdienstes beim IdP-Dienst, wobei die vorhergehende Registrierung als gelöscht zu markieren ist.

A_19855 - Inhalt der Fachdienste im Discovery Document
Fachdienste MÜSSEN im Discovery Document die URI des Fachdienstes "URI_FD", des TLS-gesicherten Bereitstellungspunkts des öffentlichen Schlüssels "URI_PUK_FD" bereitstellen und angeben, welche Algorithmen von ihnen unterstützt werden.[<=]

Das Discovery Document enthält weitere Informationen, welche den IdP-Dienst betreffen.

A_19954 - Einbindung des Primärsystems
Das Primärsystem (PVS, AVS und KVS) MUSS eine gesonderte Schnittstelle implementieren, welche die Funktionalität des Authenticators übernimmt und die eigentlich mit der Smartcard durchgeführte Challenge-Response-Verfahren bei mindestens gleichwertiger Qualität abbildet.[<=]

Hinweis: Die Aufgabe "Challenge" wird durch den Konnektor mittels Zugriff auf die im Kartenterminal gesteckte und bereits freigeschaltete SMC-B des Leistungserbringers signiert und die Aufgaben-Lösung "Response" über das Primärsystem an den IdP-Dienst zurück übertragen.

A_19956 - Primärsysteme Herkunft des Schlüsselmaterials (JWT, JWE, JWS)

~~**5 Primärsysteme MÜSSEN das für die JSON-Verschlüsselung und Signatur benötigte Schlüsselmaterial "PRK_APP" und "PUK_APP" gemäß [gemSpec_PKI] selbst erzeugen. Algorithmen und Hashverfahren sind gemäß [gemSpec_Krypt] anzuwenden.[<=]**~~

6 Funktionsmerkmale

6.1 Authorization Server Metadata (Discovery Document)

Der Authorization Server ~~ist eine künstliche Metaebene, um~~dient dazu bestehende Identitäten zu prüfen und das Prüfungsergebnis in einer einheitlichen Form abgestimmt und durch zusätzliche Mechanismen gesichert bereitzustellen. Basis dieser Dienstleistung ist ein vertrauenswürdiges Verzeichnis, aus welchem hervorgeht, an welchen Schnittstellen dieser Dienst oder seine Teildienste erreichbar ~~ist~~sind, wie diese Schnittstellen abgesichert sind und woher man die zur Etablierung der gewünschten Sicherheit erforderlichen Materialien beziehen kann. Gemäß dem verwendeten Standard ~~OpenIDConnect~~OpenID Connect mit OAuth 2.0 kommen JSON Web Token (JWT), JSON Web Encryption (JWE), JSON Web Signature (JWS) und JSON Web Key (JWK) zum Einsatz. ~~Das Adressieren der Fachdienste und serverbasierten Systeme ist einfach, da diese, sobald sie im Discovery Document eingetragen sind, allen Interessierten zugänglich sind.~~

~~Damit auch bisher nicht im Discovery Document eingetragene Akteure adressierbar werden, müssen diese dem Authorization Server bekannt gemacht werden. Bisher unbekannte Entitäten können zur Laufzeit anhand einer dynamischen Registrierung nachträglich bekannt gemacht werden und bekommen so die Möglichkeit, auf bereits bestehende Fachdienste zuzugreifen.~~

~~Damit ein vorher nicht registriertes Endgerät oder dessen Anwendungsfrontend in das Verzeichnis adressierbarer Entitäten aufgenommen werden können, müssen sich diese beim Authorization Server anmelden bzw. bei der ersten Kontaktaufnahme registrieren. Im Verlauf der Registrierung wird ein sogenannter Authenticator installiert, dessen Aufgabe es ist, die Kommunikation mit bestimmten Schnittstellen des IdP-Dienstes zu übernehmen und den Datenaustausch an die formalen Erfordernisse anzupassen.~~

Um nutzenden Anwendungen eine einheitliche Bezugsquelle für die Adressierung von Schnittstellen zu schaffen, werden die für alle Akteure grundlegenden Schnittstellen im sogenannten Discovery Document zusammengefasst und dort unter der `"URI_DISC"` gemäß ~~[~~[RFC8414 "OAuth 2.0 Authorization Server Metadata"] veröffentlicht.

Alle Akteure, welche den IdP-Dienst nutzen wollen, sind angehalten, dieses Discovery Document zu lokalisieren, herunterzuladen, zu prüfen und den Inhalt in den geplanten Betrieb einzubeziehen.

> ~~*Im aktuellen Stand des Dokumentes fehlen noch in Diskussion befindliche, Festlegungen zur Erweiterung des Integritätsschutzes des Discovery Documents sowie Ergänzungen zum Transport von Schlüsselmaterial über das Discovery Document. Die Standards sehen Verschlüsselungen vor, aber lassen die Methoden des Schlüsselaustausch offen und die gematik ist noch in Abstimmung zu praktikablen Methoden.*~~

A_20457 - Verwendung eindeutiger URI
Der IdP-Dienst MUSS alle verwendeten Adressen in Form von URL gemäß [RFC1738] angeben und in einem Discovery Document gemäß [RFC8414 # section-2] innerhalb der TI und im Internet veröffentlichen.**[<=]**

A_20688 - Discovery Document interne und externe Adressierung
Der Discovery-Endpunkt MUSS die Discovery Documents für interne und externe Adressierung sowohl innerhalb der TI als auch im Internet veröffentlichen.**[<=]**

Das Discovery Document innerhalb der TI adressiert hierbei die URI der Fachdienste und Schnittstellen des IdP-Dienstes innerhalb der TI. Das im Internet bereitgestellte Discovery Document stellt die URI der angebotenen Fachdienste im Internet mit dort auflösbaren Adressen bereit.

Hinweis: Es gibt je ein internes und externes (public) "Discovery Document". Diese unterscheiden sich in den darin angebotenen URI, welche gleichlautend im Host-Anteil auf unterschiedliche Domänen bzw. Top-Level-Domain (TLD) verweisen.

A_20689 - Internes Discovery Document - Prüfung vor Veröffentlichung
Der IdP-Dienst MUSS alle von ihm im internen Discovery Document angebotenen URL und URL anderer Dienste, insbesondere Fachdienste, vor deren Veröffentlichung im internen Discovery Document auf bloße Erreichbarkeit prüfen.**[<=]**

A_20690 - Externes Discovery Document - Prüfung vor Veröffentlichung
Der IdP-Dienst MUSS alle von ihm angebotenen URI betreiben und URI anderer Dienste, insbesondere Fachdienste, vor deren Veröffentlichung im externen Discovery Document auf bloße Erreichbarkeit prüfen.**[<=]**

6.1.1 Aufbau des Discovery Documents

Der Authorization Server muss das Discovery Document gemäß ~~[~~[RFC8414] bereitstellen ~~und dessen sichere Umsetzung erweitert durch gematik-spezifische Erweiterungen um [HEART I] & [HEART II] verwirklichen~~.

A_20439~~A_20145~~ - Das Discovery Document enthält statische Adressen
Der Discovery-Endpunkt MUSS ~~im~~sowohl im internen, als auch im externen Discovery Document die Akteure mit ~~statischen IP-Adressen und~~ ihrer URI veröffentlichen.
[<= ~~IP-Adressen bzw. URI der dynamisch registrierten Akteure MUSS der Discovery-Endpunkt in einer Datenbank abspeichern und von dort auf gezielte Anfrage bereitstellen.~~
~~[<=~~]

~~**A_19909 - Integrität der Eingangsdaten am Authenticator-Modul**~~

~~Der IdP-Dienst MUSS die Zusammenführung von Authenticator-Modul und Anwendungsfrontend gemäß [RFC7591] im Rahmen der dynamischen Registrierung der Clients organisieren.**[<=]**~~

A_20458 - Inhalte des Discovery Documents
~~**A_20146 - Redirection-Endpunkt (Herausgabe von Informationen an Redirection-Endpunkt)**~~Der Discovery-Endpunkt MUSS ~~auf gezielte Anfragen des Redirection-Endpunktes diesem die Redirection URI des Anwendungsfrontend sowie die Redirection URI des Authenticators mitteilen, wenn diese registriert und in der Datenbank aufgeführt sind.**[<=]**~~

~~**A_20147 - Redirection-Endpunkt (Verantwortlichkeit für Aktualität)**~~sowohl im internen, als auch im externen~~Der~~ Discovery~~ Endpunkt MUSS die bei der dynamischen Registrierung genutzten URI der Akteure wiedergeben, ist jedoch für die Aktualität dieser Informationen nicht verantwortlich.**[<=]**~~

Document gemäß [RFC8414 # section-2] mindestens die folgenden Attribute als URI angeben:

- "iss" (hier ist der IdP-Dienst erreichbar)
- "jwks_uri" (für den Abruf der/des PUK des Authorization Server [RFC7517])

"URI_DISC" (URI, unter welcher das ~~Der Redirection-Endpunkt hat keinen Einfluss darauf, ob ein dynamisch registrierter Akteur noch erreichbar ist, und kann daher nur die ihm bekannten Informationen veröffentlichen.~~

~~**A_20148 - Discovery Document (Datenbasis Anwendungsfrontend autoclean)**~~
~~Der Discovery-Endpunkt MUSS die dynamisch registrierten Anwendungsfrontend aus der Datenbank löschen, wenn die zugrundeliegende SUBJECT_SESSION abgelaufen ist.~~**~~[<=]~~**

- ~~**A_20149 -** Discovery Document **(Datenbasis Authenticator autoclean)**~~bereitgestellt ist)
- ~~Der Discovery-Endpunkt MUSS die dynamisch registrierte URI des Authenticators aus der Datenbasis löschen, wenn die Redirection einer Anfrage durch ein Anwendungsfrontend fehlschlägt oder die bei der Registrierung angegebene URI nicht reagiert.~~
  ~~[<=]~~"URI_AUTH" &"PUK_URI_AUTH" URI des Dienstes und des öffentlichen Schlüssels des Authorization-Endpunktes gemäß [RFC6749]
- "URI_TOKEN" &"PUK_URI_TOKEN" URI des Dienstes und des öffentlichen Schlüssels des Token-Endpunktes gemäß [RFC6749]

[<=]

Hinweis: Ein Beispiel eines Discovery Documents kann unter folgendem Link gefunden werden: HEART II # rfc.section.3.1.5

6.1.2 Erneuerung des Discovery Documents

Der Authorization Server muss das Discovery Document mit den Metainformationen zu den Teildiensten mindestens einmal täglich und immer nach Änderungen mit dem "PrK_DISC" ~~frisch~~ signieren und am mit der gematik vereinbarten Downloadpunkt "URI_~~DISK~~"DISC" bereitstellen.

A_20691~~A_20150~~ - Das Discovery Document ist maximal 24 Stunden alt
Der Discovery-Endpunkt MUSS das Discovery Document regelmäßig alle 24 Stunden oder nach durchgeführten Änderungen umgehend neu erstellen, mit dem "PrK_DISC" signieren und und am mit der gematik vereinbarten Downloadpunkt "URI_DISC" bereitstellen.
[<=]

A_20592 - Aktualisierung der Discovery Documente
Der IdP-Dienst MUSS das interne und externe Discovery Document bei Änderungen mit dem "PrK_DISC" neu signieren und am mit der gematik vereinbarten Downloadpunkt"URI_DISC" bereitstellen**[<=]**

6.1.3 Schutz des Discovery Documents

Der Authorization Server ~~muss das~~ schützt die Integrität des Discovery Document auf Dateiebene durch ~~einen darüber gebildeten signierten Hash~~ eine Signatur und während des Transportes zusätzlich ~~TLS-gesichert bereitstellen.~~ mittels TLS.

A_19874-02 - Bereitstellung des internen Discovery Documents innerhalb der TI
Der ~~Authorization Server verwendet für die zusätzliche Signatur des Hash-Wertes das von der gematik bezogene Signaturzertifikat. Die zusätzliche Bereitstellung des signierten Hash-Wertes ist nicht standardkonform und verwendet für die Signatur ein X.509-Zertifikat basierend auf dem Root-CA-Zertifikat der TI.~~IdP-Dienst MUSS das interne Discovery Document mit einem Zertifikat des Typs FD.SIG und der technischen Rolle „oid_idpd" gemäß [gemSpec_Krypt # Abschnitt 3.7] signiert, an einem spezifischen Downloadpunkt TLS-gesichert innerhalb der TI bereitstellen . <=**[<=]**

Hinweis: Das Discovery Document kann unter dem relativen Pfad .well-known/openid-configuration eingelesen werden.

A_19877-01 - Bereitstellung des externen Discovery Documents im Internet
~~**A_20151 - Zusätzlicher Schutz des Discovery Documents**~~Der IdP-Dienst MUSS das externe Discovery Document mit einem Zertifikat des Typs FD.SIG und der technischen Rolle „oid_idpd" gemäß [gemSpec_Krypt # Abschnitt 3.7] signiert und TLS-gesichert im Internet zum Download~~Der Discovery-Endpunkt MUSS neben dem signierten Discovery Document einen darüber gebildeten Hash-Wert, welcher mit dem von der gematik erteilten X.509-Signaturzertifikat zu signieren ist,~~ bereitstellen.
Die URL des Downloadpunktes lautet: "**https://idp.ti-dienste.de/.well-known/openid-configuration**" **[**<=**]**

Hinweis: Die ~~Prüfung der Zertifikatskette erfolgt dabei gegen~~für die Rolle des IdP-Dienstes vorgesehene professionOID ist in [gemSpec_OID] beschrieben. Der externe Downloadpunkt des Discovery Document ist der folgende: idp.ti-dienste.de/.well-known/openid-configuration

A_20591 - Festlegungen zur Signatur der Discovery Documente
Der IdP-Dienst MUSS die Signatur der Discovery Documente als base64-kodierte CMS-Signatur gemäß [RFC5652] realisieren und die Festlegungen aus gemSpec_Krypt#5.6.2 beachten.
Der IdP-Dienst MUSS bei der Signaturerstellung das Signaturzertifikat als Attribut *signing certificate reference* gemäß [CAdES # Kapitel 5.7.3 „Signing Certificate Reference Attributes"] einbetten. **[**<=~~aktuell gültige von der gematik veröffentlichte ROOT-CA-Zertifikat.~~
~~**[**<=~~**]**

6.2 Authorization-Endpunkt

Vorbedingung ist, dass ~~der~~das Authenticator-Modul bereits eine "SUBJECT_SESSION" mit dem Authorization Server etabliert, sich das Discovery Document heruntergeladen und dieses erfolgreich ausgewertet hat.

A_20434 - Einhaltung der Standards bei der Realisierung des Authorization-Endpunkts

~~A_19860 - Der Authorization-Endpunkt Standards~~Der IdP-Dienst MUSS die Schnittstelle „Authorization-Endpunkt" gemäß [RFC6749 "The OAuth 2.0 Authorization Framework~~]~~"] und [RFC8252 „OAuth 2.0 for Native Apps"] und weiteren darin ~~verwiesenen~~festgelegten Standards implementieren.**[<=]**

~~A_19861 - Authorization-Endpunkt Authenticator-Modul~~
~~Der Anbieter des IdP-Dienstes MUSS den Authenticator über den für das jeweilige Betriebssystem üblichen Software-Verteilungspunkt selbst bereitstellen oder bereitstellen lassen.**[<=]**~~

~~A_19862 - Authenticator im Apple App Store~~
~~Der Anbieter des IdP-Dienstes MUSS den Authenticator für das Betriebssystem Apple im dafür vorgesehenen Apple App Store für den Nutzer kostenfrei bereitstellen. **[<=]**~~

A_19863 - Schutz vor überalterter Software (Apple)
Der Anbieter IdP-Dienst MUSS dafür Sorge tragen, dass die im Apple App Store veröffentlichte Software bei Änderungen automatisiert aktualisiert wird, sodass jederzeit die dauerhafte Verwendung fehlerhafter Software ausgeschlossen werden kann.**[<=]**

~~A_19864 - Authenticator im Google Play Store~~
~~Der Anbieter des IdP-Dienstes MUSS den Authenticator für das Betriebssystem Google Android im dafür vorgesehenen Google Play Store für den Nutzer kostenfrei bereitstellen.**[<=]**~~

A_19865 - Schutz vor überalterter Software (Android)
Der Anbieter des IdP-Dienstes MUSS dafür Sorge tragen, dass die im Google Play Store veröffentlichte Software bei Änderungen automatisiert aktualisiert wird, sodass jederzeit die dauerhafte Verwendung fehlerhafter Software ausgeschlossen werden kann.**[<=]**

6.2.1 Authorization Server -Eingangsdaten

~~A_19853 - Protokollierung der Consent-Bestätigung~~
~~Am Authorization-Endpunkt MUSS der IdP-Dienst den vom Nutzer am Authenticator bestätigten `"consent"` protokollieren. Die Bestätigung des Consent wird im Zähler vermerkt und der Zeitstempel für die letzte Consent-Abstimmung im Parameter `"last_consent"` (siehe „Zähler, Zeitstempel und Performance) protokolliert.**[<=]**~~

~~A_19850 - Enschlüsseln der Eingangsdaten am Authorization-Endpunkt~~
~~Der Authorization-Endpunkt entschlüsselt die Daten mit dem zum aktuell im Discovery Document veröffentlichten öffentlichen Schlüssel `"PUK_AUTH"` gehörenden privaten Schlüssel `"PRK_AUTH"`. Ist die Entschlüsselung mit diesem Schlüssel nicht möglich, SOLL der Authorization-Endpunkt versuchen, die Entschlüsselung mit dem privaten Schlüssel der vorhergehenden Generation vorzunehmen.**[<=]**~~

~~**A_19851 - Aufbewahrung alter Schlüssel**~~
~~Der IdP-Dienst MUSS die in den letzten 72 Stunden verwendeten Schlüssel aufbewahren, um eine nachträgliche Entschlüsselung von Daten zu ermöglichen.~~ ~~**[<=]**~~

A_20698 - Annahme des Authorization Request
Der Authorization Endpunkt MUSS die im Authorization Request des Authenticator-Moduls mitgelieferten "CODE_CHALLENGE" und den "SCOPE" annehmen.**[<=]**

Hinweis: Nachfolgend wird beispielhaft der Authorization Request als HTTP GET-Request dargestellt, welcher vom Authenticator-Modul initiiert wird:

```
GET /authresponse_type=code&scope=openid%20e-
rezept&state=af0ifjsldkj&client_id=ZXJlemVwdC1hcHA&redirect_uri=https%3A%2F
%2Fapp.e-
rezept.com%2Fauthnres&code_challenge_method=S256&code_challenge=S41HgHxhXL1
CIpfGvivWYpbO9b_QKzva-9ImuZbt0Is

HTTP/1.1
Host: idp.com
X-E-Rezept-App: 1.0
Accept: application/json
User-Agent: E-Rezept-App/1.0
```

A_20376 - Verwendung des Attributes "state"
Der Authorization-Endpunkt MUSS den vom Anwendungsfrontend initierten "state"-Parameter gemäß [RFC6749 # section-10.12] bei einer Redirection an den Client in seiner Antwort verwenden. **[<=]**

A_20731~~**A_19852**~~ **- Verwendung des Attributes "auth_time"**
Der Authorization-Endpunkt MUSS den Parameter "auth_time" mit dem Zeitpunkt der letzten Authentisierung gegen das zugelassene Authentifizierungsmittel (z.B. Auslösen der Signatur durch Smartcard ~~mit PIN-Eingabe~~in freigeschaltetem Zustand) setzen.**[<=]**

~~**A_19848 - Verwendung des Attributes "Bearer"**~~

~~Der Token-Endpunkt MUSS Token so ausstellen, dass diese die eineindeutige Kennung des Inhabers "bearer" enthalten. [rfc6750].~~ ~~**[<=]**~~

~~Anhand dieser kann zu jeder Zeit die Quelle des Tokens auf der Nutzerseite adressiert werden.~~

~~**A_19849 - ACCESS_CODE und ID_ - oder REFRESH_TOKEN nur für gültige Zertifikate**~~
~~Der Authorization-Endpunkt MUSS vorgetragene Zertifikate des Antragstellers gegen den OCSP-Responder innerhalb der TI auf aktuelle Gültigkeit prüfen.~~
~~**[<=]**~~

~~**A_19835 - Entschlüsselung des Consent**~~
~~Der Authorization-Endpunkt MUSS den verschlüsselten Consent mit seinem eigenen privaten Schlüssel "PRK_AUTH" entschlüsseln.~~**[<=]**

~~**A_19836 - Signaturprüfung des Consent**~~
~~Der Authorization-Endpunkt MUSS die Signatur des entschlüsselten "consent" gegen den PUK des Authenticators "PUK_AUTH" prüfen. [<=]~~

A_20440~~A_19837~~ - Schematische Prüfung des Consent
Der IdP-Dienst MUSS den eingereichten Consent auf dessen Übereinstimmung mit dem vorliegenden ~~Schema (Claim~~Claim (mit dem Fachdienst abgestimmte Key/Value-Paare im Payload des Token) zum beantragten Token abgleichen, insbesondere die "redirect_uri" aus dem Registrierungszusammenhang.[<=]

A_20379 - Abbruch bei schematischer Inkonsistenz im Consent
~~Stimmen das Schema des Consent und das des vorliegenden Claims nicht überein, MUSS~~ Der Authorization-Endpunkt MUSS die Bearbeitung mit dem registrierten Fehlercode und einer für den Nutzer verständlichen Fehlermeldung abbrechen, wenn das Schema des Consent und das des vorliegenden Claims nicht übereinstimmen.[<=]~~.~~ ~~[<=~~]

A_20310~~A_19838~~ - Verarbeitung des Consent
~~Hat~~Der Authorization-Endpunkt MUSS die Ausstellung des vereinbarten "AUTHORIZATION_CODE" mit den im Claim vorliegenden Parametern veranlassen, wenn der Authorization-Endpunkt den eingereichten Consent allen vorgesehenen Prüfungen unterzogen hat und ~~sind~~ dabei keine Fehler aufgetreten sind. [<=]

A_20459 - Das Attribut AUTH_TIME muss in allen Token unverändert bleiben
~~, MUSS~~ Der Authorization-Endpunkt ~~die Ausstellung des oder der im Claim vereinbarten "ID_TOKEN" und ggf. "REFRESH_TOKEN" mit~~DARF den ~~im Claim vorliegenden Parametern veranlassen. Der~~ Zeitpunkt der letzten Authentisierung ~~MUSS~~ im ~~Parameter~~Attribut "auth_time" ~~festgehalten und DARF bis zur erneuten Authentisierung~~ NICHT verändern.~~verändert werden.~~[<=]

A_20699 - Annahme der signierten "CHALLENGE"
Der Authorization Endpunkt MUSS die "CHALLENGE", signiert mit dem Zertifikat der Smartcard des Nutzers, übertragen durch das Authenticator-Modul annehmen.[<=]

Hinweis: Der folgende Aufruf skizziert einen beispielhaften HTTP GET-Request an den Authorization Endpunkt, welcher vom Authenticator-Modul initiiert wird:

```
GET /authresponse_type=code&scope=openid%20e-
rezept&state=af0ifjsldkj&client_id=ZXJlemVwdC1hcHA&redirect_uri=https%3A%2F
%2Fapp.erezept.com%2Fauthnres&code_challenge_method=S256&code_challenge=S41
HgHxhXL1CIpfGvivWYpbO9b_QKzva-9ImuZbt0Is

HTTP/1.1
Host: idp.com
X-E-Rezept-App: 1.0
Accept: application/json
User-Agent: E-Rezept-App/1.0
```

A_20522 - Erstellen einer "SESSION_ID"
Der Authorization-Endpunkt MUSS eine neue "SESSION_ID" anlegen, sobald ein Authorization Request eingeht.[<=]

A_20523 - Zusammenstellung der Claims zum "user_consent"
Der Authorization-Endpunkt MUSS die für den vorgetragenen "SCOPE" vom einfordernden Fachdienst erwarteten Claims zur "USER_CONSENT"-Anfrage zusammenstellen.[<=]

A_20460~~A_19839~~ - Der Authorization-Endpunkt bestätigt ausschließlich Zertifikatsinformationen
~~Um sicher zu stellen, dass der Nutzer berechtigt ist, die vorgetragene Identität (Zertifikat) zu nutzen, MUSS~~ Der Authorization-Endpunkt MUSS bei der Annahme des Zertifikates durch ein Challenge-Response-Verfahren prüfen, ob der Nutzer auch die zum Zertifikat (Besitz) gehörige PIN (Wissen) kennt, um sicherzustellen, dass der Nutzer berechtigt ist, die vorgetragene Identität (Zertifikat) zu nutzen.[<=~~.[<=~~]

A_20521 - Inhalt der Challenge an das Authenticator-Modul
Der IdP-Dienst MUSS die ihm vorliegenden Session-Informationen (z.B. "SESSION_ID", "CODE_CHALLENGE", "SCOPE" und alle Informationen über Anwendungsfrontend und Authenticator-Modul) mit seinem privaten Schlüssel "PrK_AUTH" signieren und als JWT ergänzt um die "USER_CONSENT"-Anfrage per HTTP-POST an das Authenticator-Modul senden.[<=]

Hinweis: Nachfolgend wird beispielhaft ein "CHALLENGE_TOKEN" in Form eines JSON Web Token (JWT) dargestellt:

```
Challenge JWT:
challenge_headers = {
    "typ":"JOSE+JSON",
    "iat":1591714252326,
    "exp":1591714552326,
    "jti":"c3a8f9c8-aa62-11ea-ac15-6b7a3355d0f6",
    "snc":"sLlxlkskAyuzdDOwe8nZeeQVFBWgscNkRcpgHmKidFc"
}
challenge_payload = {
    "response_type":"code",
    "scope":"openid e-rezept",
    "client_id":"ZXJlemVwdC1hcHA",
"state":"af0ifjsldkj",
    "redirect_uri":"https://app.e-rezept.com/authnres",
    "code_challenge_method":"S256",
    "code_challenge":"S41HgHxhXL1CIpfGvivWYpbO9b_QKzva-9ImuZbt0Is"
}
```

Der Authorization-Endpunkt hat den "CHALLENGE_TOKEN" mit seinem privaten Schlüssel "PRK_AUTH" signiert. Der folgende Aufruf skizziert beispielhaft die Antwort des Authorization-Endpunktes, welche vom Authenticator-Modul angenommen wird. Der "CHALLENGE_TOKEN" wird dabei nur angedeutet:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{
  "challenge":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpPU0UrSlNPTiIsImlhdCI6MTU5MTcxNDI1MjMy.....
",
  "user_consent": {
    "client_name": "E-Rezept App",
```

```
    "url": "https://e-rezept.com/",
    "requested_scope": {
      "openid": "Der Zugriff auf den ID Token"
      "e-rezept": "Zugriff auf die E-Rezept Funktionalität."
    },
"show_once": true,
"amr": ["JWT-Challenge-Response"]
    // ggf. mehr Informationen, welche dem Nutzer angezeigt werden sollen,
wie die Auflistung der mit der Zustimmung weitergegebenen Daten
  }
}
```

A_20604 - Signatur der Challenge
Der IdP-Dienst MUSS die Challenge für die Authentisierung mit einem Zertifikat des Typs FD.SIG und der technischen Rolle „oid_idpd" gemäß [gemSpec_Krypt # Abschnitt 3.7] signieren. **[<=]**

~~**A_20143 - Preisgabe von Informationen durch den Userinfo-Endpunkt**~~
~~Der Userinfo-Endpunkt DARF andere als die durch Zertifikate nachgewiesenen und mit dem Fachdienst im Claim vereinbarten Informationen gemäß der folgenden Zertifikatstypen NICHT preisgeben:~~

- ~~**A_20313** C.CH.AUT [OID:1.2.276.0.76.4.70] bei eGK (elektronische Gesundheitskarte)~~
- ~~C.HP.AUT [OID:1.2.276.0.76.4.75] bei eHBA (elektronischer Heilberufsausweis)~~
  - Inhalte des Claims
- ~~C.HCI.AUT [OID:1.2.276.0.76.4.77] bei SMC-B (Secure Module Card – B)~~

~~**[<=]**~~

~~**A_19840 - Inhalte des Claims**~~Der IdP-Dienst MUSS "ID_TOKEN", "ACCESS_TOKEN~~Der IdP-Dienst MUSS~~ "ID_TOKEN" und "~~REFRESH~~SSO_TOKEN" für unterschiedliche Fachdienste gemäß den mit dem jeweiligen Fachdienst abgestimmten Claims bereitstellen. Sind Inhalte des Claims teilweise oder das gesamte Claim für einen registrierten Fachdienst nicht gesetzt, befüllt der IdP-Dienst die einzelnen Parameter der Gültigkeitsdauer ("SUBJECT_SESSION", "~~ACCESS_~~AUTHORIZATION_CODE", "~~REFRESH~~ACCESS_TOKEN", "SSO_TOKEN" und "ID_TOKEN") gemäß der spezifizierten Maximalwerte. ~~der nachfolgenden Anforderungen (A_19841, A_19842, A_19843 und A_19844).~~ **[<=]**

A_20692~~A_19841~~ - Maximale Gültigkeitsdauer einer "SUBJECT_SESSION"
~~Die Gültigkeitsdauer~~ Der Authorization Server DARF die zeitliche Gültigkeit einer"SUBJECT_SESSION" ~~DARF~~ NICHT länger als 86400 Sekunden (24 Stunden) ~~betragen~~einstellen.
Der Parameter "auth_time" beinhaltet den Zeitpunkt der letzten Authentisierung.**[<=]**

~~Es liegt in der Verantwortung und im Ermessen des Betreibers des Fachdienstes, Anforderungen zu definieren, wie lange die letzte Authentisierung nachgenutzt werden darf.~~
A_20314 - Maximale Gültigkeitsdauer des "AUTHORIZATION_CODE"
~~**A_19842 - Maximale Gültigkeitsdauer des "ACCESS_CODE"**~~Der Authorization Server DARF die zeitliche Gültigkeit des "~~ACCESS~~AUTHORIZATION_CODE" NICHT länger als 180 Sekunden (Challenge-Response) und nach dessen Übergabe an das Anwendungsfrontend nicht länger als 60 Sekunden einstellen.**[<=** ~~Der Authorization~~

~~Server DARF verspätet eingehenden "ACCESS_CODE" NICHT in "ID_TOKEN" eintauschen.~~
~~[<=]~~

A_20315 - "AUTHORIZATION_CODE" nach Gültigkeitsende nicht mehr verwenden
Der Authorization Server DARF außerhalb der Gültigkeitsdauer eingehenden "AUTHORIZATION_CODE" NICHT in "ID_TOKEN", "ACCESS_TOKEN" oder "SSO_TOKEN" eintauschen.**[<=]**

Die Gültigkeitsdauer des "~~ACCESS_~~AUTHORIZATION_CODE" wird im Claim des angesprochenen Fachdienstes definiert.

A_20462 - Maximale Gültigkeitsdauer des "ID_TOKEN"
~~**A_19843 - Maximale Gültigkeitsdauer des "REFRESH_TOKEN"**~~Der Token-Endpunkt DARF "ID_TOKEN" mit einer~~Die maximale~~ Gültigkeitsdauer von ~~"REFRESH_TOKEN" DARF NICHT länger~~mehr als ~~14400~~86400 Sekunden (~~4~~24 Stunden) NICHT ausstellen.~~betragen.~~**[<=]**

A_20463 - Maximale Gültigkeitsdauer des "ACCESS_TOKEN"
Der Token-Endpunkt DARF "ACCESS_TOKEN" mit einer Gültigkeitsdauer von mehr als 300 Sekunden (5 Minuten) NICHT ausstellen.**[<=]**

Die Gültigkeitsdauer des "~~REFRESH~~ACCESS_TOKEN" wird im Claim des angesprochenen Fachdienstes definiert.

~~**A_19844 - Maximale Gültigkeitsdauer des "ID_TOKEN"**~~

~~Die maximale Gültigkeitsdauer des "ID_TOKEN" DARF NICHT länger als 900 Sekunden (15 Minuten) betragen.~~
~~[<=]~~

~~Die Gültigkeitsdauer des "REFRESH_TOKEN" wird im Claim des angesprochenen Fachdienstes definiert.~~

~~**A_19845 - Nutzer-Informationen im Claim**~~
~~Fachdienste MÜSSEN die im Claim angeforderten Informationen über den Nutzer bei ihrer Registrierung angeben.[<=]~~

A_20464~~A_20140~~ - Token-Endpunkt (Datensparsamkeit)
Der Token-Endpunkt DARF andere Informationen, als die im Claim geforderten, ~~DARF der Token-Endpunkt~~ NICHT herausgeben.**[<=]**

~~**A_20141 - Token (Identifikation des Nutzers)**~~
~~Alle Token MÜSSEN die eineindeutige Kennung des Inhabers "bearer" z.B. Telematik-ID enthalten. [<=]~~

A_20318~~A_19977~~ - Keine Token für widerrufene Entitäten
Der Authorization-Endpunkt ~~MUSS ausschließen, "ACCESS_CODE", "ID_TOKEN" oder "REFRESH_TOKEN"~~DARF für nicht existente Entitäten NICHT

einen "AUTHORIZATION_CODE", einen "ID_TOKEN", einen "ACCESS_TOKEN" oder widerrufene Entitäten einen "SSO_TOKEN" auszustellen. [<=]

A_20465A_19978 - Zertifikatsprüfung gegen OCSP-Responder
Der Authorization-Endpunkt MUSS das Zertifikat des Antragstellers MUSS immer gegen den zugehörigen OCSP-Responder innerhalb der TI auf Gültigkeit prüfen.geprüft werden.[<=]

Anhand der eindeutigen Kennung (Telematik-ID bzw. KVNR) kann zu jeder Zeit die Quelle des Tokens auf der Nutzerseite adressiert werden.

6.2.2 Authorization-Endpunkt Ausgangsdaten

Konnten alle Prüfungen des eingereichten Consent erfolgreich abgeschlossen werden, erstellt der Authorization-Endpunkt ein "ID_TOKEN", ggf."ACCESS_TOKEN", ergänzt durch ein damit verbundenes "REFRESH"SSO_TOKEN". Die Übertragung des "ID_TOKEN" erfolgtder Tokenerfolgt jedoch nicht direkt über dendas Authenticator-Modul, sondern in Form eines "ACCESS_AUTHORIZATION_CODE". Dieser "ACCESS_CODE" wird vom Authorization-Endpunkt so ausgestellt (verschlüsselt), dass dieser nur vom Anwendungsfrontend verwendet werden kann. Das/Die Token werden am Token-Endpunkt zum Download bereitgestellt, wo das jeweilige Anwendungsfrontend diese gegen gleichzeitige Vorlage von "ACCESSauthorization_code" und des eigenen "SECRET" (code_verifier", auf welchem der bereits vorliegende Hash-Wert beruht), erhält.

A_20377 - Verwendung des Attributes "state"
Der Authroization-Endpunkt MUSS den "state"-Parameter [RFC6749 # section-10.12] des Anwendungsfrontends in allen darauf basierenden Responses verwenden.[<=]

A_20694 - Zusammenstellung des "SSO_TOKEN"
A_19846 - Signatur des "ACCESS_CODE"Der Authorization Endpunkt MUSS den "SSO_TOKEN" so zusammenstellen, dass alle Informationen, welche für die Ausstellung eines neuen "ACCESS_TOKEN" benötigt werden, im Token vorhanden sind. [<=]

A_20695 - Signieren des "SSO_TOKEN"
Der Authorization Endpunkt MUSS den "SSO_TOKEN" Server MUSS den "ACCESS_CODE" mit demseinem eigenen privaten Schlüssel signieren "PrK_AUTH".[<=]

A_20696 - Verschlüsselung des "SSO_TOKEN"
Der Authorization Endpunkt verschlüsselt den "SSO_TOKEN" mit seinem öffentlichen Schlüssel "PUK_AUTH" für sich selbst.[<=]

A_20697 - Zusammenstellung des "AUTHORIZATION_CODE"
Der Authorization Endpunkt erzeugt den "AUTHORIZATION_CODE" anhand " signieren, damit der vom Authenticator-Modul übergebenen Daten im "CHALLENGE".[<=]

A_20319 - Signatur des "AUTHORIZATION_CODE"
Der IdP-Dienst MUSS den "AUTHORIZATION_CODE" für die Authentisierung mit einem Zertifikat des Typs FD.SIG und der technischen Rolle „oid_idpd" gemäß [gemSpec_Krypt # Abschnitt 3.7] signieren damit das Authenticator-Modul sicher gewährleisten kann, dass der eingehende "ACCESS"AUTHORIZATION_CODE" tatsächlich von diesemvom IdP-Dienst stammt [[RFC7519 # section-7.1].[<=]

A_20693 - Senden des "AUTHORIZATION_CODE" und "SSO_TOKEN" an die "REDIRECT_URI"

~~**A_19847 - Verschlüsselung des "ACCESS_CODE"**~~

~~Der Authorization-Endpunkt MUSS den "ACCESS_CODE" vor dem Übertragen mit dem öffentlichen Schlüssel des Anwendungsfrontend "PUK_FRONT" verschlüsseln [RFC7519 # section-7.1 und RFC7523 # section-7].**[<=]**~~

Der IdP-Dienst MUSS den "AUTHORIZATION_CODE" und den "SSO_TOKEN" an das Authenticator-Modul über eine Redirection an die registrierte "REDIRECT_URI" des Anwendungsfrontends senden.**[<=]**

A_20320 - Sichere Übertragung des "AUTHORIZATION_CODE"

~~**A_19832 - Sichere Übertragung des "ACCESS_CODE"**~~Der Authorization-Endpunkt MUSS den Transport des ~~Authorization-Code~~"AUTHORIZATION_CODE" über unsichere Netze (z.B. Internet) durch Verwendung von Transport Layer Security (TLS) gemäß den Vorgaben der [gemSpec_Krypt] sichern ~~[~~[RFC7523 # section-7].**[<=]**

~~6.3 Redirection-Endpunkt~~

~~Der Authorization-Server muss einen Redirection-Endpunkt gemäß [RFC6749 # section-3.1.2] bereitstellen, damit das Anwendungsfrontend die Adressierung zum Authenticator über den IdP-Dienst auflösen kann. Ohne Redirection-Endpunkt ist es ein schwieriges oder gegebenenfalls unmögliches Unterfangen, die aktuelle IP-Adresse und somit URI des Authenticators zu ermitteln, da keine namensbasierte Adressierung erfolgen kann. Es muss daher einen Dienst geben, der vom Anwendungsfrontend aus leicht erreichbar ist und der zudem die Auflösung des Adressierungshindernisses beseitigt.~~

~~Der IdP-Dienst muss zu diesem Zweck eine Datenbank betreiben, anhand welcher es einem Anwendungsfrontend möglich wird, den mit ihm verknüpften Authenticator zu ermitteln. Bei der Registrierung des Authenticators wird dieser durch den Authorization Server mit einem eindeutigen Identifikationsmerkmal versehen und in der Datenbank abgespeichert. Bei der Registrierung eines Anwendungsfrontend wird der Nutzer aufgefordert, die eindeutige Verknüpfung zwischen dem Anwendungsfrontend und dem Authenticator zu bestätigen. Der Authorization Server gibt bei der Registrierung eines Anwendungsfrontend eine URL oder einen QR-Code bekannt, welche mit dem betroffenen Authenticator aufzurufen ist. Ruft der Authenticator diese URL durch Eingabe von Hand, durch das Nutzen einer entsprechenden Verlinkung oder das Einscannen des QR-Code auf, wird vom IdP-Dienst die Rückfrage gestellt, ob die Verknüpfung zwischen Authenticator und diesem Anwendungsfrontend erfolgen soll. Ist diese Rückfrage bestätigt, speichert der IdP-Dienst diese n:m Beziehung in der Datenbank ab und setzt sie mit einem Zeitstempel versehen auf aktiv. Der Zeitstempel dient hier der Forensik, woraus sich ergibt, dass der Eintrag nicht mehr gelöscht wird. Im Falle einer späteren Löschung findet die Deaktivierung des Eintrags mit erneutem Zeitstempel statt. Auf diese Weise kann eine einmal etablierte Beziehung, auch lange nachdem diese wieder entfernt wurde, erkannt werden.~~

~~Ruft nun das Anwendungsfrontend den Redirection-Endpunkt auf, wobei es seinen eigenen Identifier überträgt, kann der Authorization Server den oder die für dieses Anwendungsfrontend registrierten Authenticator ermitteln und den HTTP/1.1 GET oder POST Request an den Authenticator weiterleiten.~~

Hinweis:

Es ist erforderlich, dass der Authenticator mittels HTTP/1.1 POST-Request erreichbar ist und die Anrufe nicht an einer Firewall oder einem Application Layer Gateway (ALG) geblockt werden. Ist dies der Fall, muss der Authenticator erst eine Verbindung zum Authorization Server etablieren, damit diese Sicherheitsfunktion kurzzeitig abgestellt wird. Der IdP-Dienst muss in diesem Fall auch den von der Firewall oder dem ALG übermittelten Port in die Redirection mit einbeziehen.

A_20106 - Bereitstellung Redirection-Endpunkt
Der Authorization Server MUSS einen Redirection-Endpunkt gemäß [RFC6749 # section-3.1.2] bereitstellen.**[<=]**

A_20110 - Absicherung Redirection-Endpunkt
Der Redirection-Endpunkt MUSS HTTP/1.1 GET oder POST Requests ohne Fehlermeldung ablehnen, wenn der im HTTP/1.1-Header angegebene Identifier in der Datenbank nicht vorkommt. Anwendungsfrontend und Authenticator MÜSSEN vor der Nutzung des Redirection-Endpunkt beim Authorization Server registriert sein.**[<=]**

6.3.1 Eingangsdaten Redirection-Endpunkt

A_20107 - Redirection-Endpunkt Auswertung HTTP-Header
Der Redirection-Endpunkt MUSS anhand der Header-Informationen (HTTP/1.1 GET oder POST Request) die ID des Token des aufrufenden Anwendungsfrontend und damit in der Datenbank den dazugehörigen Authenticator ermitteln, an welchen die Anfrage als Redirection weiterzuleiten ist.**[<=]**

6.3.2 Ausgangsdaten Redirection-Endpunkt

A_20108 - Redirection-Endpunkt Ergänzung mit Portangaben
Der Redirection-Endpunkt MUSS die Weiterleitung eines HTTP-Request an einen Authenticator durch dessen priviligierten Port ergänzen.**[<=]**

Hinweis: Nachfolgend wird beispielhaft ein "AUTHORIZATION_CODE" in Form eines JSON Web Token (JWT) dargestellt:

```
Authorization Code:
code_header = {
    "typ": "JOSE",
    "jti": "18017c1c-aa7b-11ea-ac15-6b7a3355d0f6",
    "iat": 1591714352326,
    "exp": 1591714652326,
    "msg_type": "code"
}
code_payload = {
    "response_type": "code",
    "scope": "openid e-ezept",
    "client_id": client_id,
"state":"af0ifjsldkj",
    "redirect_uri": "https://app.e-rezept.com/authnres",
    "code_challenge_method": "S256",
    "code_challenge": "S41HgHxhXL1CIpfGvivWYpbO9b_QKzva-9ImuZbt0Is",
    "claims": {
      "sub": "RabcUSuuWKKZEEHmrcNm_kUDOW13uaGU5Zk8OoBwiNk",
  // die ausgelesenen Werte aus dem Smart Card Zertifikat
```

```
      "professionOID": "<die Profession OID>",
  "idNummer":"<KV-Nummer>",
  "name": "<Name des Versicherten>"
    }
}
```

Hinweis: Nachfolgend wird beispielhaft ein "SSO_TOKEN" in Form eines JSON Web Token (JWT) dargestellt (Der IdP entscheidet selbst über den Inhalt , da nur er diesen auch lesen kann. Entscheidend ist, dass die Informationen zur Ausstellung eines neuen "AUTHORIZATION_CODE" - die Nutzer Informationen aus dem Smartcard-Zertifikat des Nutzers - enthalten sind):

```
{ // protected header
      "alg": "dir",
      "enc": "A256GCM",
      "kid": <key identifier of the ecryption key>,
      "jti": "c3a8f9c8-aa62-11ea-ac15-6b7a3355d0f6",
      "iat": <issuance time>,
      "exp": <expiration time of the token>,
      "typ": "JOSE",
      "msg_typ": "SSO-Token"
    }
    .
    { // body
      "iss": "https://idp.com/oidc",
      "aud": "https://idp.com/oidc",
      "cnf": {
        "jwk": {
          "kty": "EC",
          "use": "sig",
      "x": "l8tFrhx-34tV3hRICRDY9zCkDlpBhF42UQUfWVAWBFs",
      "y": "9VE4jf_Ok_o64zbTTlcuNJajHmt6v9TDVrU0CdvGRDA",
      "crv": "P-256"
        }
      },
      "claims": {
"professionOID": "<die Profession OID>",
"idNummer":"<KV-Nummer>",
"name": "<Name des Versicherten>"
      }
    }
```

Hinweis: Nachfolgend wird beispielhaft die Antwort des Authorization Endpunkt als Redirection dargestellt. Der "AUTHORIZATION_CODE" und der "SSO_TOKEN" werden nur angedeutet":

```
HTTP/1.1 302 Found
Location: https://app.e-rezept.com/authnres?code=eyJhbGciOiJkaXIiL
CJlbmMiOiJBMjU2R0NNIiwiZXhwIjoxNT....&ssotoken=eyJhbGciOiJkaXIiLCJlbmMiOiJB
MjU2R0NNIiwiZXhwIjoxNTkxNzE0NjU....&state=af0ifjsldkj
```

~~6.4~~6.3 Token-Endpunkt

Am Token-Endpunkt nimmt der Authorization Server ~~einerseits "ACCESS~~**den** "AUTHORIZATION_CODE~~" und andererseits "REFRESH_TOKEN", welche~~", welchen er selbst am Authorization~~-Endpunkt oder am Token~~-Endpunkt ausgegeben hat, entgegen. Da beide vom Authorization Server selbst erstellt wurden, ist deren Prüfung auf Integrität keine besondere Herausforderung. Allerdings muss der Token-Endpunkt beim Einreichen eines "~~ACCESS~~AUTHORIZATION_CODE" das dabei mit übertragene "~~SECRET~~CODE_VERIFIER" verarbeiten, um mittels Vergleich der Hash-Werte die Übereinstimmung des ~~Access_Code~~"AUTHORIZATION_CODE" einreichenden mit dem ursprünglich authentisierten Client sicherzustellen. Das verwendete Hash-Verfahren ist im Authorization Request anzugeben.

~~6.4.1~~6.3.1 Token-Endpunkt Eingangsdaten

A_20321 - Annahme und Prüfung von "AUTHORIZATION_CODE" und "CODE_VERIFIER"
~~A_19825 - Annahme und Prüfung von "ACCESS_CODE" und "SECRET"~~Der Token-Endpunkt MUSS den vom Anwendungsfrontend übertragenen "~~ACCESS~~AUTHORIZATION_CODE" nach Überprüfung des zeitgleich eingereichten "~~SECRET~~CODE_VERIFIER" entwerten und das ~~mit dem ursprünglichen "CONSENT" ausgestattete~~ "ID_TOKEN" und "ACCESS_TOKEN" gesichert herausgeben. Der Token-Endpunkt MUSS die Überprüfung des "CODER_VERIFIER" gegen die "CODE_CHALLENGE" mit S256 (Algorithmus nach [RFC7636 # section-4.2]) durchführen.**[<=~~[<=~~]**

A_20474 - "AUTHORIZATION_CODE" einmalige Verwendung
~~A_19826 - "ACCESS_CODE" einmalige Verwendung~~Der Authorization Server MUSS ~~beide bestehenden "SUBJECT_SESSION" terminieren, wenn ein "ACCESS_CODE"~~sicherstellen, dass auf einen wiederholt ~~eingereicht wird. [<=]~~

~~Hinweis: Da es nicht sicher ist, ob in einem solchen Fall der erste oder zweite Vortragende den "ACCESS_CODE" rechtmäßig verwendet, müssen beide mit dem "ACCESS_CODE" in Verbindung stehenden "SUBJECT_SESSION" eliminiert~~eingereichten "AUTHORIZATION_CODE" keine weiteren Token herausgegeben werden.[<=]

A_20323 - TOKEN-Ausgabe Protokollierung in allen Fällen
~~A_19827 - "ID_TOKEN" Protokollierung in allen Fällen~~Der Token-Endpunkt MUSS die Herausgabe ~~eines "ID_~~der "TOKEN" im Positiv- wie auch im Negativfall protokollieren.~~[<=]~~

~~**A_19828 - Annahme und Prüfung des "REFRESH_TOKEN"**~~
~~Der Token-Endpunkt MUSS die von ihm selbst ausgegebenen "REFRESH_TOKEN" annehmen und mit seinem "PRK_TOKEN" entschlüsseln. Treten bei der Entschlüsselung Bitfehler auf, MÜSSEN "REFRESH_TOKEN" und "SUBJECT_SESSION" sofort terminiert werden.[<=]~~

~~Die erfolgreiche Entschlüsselung ist ein Garant dafür, dass das "REFRESH_TOKEN" seit dessen Herausgabe unverändert ist.~~

A_19829 - "REFRESH_TOKEN" Protokollierung nur im Negativfall [<=] Der Token-Endpunkt MUSS nur im Negativfall, also im Falle der Ablehnung des "REFRESH_TOKEN", dessen Ablehnung protokollieren.
[<=]

6.4.26.3.2 Token-Endpunkt Ausgangsdaten

Da perspektivisch eine Vielzahl unterschiedlicher Fachdienste mit den vom Authorization Server bereitgestellten "ID_TOKEN" erreichbar gemacht werden, müssen alle Schnittstellen und Protokolle möglichst sicher umgesetzt werden. Hierzu gehört, dass eine mögliche „KANN"- oder „SOLL" Signatur gemäß HEART-Erweiterung [openid-heart-1_0 Stand 2017-05-31 und openid-heart-oauth2] verpflichtend wird. Ebenso müssen alle Token und alle mit der Beantragung der Token in Verbindung zu bringenden Informationsaustausche zusätzlich zum TLS mit den öffentlichen Schlüsseln der Empfänger verschlüsselt werden.

Anwendungen des Nutzers und Authenticator reichen hierzu bei der dynamischen Registrierung ihre öffentlichen Schlüssel "PUK_APP" und "PUK_FRONT" ein, wodurch diese für alle folgenden Prozessschritte von Beginn an allen Beteiligten zur Signaturprüfung und Verschlüsselung bereitgestellt werden können.

Alle vom Dienstanbieter verbreitetenIdP-Dienst herausgegebenen Informationen müssen vor dem Verschlüsseln (mit dem PUK des Empfängers) mit dem privateKey des Dienstesjeweiligen Teildienstes signiert sein, da die mit einer TLS-Sicherung verbundene Herausgeberidentifizierung TLS abgesicherte Verbindung nicht in allen Anwendungsszenarien gegeben ist.

Der Empfänger der Daten muss also auch ohne TLS in der Lage sein, die Herkunft unddie Integrität der Daten prüfen zu können. Daher werden Verschlüsselung und Signatur immer als MUSS-Anforderung spezifiziert.übertragenen Daten gewährleistet.

Die Absicherung des Transportweges erfolgt ausschließlich serverseitig, da gespeichertem Schlüsselmaterial im Endgerät des Nutzers, ebenso wie dem Schlüsselmaterial des Anwendungsfrontend nicht vertraut werden kann.

A_19820 - Erfolgreiche Antwort auf "REFRESH_TOKEN"
Erfüllt das Token alle Voraussetzungen gemäß [openid-connect-core], beinhaltet es alle im Claim geforderten Attribute und konnte es erfolgreich geprüft werden, MUSS der Token-Endpunkt dem Anwendungsfrontend das "ID_TOKEN" und ggf. ein "REFRESH_TOKEN" ausstellen.**[<=]**

A_20524 - Befüllen der Claims "given_name", "family_name", "organizationName", "professionOID" und "idNummer"
Der Token-Endpunkt MUSS benötigte Attribute in Claims für das auszustellende "ACCESS_TOKEN" und das "ID_TOKEN" ausschließlich aus dem ihm mit der "challenge" eingereichten Authentifizierungs-Zertifikat der Smartcard (eGK, HBA oder SMC-B) beziehen.
Der Token-Endpunkt MUSS das Attribut "given_name" und "family_name" der juristischen und natürlichen Personen sowie die Attribute "organizationName", "professionOID" und "idNummer" entsprechend des Datenformates der

Informationsquelle (Zertifikat) wie folgt befüllen:

Tabelle 5: TAB_IDP_DIENST_0005 Befüllung der Attribute "given_name", "family_name", "organizationName", "professionOID" und "idNummer"

Attribute	Leistungserbringer (HBA) Quell-Zertifikat: C.HCI.AUT	Leistungserbringerinstitution (SMC-B) Quell-Zertifikat: C.HP.AUT	Versicherte (eGK) Quell-Zertifikat: C.CH.AUT
Attribute "given_name" (Zertifikatsfeld)	Vorname (surname)	Vorname des Verantwortlichen/ Inhabers (surname)	Vorname (surname)
Attribute "family_name" (Zertifikatsfeld)	Nachname (givenName)	Nachname des Verantwortlichen/ Inhabers (givenName)	Nachname (givenName)
Attribute "organizationName" (Zertifikatsfeld)	leer (organizationName)	Organisationsbezeichnung (organizationName)	Herausgeber (organizationName)
Attribute "professionOID" (Zertifikatsfeld)	professionOID (Admission/professionOID)	professionOID (Admission/professionOID)	professionOID (Admission/professionOID)
Identifier "idNummer" (Zertifikatsfeld)	Telematik-ID (Admission/ registrationNumber)	Telematik-ID (Admission/ registrationNumber)	unveränderlicher Anteil der KVNR (organizationalUnitName)

[<=]

Hinweis: Nachfolgend wird ein beispielhafter Payload eines "ACCESS_TOKEN" dargestellt.

```
{
    "iss": "https://idp1.telematik.de/jwt",
    "sub": "RabcUSuuWKKZEEHmrcNm_kUDOW13uaGU5Zk8OoBwiNk",
    "professionOID": "1.2.276.0.76.4.50",
    "nbf": 1585336956,
    "exp": 1585337256,
    "iat": 1585336956,
    "given_name": "der Vorname",
    "family_name": "der Nachname",
    "organizationName": "Institutions- oder Organisations-Bezeichnung",
    "idNummer": "3-15.1.1.123456789",
    "jti": "<IDP>_01234567890123456789",
    "aud": "https://erp.telematik.de/login"
```

}

A_20327 - Signatur des "ID_TOKEN", "ACCESS_TOKEN" und "SSO_TOKEN"
~~**A_19821 - Signatur des "ID_TOKEN" und "REFRESH_TOKEN"**~~Der Token-Endpunkt MUSS alle erstellten "ID_TOKEN", "ACCESS_TOKEN" und "~~REFRESH~~SSO_TOKEN" mit seinem privateKey "PRK_TOKEN" gemäß ~~[gemSpec_Krypt#6]~~ signieren, um dessen Integrität sicherzustellen und eine eineindeutige Erklärung über dessen Herkunft abzugeben. ~~[RFC7523#Abschnitt 3~~[RFC7523 # section-3 Spiegelpunkt 9 und openid-heart-oauth2-1_0.html#rfc.section.3.2.1] ~~ist~~sind zu gewährleisten.**[<=]**

A_20328 - Verschlüsselung des "ACCESS_TOKEN"
~~**A_19822 - Verschlüsselung des "ID_TOKEN"**~~Der Token-Endpunkt MUSS das "~~ID~~ACCESS_TOKEN" mit dem öffentlichen Schlüssel des Fachdienstes "PUK_FD" verschlüsseln, ~~für welchen dieses bestimmt ist,~~ um das "~~ID~~ACCESS_TOKEN" vor Kenntnisnahme durch Dritte, z.B. auch auf dem Endgerät des Nutzers, zu schützen ~~[~~[RFC6750 # section-5.2].
[<=] ~~und openid-heart-oauth2-1_0.html#rfc.section.3.2.1]. [<=]~~

A_20329 - Sichere Übertragung von "ID_TOKEN", "ACCESS_TOKEN" und "SSO_TOKEN"
~~**A_19816 - Verschlüsselung des "REFRESH_TOKEN"**~~Der Token-Endpunkt MUSS ~~das "REFRESH_TOKEN" mit seinem eigenen öffentlichen Schlüssel "PUK_TOKEN" verschlüsseln [RFC6750 # section-5.2 und openid-heart-oauth2-1_0.html#rfc.section.3.2.1]. [<=]~~

~~**A_19817 - Signatur des "REFRESH_TOKEN"**~~ "ID_TOKEN", "ACCESS~~Der Token-Endpunkt MUSS das "REFRESH_TOKEN" mit seinem privaten Schlüssel "PRK_TOKEN" signieren [RFC6750 # section-5.2 und openid-heart-oauth2-1_0.html#rfc.section.3.2.1]. [<=]~~

~~**A_19818 - "REFRESH_TOKEN" mathematische und zeitliche Gültigkeit**~~
~~Der Token-Endpunkt MUSS die Signatur entgegengenommener "REFRESH_TOKEN" auf zeitliche Gültigkeit und mathematische Korrektheit überprüfen.[<=]~~

~~**A_19819 - "REFRESH_TOKEN" Integritätsprüfung**~~
~~Der Token-Endpunkt MUSS anhand der Signatur die Integrität des "REFRESH_TOKEN" sicherstellen.[<=]~~

~~**A_19809 - Sichere Übertragung von "ID_TOKEN" und "REFRESH_TOKEN"**~~und "SSO~~Der Token-Endpunkt MUSS "ID_TOKEN" und "REFRESH~~_TOKEN" beim Transport mit Transport Layer Security (TLS) gemäß [~~BCP195] und [~~gemSpec_Krypt] schützen.
[<=]

A_20330 - Ausgabe der Token
~~**A_19811 - Adressierung der Token beim Versand**~~Der Token-Endpunkt MUSS für den Versand der "ID_TOKEN" und "~~REFRESH~~ACCESS_TOKEN" an das Anwendungsfrontend, die vom Authenticator-Modul im Consent der mit dem Vorgang verbundenen "SUBJECT_SESSION" gemeldete URI verwenden. Eine URI-Umleitung MUSS ausgeschlossen werden [RFC6749 # section-10.6].**[<=]** ~~. [<=]~~

Hinweis: Nachfolgend wird beispielhaft ein "ID_TOKEN" und ein "ACCESS_TOKEN" in Form eines JSON Web Token (JWT) dargestellt:

ID Token:

```
idt_headers = {
    # "typ":"JOSE+JSON"
}
idt_payload = {
 "iss": "https://idp.com/oidc",
 "sub": "RabcUSuuWKKZEEHmrcNm_kUDOW13uaGU5Zk8OoBwiNk",
 "aud": [client_id],
 "iat": 1591714452326,
 "exp": 1591714752326,
 "at_hash": at_hash
}
```

Hinweis: Der ID Token wird vom Token Endpunkt signiert (mit "PrK_TOKEN")

Access Token:

```
at_headers = {
    "typ":"at+JWT"
}
at_payload = {
    "iss": "https://idp.com/oidc",
    "sub": "RabcUSuuWKKZEEHmrcNm_kUDOW13uaGU5Zk8OoBwiNk",
    "aud": "https://rs.e-rezept.com/",
    "client_id": client_id,
    "scope": "openid e-rezept",
    "iat": 1591714452326,
    "exp": 1591714752326,
    "jti": "d8557394-ab37-11ea-ac15-6b7a3355d0f6",
    "professionOID":"<die Profession OID>",
"idNummer":"<KV-Nummer>",
"name":"<Name des Versicherten>"
}
```

Hinweis: Der Access Token wird vom Token Endpunkt signiert (mit "PrK_TOKEN") und für den Fachdienst verschlüsselt (mit "PUK_FD"). Nachfolgend wird beispielhaft die Antwort des Token Endpunkt als dargestellt. Der "ID_TOKEN" und der "ACCESS_TOKEN" werden nur angedeutet.

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{"token_type": "Bearer",
 "expires_in": 300,
 "id_token": "...",
 "access_token": "...",
}
```

6.5 Token Introspection-Endpunkt

Der Token Introspection-Endpunkt bietet den Fachdiensten die Möglichkeit, vom Token-Endpunkt herausgegebene "ID_TOKEN" auf deren Bindung zu einer bestimmten Entität zu überprüfen. Der Token Introspection-Endpunkt liefert an dieser Schnittstelle gegen Vorlage des "ID_TOKEN" die von ihm mit dem Token im Consent bestätigten Daten des verwendeten Claim. Hier besteht für einen Fachdienst die Möglichkeit, für das ihm vorgelegte Token dessen aktuellen Gültigkeitsstatus zu erfragen. Auf diese Weise ist es möglich, Token auch während ihrer eigentlich noch andauernden Gültigkeit zu widerrufen.

Die Token Introspection soll von Fachdiensten mindestens einmal während der Gültigkeitsdauer des "ID_TOKEN" erfolgen.

A_19812 - Token Introspection Timeout
Der Token Introspection-Endpunkt MUSS auch unter maximaler Belastung eine Token Introspection ordnungsgemäß beantworten, sodass das Timeout von 3 Sekunden beim Fachdienst nicht erreicht wird.**[<=]**

6.5.1 Token Introspection-Endpunkt Eingangsdaten

A_19806 - Prüfung von "ID_TOKEN" am Introspection-Endpunkt
Um dem Fachdienst die Gültigkeitsprüfung des "ID_TOKEN" zu ermöglichen, MUSS der Introspection-Endpunkt gegen Vorlage des "ID_TOKEN" Auskunft über die mit dem "ID_TOKEN" bestätigten Meta-Informationen erteilen. Das zu prüfende "ID_TOKEN" MUSS mit dem Parameter "token_type_hint" mit dem String "id_token" gekennzeichnet sein [RFC7662 # section-2.1]. **[<=]**

A_19807 - Nur Fachdienste führen Token Introspection durch
Der Token Introspection-Endpunkt DARF Anfragen zur Introspection von "ID_TOKEN" NICHT von anderen als Fachdiensten "URI_FD" annehmen [rfc7662#section-4]. **[<=]**

Hinweis: Zur Token Introspection berechtigt ist ausschließlich der Fachdienst als Empfänger des "ID_TOKEN". Eine Introspection durch das Anwendungsfrontend ist nicht möglich, da das "ID_TOKEN" mit dem öffentlichen Schlüssel des Fachdienstes "PUK_FD" verschlüsselt ist und somit auch nur der adressierte Fachdienst dieses entschlüsseln kann.

Hinweis: Prüfung von "REFRESH_TOKEN" am Introspection-Endpunkt [rfc7662#section-5]

Eine Prüfung der Gültigkeit durch das Anwendungsfrontend ist nicht möglich, da das "REFRESH_TOKEN" nur gegen ein "ID_TOKEN" eingetauscht werden kann. Das "REFRESH_TOKEN" ist durch den String "refresh_token" im Parameter "token_type" gekennzeichnet und kann nur vom Token-Endpunkt entschlüsselt werden, da es mit dessen "PUK_TOKEN" verschlüsselt wurde.

6.5.2 Token Introspection-Endpunkt Ausgangsdaten

Der Token Introspection-Endpunkt prüft nach dem Einreichen einer Token Introspection-Anfrage seine Zuständigkeit. Er erkennt seine Zuständigkeit zum einen daran, dass die

ihm vorgetragene Introspection-Anfrage mit seinem eigenen öffentlichen Schlüssel "PUK_INTRO" verschlüsselt ist, und zum anderen daran, dass die Token-ID (das vorgetragene Token) sich in der Datenbank des Authorization Servers befindet.

A_19808 - Inhalte der Token Introspection Antwort
Nach einer erfolgreichen Prüfung eines "ID_TOKEN" MUSS der Token Introspection-Endpunkt dem anfragenden Fachdienst eine Token Introspection-Antwort geben [rfc7662#section-2.2]. **[<=]**

A_19798 - Speichern der Token Introspection Antwort (Token Introspection Response Caching)
Fachdienste SOLLEN die Antwort des Token Introspection-Endpunktes zwischenspeichern [rfc7662#section-2.2]. **[<=]**

A_19975 - Haltbarkeit der Token Introspection Antwort
Die Zwischenspeicherung DARF NICHT länger als die halbe Gültigkeitsdauer des "ID_TOKEN" betragen [rfc7662#section-2.2]. **[<=]**

Beispielberechnung:
```
Math.floor("iat"+("exp"-"iat")/2).
```

A_19799 - Die Token Introspection Antwort ist signiert
Das Ergebnis der Token Introspection MUSS vom Token Introspection-Endpunkt mit dessen "PrK_INT" signiert werden, um die Integrität der Daten sicher zu stellen und deren Herkunft glaubhaft zu belegen [JWT introspection response # section-5].
[<=]

A_19800 - Die Token Introspection Antwort ist verschlüsselt
Der Token Introspection-Endpunkt MUSS die Introspection-Antwort vor dem Versand an den jeweiligen Fachdienst mit dessen öffentlichen Schlüssel "PUK_FD", wie im Discovery Document angeboten, verschlüsseln [JWT introspection response # section-5]. **[<=]**

A_19796 - Verwendung von Transport Layer Security (TLS) bei Token Introspection
Der Token Introspection-Endpunkt MUSS zusätzlich zu den anderen Schutzmaßnahmen (JWT-Verschlüsselung und JWT-Signatur), Transport Layer Security (TLS) gemäß [BCP195] und [gemSpec_Krypt] nutzen [JWT introspection response # section-8.2].
[<=]

A_20138 - Reaktion auf Fehler bei der Introspection Anfrage
Der Introspection-Endpunkt MUSS fehlerhafte Anfragen oder solche, bei denen er nicht zuständig ist, gemäß [RFC7662 # section-2.3] mit einem HTTP/1.1 Errorcode 401 "invalid_token" beantworten.**[<=]**

6.6 Token Revocation Endpunkt

"ID_TOKEN" und "REFRESH_TOKEN" sowie die zugrundeliegende "SUBJECT_SESSION" haben eingeschränkte Lebenszyklen von wenigen Minuten bis zu mehreren Stunden. Da es möglich ist, dass Token auf dem Transportweg gestohlen oder unberechtigt kopiert werden, muss es Möglichkeiten geben, die Gültigkeit vor dem natürlichen zeitlichen Ableben zu beenden. So muss es beispielsweise beim Verlust des mobilen Endgerätes die Möglichkeit geben, die mit dem Gerät etablierte "SUBJECT_SESSION" (Sicherheitsbeziehung) zu terminieren. Der IdP-Dienst stellt hierfür einen Token Revocation Endpunkt bereit, welcher gemäß [RFC7009] reagiert.

6.6.1 Token Revocation Endpunkt Eingangsdaten

Der Token Revocation Endpunkt stellt einen sehr wichtigen Dienst bereit, mit dessen Hilfe es möglich ist, bereits ausgestellte noch aktive "ID_TOKEN" und ggf. "REFRESH_TOKEN" zu widerrufen. Wenngleich diese Schnittstelle äußerst selten verwendet wird, ist es umso wichtiger, dass sie in diesen Situationen einwandfrei reagiert und die erwarteten Maßnahmen standardkonform umsetzt.

Der Token Revocation Endpunkt wird daher strikt nach den Vorgaben des aktuellen Standards (derzeit [RFC7009]) umgesetzt. Die Token Revocation ist frei übersetzt als Widerrufs-Anfrage zu verstehen, da der Widerruf nicht ungeprüft durchgeführt wird.

A_19793 - Gestaltung des Antrags auf Token-Widerruf (Token Revocation)
Der Fachdienst oder das Anwendungsfrontend, welches berechtigt ist, den Widerruf eines Token zu beantragen, MUSS diese Anfrage gemäß [RFC7009 # section-2.1] umsetzen.**[<=]**

A_19794 - Mindestangaben für den Token-Widerruf (Token Revocation minimal information)
Das Anwendungsfrontend MUSS das Token im Parameter "token" und den Hinweis auf den Token-Typ "token_type_hint" mit dem Wert ("SUBJECT_SESSION", "ID_TOKEN" oder "REFRESH_TOKEN") beim Token Widerruf angeben.[RFC7009 # section-4.1.2] **[<=]**

A_20131 - Nur vollständige Token-Widerrufsanfragen werden bearbeitet
Der Token Revocation Endpunkt DARF einen Token-Widerruf NICHT bearbeiten, wenn die Anfrage unvollständig ist.**[<=]**

A_19795 - Token-Widerrufsanfragen sind zu signieren
Der Token Revocation Endpunkt DARF NICHT auf unsignierte Widerrufsanfragen durch Widerruf des Tokens reagieren. [RFC7009 # section-5]
[<=]

Der Token Revocation Endpunkt muss bei eingehenden Widerrufsanfragen die Herkunft und Integrität der Anfrage überprüfen können, weswegen die Anfrage mit dem privaten Schlüssel "PRK_FRONT" zu signieren ist.

A_19792 - Widerruf des "REFRESH_TOKEN" ("refresh_token" Revocation)
Der Widerruf des "REFRESH_TOKEN" "REFRESH_TOKEN" bezieht sich ausschließlich auf das "REFRESH_TOKEN" und führt nicht zum Widerruf des auf Basis des "REFRESH_TOKEN"

ausgestellten "ID_TOKEN" [RFC7009 # section-2].
Der Revocation-Endpunkt MUSS sicherstellen, dass nur das "REFRESH_TOKEN" widerrufen wird, wenn der Widerrufsantrag sich auf das "REFRESH_TOKEN" bezieht [RFC7009 # section-2]. [<=]

A_19791 - Widerruf des "ID_TOKEN" ("id_token" Revocation)
Der Revocation-Endpunkt MUSS sicherstellen, dass nur das "ID_TOKEN" widerrufen wird, wenn der Widerrufsantrag sich auf das "ID_TOKEN" bezogen hat. [RFC7009 # section-2][<=]

Der Widerruf des "ID_TOKEN" bezieht sich ausschließlich auf das "ID_TOKEN" und führt nicht zum Widerruf des möglicherweise zugrundeliegenden "REFRESH_TOKEN".

A_19788 - Widerruf der "SUBJECT_SESSION" durch Authenticator
Der Authenticator MUSS den Widerruf der "SUBJECT_SESSION" (Back-Channel Session Revocation) durch willentliches "Logoff" des Nutzers oder durch Deinstallation des Authenticators gemäß [RFC8417] beim Revocation-Endpunkt einreichen.[<=]

A_20018 - Widerruf der "SUBJECT_SESSION" durch Fachdienste
Wird von einem Fachdienst beim Revocation-Endpunkt ein Antrag auf Widerruf der "SUBJECT_SESSION" (Backchannel Session Revocation) eines Anwendungsfrontends eingereicht, MUSS der Revocation-Endpunkt den Widerruf aller Token, welche auf dieser "SUBJECT_SESSION" basieren, durchführen. Zudem MUSS der Revocation-Endpunkt den mit der "SUBJECT_SESSION" verknüpften Authenticator in seiner Datenbasis als gelöscht markieren. [RFC8417][<=]

A_19789 - Widerruf der "SUBJECT_SESSION" durch Backchannel Revocation
Der Revocation-Endpunkt MUSS alle aktiven "ID_TOKEN" und "REFRESH_TOKEN" mit sofortiger Wirkung widerrufen und dem Token Introspection-Endpunkt die notwendigen Informationen zukommen lassen, wenn eine Backchannel-Revocation beantragt wird, damit widerrufene Token zukünftig als bereits widerrufen identifiziert werden. [RFC8417][<=]

Hinweis: Die Backchannel-Revocation kann aktiv durch den Nutzer selbst am Authenticator erfolgen. Zusätzlich ist eine Backchannel-Revocation durch den Fachdienst möglich, wobei der Fachdienst das "ID_TOKEN" mit dem Auftrag zur Revocation, mit seinem privaten Schlüssel "PRK_FD" signiert, am Revocation Endpunkt einreicht.

A_19790 - Backchannel Revocation Information an Authorization-Endpunkt
Der Revocation-Endpunkt MUSS die "SUBJECT_SESSION" zwischen Authenticator und Authorization-Endpunkt terminieren, sodass ohne erneute Authentifizierung durch das Endgerät des Nutzers kein weiteres Token beantragt werden kann. [RFC8417]
[<=]

A_19786 - Verwendung widerrufener Token

7 Das Anwendungsfrontend, welches ein "ID_TOKEN" oder Refresh-Token widerrufen hat, DARF dieses Token NICHT erneut verwenden und MUSS dessen lokale Löschung sicherstellen.
[<=]

A_19787 - Verwendung terminierter "SUBJECT_SESSION"
Hat ein Authenticator die "SUBJECT_SESSION" widerrufen, MUSS der Authorization-Endpunkt die weitere Beantragung von "ID_TOKEN" oder "REFRESH_TOKEN" basierend auf dieser widerrufenen "SUBJECT_SESSION" unterbinden.**[<=]**

7.1.1 Token Revocation Endpunkt Ausgangsdaten

Der Token Revocation Endpunkt beantwortet die Widerrufsanträge gewissermaßen nicht. Das Ergebnis des Widerrufsantrags stellt sich in der Form dar, dass ein erfolgreicher Widerruf mit dem HTTP-Status-Code 200 beantwortet wird. Ebenso werden fehlerhafte Widerrufsanträge oder Anträge, welche sich auf nicht existente Token bzw. Sessions beziehen, mit dem HTTP-Status-Code 200 bedient.

Abweichend hiervon kann der Token Revocation Endpunkt eine Fehlermeldung liefern, wenn das zu sperrende Token eine Sperrung nicht vorsieht oder die Berechtigung zum Widerruf nicht vorliegt. In solchen Fällen reagiert der Token Revocation Endpunkt mit dem HTTP-Status-Code 503, woraus der Authenticator oder das Anwendungsfrontend schließen kann, dass das Token bzw. die "SUBJECT_SESSION" noch existent sind.

A_19784 - Rückmeldung des Status-Code der erfolgreichen Widerrufsumsetzung [RFC7009#section-2.2]
Der Token Revocation Endpunkt MUSS den vorgebrachten Sperrantrag mit HTTP-Status-Code 200 beantworten, wenn der Widerruf durchgeführt wurde oder der Widerrufsantrag fehlerhaft oder unvollständig war. **[<=]**

A_19783 - Rückmeldung des Status-Code der nicht erfolgten Widerrufsumsetzung
Der Revocation-Endpunkt MUSS eine nicht erfolgte Umsetzung eines Widerrufsantrags mit dem HTTP-Status-Code 503 beantworten.
Die Rückmeldung kann um den Hinweis "unsupported_token_type" oder im Falle einer Verzögerung mit "Retry-After" ergänzt werden. **[<=]**

7.2 Userinfo-Endpunkt

Der Userinfo-Endpunkt ist eine Basis-Schnittstelle des JSON-Web-Token-Standards und bietet Informationen über den Nutzer des Tokens an. Hier kann der angesprochene Fachdienst gegen Vorlage des "ID_TOKEN" im get-Request in Erfahrung bringen, welche Daten im Zusammenhang mit dem "ID_TOKEN" im "CONSENT" bestätigt wurden.

Mögliche Inhalte eines Standard-Claims und somit der Rahmen der im "CONSENT" zu bestätigenden Informationen sind die durch das jeweilige Identifikationsmittel bereitgestellten Informationen.

Bei der elektronischen Gesundheitskarte (eGK) gehen die Informationen, die im Consent bestätigt werden können, aus [gemSpec_PKI#5.1.3.1 C.CH.AUT und C.CH.AUT_ALT – Authentisierung eGK] hervor.

Beim elektronischen Heilberufsausweis (eHBA) ergeben sich diese Informationen aus der Spezifikation [gemSpec_PKI#5.2.1.1 C.HP.AUT – Authentisierung HBA].

Bei der Verwendung einer Secure Module Card eines Leistungserbringers (SMC-B) ergibt sich der Umfang der Informationen aus [gemSpec_PKI#5.3.4 X.509 Zertifikatsprofile der SMC-B] hier genauer der Profiltyp C.HCI.AUT (gemäßTab_PKI_238).

Andere als die aus den Zertifikaten hervorgehenden Informationen über den Nutzer kann und darf der Userinfo-Endpunkt nicht preisgeben, da deren Herkunft nicht nachweisbar ist.

A_19782 - Informationen am Userinfo-Endpunkt

Der Userinfo-Endpunkt DARF Informationen, welche nicht nachweislich aus einem durch einen zugelassenen TSP der Telematikinfrastruktur (TI) herausgegebenen Authentisierungszertifikat hervorgehen, NICHT preisgeben. Weitere Informationen, abgesehen von META-Informationen,DARF der Userinfo-Endpunkt NICHT preisgeben.**[<=]**

~~8~~ 7 Anhang A – Verzeichnisse

~~8.1~~ 7.1 Abkürzungen

Kürzel	Erläuterung
AVS	Apothekenverwaltungssystem
DLL	Dynamic Link Library
eGK	Elektronische Gesundheitskarte
HBA	Heilberufsausweis
IdP	Identity Provider
JSON	JavaScript Object Notation
JWE	JSON Web Encryption
JWS	JSON Web Signature
JWT	JSON Web Token
NFC	Near Field Communication (Kommunikation im Nahfeld einer Antenne)
OAuth 2.0	Open Authorization 2.0
OCSP	Online Certificate Status Protocol
OIDC	OpenID Connect
PIN	Personal Identification Number
PKI	Public Key Infrastructure
PVS	Praxisverwaltungssystem
QES	Qualifizierte Elektronische Signatur
SMC-B	Security Module Card Typ B, Institutionenkarte
TI	Telematikinfrastruktur
TLD	Top Level Domain

TLS	Transport Layer Security
TSL	Trust-service Status List
URI	Uniform Resource Identifier

~~8.2~~7.2 Glossar

Begriff	Erläuterung
Access Token	Ein Access Token (nach [RFC6749 # section-1.4]) wird vom Client (Anwendungsfrontend) benötigt, um auf geschützte Daten eines Resource Servers zuzugreifen. Die Representation kann als JSON Web Token erfolgen.
Authorization Server	OAuth2 Rolle (siehe [RFC6749 # section-1.1]): Der Authorization Server ist Teil des IdP-Dienstes. Der Server authentifiziert den Resource Owner (Nutzer) und stellt Access Tokens für den vom Resource Owner erlaubten Anwendungsbereich (Scope) für einen Resource Server bzw. eine auf einem Resource Server existierende Protected Resource aus.
Claim	Ein Key/Value-Paar im Payload eines JSON Web Token.
Client	OAuth2 Rolle (siehe [RFC6749 # section-1.1]): Eine Anwendung (Relying Party), die auf geschützte Ressourcen des Resource Owners zugreifen möchte, die vom Resource Server bereitgestellt werden. Der Client kann auf einem Server (Webanwendung), Desktop-PC, mobilen Gerät etc. ausgeführt werden.
Consent	Zustimmung des Nutzers zur Verarbeitung der angezeigten Daten. Der Consent umfasst die Attribute, welche vom IdP-Dienst bezogen auf die im Claim des jeweiligen Fachdienstes eingeforderten Attribute zusammenfasst. Es besteht Einigkeit zwischen dem was gefordert wird und welche Attribute im Token bestätigt werden.
Discovery Document	Ein OpenID Connect Metadatendokument (siehe [openid-connect-discovery 1.0]), das den Großteil der Informationen enthält, die für eine App zum Durchführen einer Anmeldung erforderlich sind. Hierzu gehören Informationen wie z.B. die zu verwendenden URLs und der Speicherort der öffentlichen Signaturschlüssel des Dienstes.

Funktionsmerkmal	Der Begriff beschreibt eine Funktion oder auch einzelne, eine logische Einheit bildende Teilfunktionen der TI im Rahmen der funktionalen Zerlegung des Systems.
ID Token	Ein auf JSON basiertes und nach [RFC7519] (JWT) genormtes Identitäts-Token, mit dem ein Client (Anwendungsfrontend) die Identität eines Nutzers überprüfen kann.
Open Authorization 2.0	Ein Protokoll zur Autorisierung für Web-, Desktop und Mobile Anwendungen. Dabei wird es einem Endbenutzer (Resource Owner) ermöglicht, einer Anwendung (Client) den Zugriff auf Daten oder Dienste (Resources) zu ermöglichen, die von einem Dritten (Resource Server) bereitgestellt werden.
OpenID Connect	OpenID Connect (OIDC) ist eine Authentifizierungsschicht, die auf dem Autorisierungsframework OAuth 2.0 basiert. Es ermöglicht Clients, die Identität des Nutzers anhand der Authentifizierung durch einen Autorisierungsserver zu überprüfen (siehe [openid-connect-core 1.0]).
JSON Web Token	Ein auf JSON basiertes und nach [RFC7519] (JWT) genormtes Access-Token. Das JWT ermöglicht den Austausch von verifizierbaren Claims innerhalb seines Payloads.
Resource Owner	OAuth2-Rolle (siehe [RFC6749 # section-1.1]): Eine Entität (Nutzer), die einem Dritten den Zugriff auf ihre geschützten Ressourcen gewähren kann. Diese Ressourcen werden durch den Resource Server bereitgestellt. Ist der Resource Owner eine Person, wird dieser als Nutzer bezeichnet.
Resource Server	OAuth2 Rolle (siehe [RFC6749 # section-1.1]): Der Server (Dienst), auf dem die geschützten Ressourcen (Protected Resources) liegen. Er ist in der Lage, auf Basis von Access Tokens darauf Zugriff zu gewähren. Ein solcher Token repräsentiert die delegierte Autorisierung des Resource Owners.
SSO Token	Gegen Vorlage eines gültigen SSO Token ist keine erneute Nutzerauthentisierung für die Ausstellung eines Access Tokens am IdP-Dienst nötig.
Token Endpunkt	Ein Endpunkt des Authorization Servers, welcher für die Ausstellung von Token ("ID_TOKEN" und "ACCESS_TOKEN") zuständig ist.

Das Glossar wird als eigenständiges Dokument (vgl. [gemGlossar]) zur Verfügung gestellt.

~~8.3~~7.3 Abbildungsverzeichnis

~~8.4~~7.4 Tabellenverzeichnis

~~8.5~~7.5 Referenzierte Dokumente

~~8.5.1~~7.5.1 Dokumente der gematik

Die nachfolgende Tabelle enthält die Bezeichnung der in dem vorliegenden Dokument referenzierten Dokumente der gematik zur Telematikinfrastruktur. Der mit der vorliegenden Version korrelierende Entwicklungsstand dieser Konzepte und Spezifikationen wird pro Release in einer Dokumentenlandkarte definiert; Version und Stand der referenzierten Dokumente sind daher in der nachfolgenden Tabelle nicht aufgeführt. Deren zu diesem Dokument jeweils gültige Versionsnummern sind in der aktuellen, von der gematik veröffentlichten Dokumentenlandkarte enthalten, in der die vorliegende Version aufgeführt wird.

[Quelle]	Herausgeber: Titel
[gemGlossar]	gematik: Einführung der Gesundheitskarte – Glossar
[gemILF_PS_eRp]	gematik: Spezifikation Implementierungsleitfaden Primärsysteme - E-Rezept
[gemSpec_IDP_Frontend]	gematik: Spezifikation Identity Provider-Frontend
[gemSpec_IDP_FD]	gematik: Spezifikation Identity Provider-Fachdienst

[gemSpec_Krypt]	gematik: Übergreifende Spezifikation: Verwendung kryptographischer Algorithmen in der Telematikinfrastruktur
[gemSpec_OID]	gematik: Übergreifende Spezifikation: Festlegung von OIDs
[gemSpec_PKI]	gematik: Übergreifende Spezifikation: PKI
[gemSpec_Perf]	gematik: Übergreifende Spezifikation: Performance und Mengengerüst TI-Plattform

~~8.5.2~~7.5.2 Weitere Dokumente

[Quelle]	Herausgeber (Erscheinungsdatum): Titel
~~HEART I~~[openid-connect-core]	~~openid-heart-openid-connect-1_0 [https://openid.net/specs/openid-heart-openid-connect-1_0-2017-05-31.html] (Stand: 03.10.2016)~~ OpenID Connect Core 1.0 (November 2014) https://openid.net/specs/openid-connect-core-1_0.html
~~HEART II~~	~~Health Relationship Trust Profile for OAuth 2.0 [https://openid.net/specs/openid-heart-oauth2-1_0.html] (Stand: 08.07.2018)~~
~~OpenID Connect Core~~[openid-connect-discovery]	~~OpenID Connect Core 1.0 [https://openid.net/specs/openid-connect-core-1_0.html] (Stand: 08.11.2014)~~ OpenID Connect Discovery 1.0 (November 2014) https://openid.net/specs/openid-connect-discovery-1_0.html
~~RFC3986~~[RFC6749]	~~URI~~The OAuth 2.0 Authorization Framework (Oktober 2012) https://tools.ietf.org/html/rfc6749
~~RFC7009~~[RFC6750]	~~JSON REVOCATION~~The OAuth 2.0 Authorization Framework: Bearer Token Usage (Oktober 2012) https://tools.ietf.org/html/rfc6750
~~RFC7165~~[RFC7033]	~~JOSE~~Webfinger (September 2013) https://tools.ietf.org/html/rfc7033
[RFC7231]	~~HTTP~~Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content (Juni 2014) https://tools.ietf.org/html/rfc7231
[RFC7515]	~~JWS JSON SIGNATURE~~JSON Web Signature (Mai 2015) https://tools.ietf.org/html/rfc7515

[RFC7516]	~~JWE JSON ENCRYPTION~~JSON Web Encryption (Mai 2015) https://tools.ietf.org/html/rfc7516
~~RFC7517~~	~~JWK JSON KEY~~
~~RFC7518~~	~~JWE JSON ALGORITHM~~
[RFC7519]	~~JWT JSON WEB TOKEN~~JSON Web Token (Mai 2015) https://tools.ietf.org/html/rfc7519
~~RFC7520~~	~~JOSE Protection~~
~~RFC7521~~	~~Assertion Authorization~~
~~RFC7522~~	~~Assertion SAML 2.0~~
[RFC7523]	~~JSON Token Profile~~JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants (Mai 2015) https://tools.ietf.org/html/rfc7523
~~RFC6749~~	~~Oauth2~~
~~RFC7591~~	~~Dynamic Registration~~
~~RFC6750~~	~~Oauth2 Bearer~~
[RFC7636]	~~Oauth Proof Key for Public Client~~ Proof Key for Code Exchange by OAuth Public Clients (September 2015) https://tools.ietf.org/html/rfc7636
~~RFC7662~~	~~OAuth Token Introspection~~
[RFC8252]	OAuth 2.0 for Native Apps (Oktober 2017) https://tools.ietf.org/html/rfc8252
~~RFC8417~~	~~Security Event Token~~
~~CAB-Forum~~	~~Liste vertrauenswürdiger Zertifikatsherausgeber (Root-CAs) für Anwendungen im Internet https://cabforum.org/members/~~

2436