Freie Hansestadt Bremen
DER SENATOR FÜR FINANZEN

Integrationskonzept

– Datenschutzcockpit –

Status	freigegeben
Datum	15. Mai 2024
Version	V1.1

Inhalt

Dokumenten Daten

Zuständiger Bereich	Datenschutzcockpit Leitstelle
Grundlage des Dokuments	-
Autor	Jörg Laggies (FHB extern)
Herausgeber	Kerstin Sprock (FHB)

Versionshistorie

Datum	Autor	Kommentar
27. Jun. 2023	Jörg Laggies (FHB extern)	Initiale Erstellung
29. Sep. 2023	Jörg Laggies (FHB extern),]init[	Dokumentation auf XDatenschutzcockpit 1.0.0 angepasst
15. Mai 2024	Jörg Laggies (FHB extern)	Überarbeitung und Versionierung v1.1

Abbildungsverzeichnis

1 Einleitung

Das Datenschutzcockpit (DSC) ist eine webbasierte Anwendung, die ein zusätzliches Transparenzangebot im Rahmen des Onlinezugangsgesetzes (OZG) darstellt. Die Rechtsgrundlage sowie auch die wesentlichen Rahmenbedingungen ergeben sich damit auch aus eben diesem Gesetz. Auftraggeber ist das Bundesministerium des Innern und für Heimat (BMI), derzeit per Verwaltungsvereinbarung vertreten durch das Bundesverwaltungsamt (BVA).

Das vorliegende Dokument versteht sich als Ergänzung zum Fachstandard XDatenschutzcockpit und referenziert an den relevanten Stellen auf eben diesen.

2 Token-Authentifizierung und Ende-zu-Ende-Verschlüsselung

Im Datenschutzcockpit werden gemäß § 10 Abs.2 Satz 2 OZG keine Protokoll- oder Inhaltsdaten gespeichert, sondern für die Dauer einer Nutzer-Session von den öffentlichen Stellen über eine Schnittstelle abgerufen. Die über die Schnittstelle zu übermittelnden Nachrichten und deren Inhalte werden im Fachstandard **XDatenschutzcockpit** (Herausgeber: KoSIT) spezifiziert. Die aktuelle Version des Standards kann im XRepository bezogen werden .

Der Transport der Nachrichten kann je nach Register über verschiedene Transportwege erfolgen (z.B. OSCI, XTA, SOAP, XML/REST etc.). Details dazu werden im Standard XDatenschutzcockpit geregelt und sind nicht Gegenstand dieser Dokumentation.

Zum Schutz der übertragenen personenbezogenen Protokoll- und Inhaltsdaten sowie zur Erfüllung von § 10 Abs. 2 Satz 5 OZG, werden die an das Datenschutzcockpit übermittelten Fachdaten verschlüsselt. Dabei handelt es sich um eine Ende-zu-Ende-Verschlüsselung zwischen dem Browser der Datenschutzcockpit-Nutzer:in und dem jeweiligen Register.

Der vorliegende Abschnitt beschreibt das Vorgehen bei der Authentifizierung am Datenschutzcockpit sowie die Vorgehensweise bei der Ende-zu-Ende-Verschlüsselung der XDatenschutzcockpit-Fachnachrichten. Eine Dokumentation der Inhalte der Fachnachrichten erfolgt nicht, da diese bereits im Fachstandard **XDatenschutzcockpit** beschrieben werden.

Hinweis: In der Dokumentation wird der Begriff Register als Synonym für alle öffentliche Stellen und deren Verfahren (Fachverfahren, Datenbanken, Register usw.) verwendet, die gem. § 9 IDNrG dazu verpflichtet sind, Datenübermittlungen zu protokollieren und an das Datenschutzcockpit zu übermitteln.

2.1 DSC-Request-Key

Die Kommunikation zwischen dem Browser des DSC-Nutzers und dem jeweilig abgefragten Register erfolgt mittels E2E-Verschlüsselung - und zwar im wörtlichen Sinn: Register-Anfragen werden im Browser des Nutzers für das befragte Zielregister verschlüsselt und Register-Antworten werden im Register für den anfragenden Browser verschlüsselt.

Der Browser erzeugt hierzu pro Register-Anfrage einen temporären symmetrischen Schlüssel (DSC-Request-Key), den er sich nur solange merkt, bis die zugehörige Register-Antwort eingetroffen ist (oder ein Timeout eingetreten ist). Mit diesem symmetrischen Schlüssel werden also sowohl der XDSC-Request als auch der XDSC-Response verschlüsselt.

Der symmetrische Schlüssel wird mit dem öffentlichen Zertifikat des Registers aus der Verwaltungs-PKI (abgefragt aus DVDV) verschlüsselt und somit sicher zum Register übermittelt. Das Register kann damit also den XDSC-Request entschlüsseln und den XDSC-Response verschlüsseln.

Die E2E-Verschlüsselung verhindert Man-in-the-Middle-Angriffe (MITM-Angriffe) auf dem Weg vom Browser zum jeweiligen Register, sowohl in zentralen DSC-Komponenten, als auch in sonstigen Zwischenstationen zu den Registern (Intermediäre, Proxies etc.). Keine Komponente zwischen dem Browser des DSC-Nutzers und dem Register kann den symmetrischen Schlüssel entschlüsseln und damit XDSC-Requests einsehen oder valide XDSC-Responses erzeugen.

2.2 DSC-Request-Token (Authentifizierungstoken)

Jede Registerabfrage enthält einen Authentifizierungs-Token, welches die Zulässigkeit der DSC-Registerabfrage (Statusabfrage, Protokolldatenabfrage, Inhaltsdatenabfrage) ausweist. Dieses DSC-Request-Token kann und muss auf der Register-Seite geprüft werden, ob die Register-Anfrage zulässig ist.

Das DSC-Request-Token enthält sowohl Informationen zum anfragenden DSC-Nutzer (Identifikationsnummer) als auch zum angefragten Register sowie mehrere Zeitstempel zur Vermeidung von Replay-Angriffen. Es ist ansonsten zustandslos, für jede Registeranfrage wird ein neues DSC-Request-Token erzeugt (keine Session).

Das DSC-Request-Token wird durch die Identity-Provider-Komponente (IDP) ausgestellt, welche auch den Login-Kontext des DSC-Nutzers (mit Vertrauensniveau hoch, also eID) umsetzt. Der Browser des DSC-Nutzers erfragt pro Register-Anfrage ein DSC-Request-Token beim IDP. Das DSC-Request-Token wird direkt aus dem Login-Kontext des Nutzers abgeleitet und mit der Register-spezifischen Anfrage verschränkt und durch den IDP signiert. Der IDP agiert somit als zentraler Vertrauensdienst, sowohl für die eID-Authentifizierung auf Basis der eID-Daten, als auch für die Ausstellung von DSC-Request-Tokens.

Die DSC-Backend-Komponenten DSC-Backend und IDP sind ansonsten vollständig voneinander separiert und haben keine direkten Schnittstellen miteinander.

Das DSC-Request-Token verhindert Man-in-the-Middle-Angriffe (MITM-Angriffe) auf dem Weg vom Browser zum jeweiligen Register, sowohl in zentralen DSC-Komponenten, als auch in sonstigen Zwischenstationen zu den Registern (Intermediäre, Proxies etc.). Keine andere Komponente kann valide DSC-Request-Tokens erzeugen, da diese mit dem privaten

Schlüssel des DSC-IAM signiert werden. Diese Signatur muss auf Register-Seite geprüft werden (gegen DVDV-Eintrag). Niemand kann damit eine andere Identität vorgeben oder ohne eID-Kontext handeln. Der DSC-Nutzer kann im Browser auch keine andere IdNr vorspiegeln, da diese Bestandteil des signierten Tokens ist.

3 Laufzeitsicht

Die folgende Laufzeitsicht basiert auf folgenden Vorbedingungen:

- Ein DSC-Nutzer ist im DSC-IAM authentifiziert
  - Bestehender Login-Kontext mit hohem Vertrauensniveau (eID)
  - Identifikationsnummer (IDNr) liegt im IAM vor
- Einzelnes Register wird angefragt mit ServiceUri und AuthorityKey
  - Register hat XDSC-Schnittstelle und DVDV-Eintrag mit ServiceUri und AuthorityKey
  - DSC-Frontend kennt über vorhergehende Schritte ServiceUri und AuthorityKey

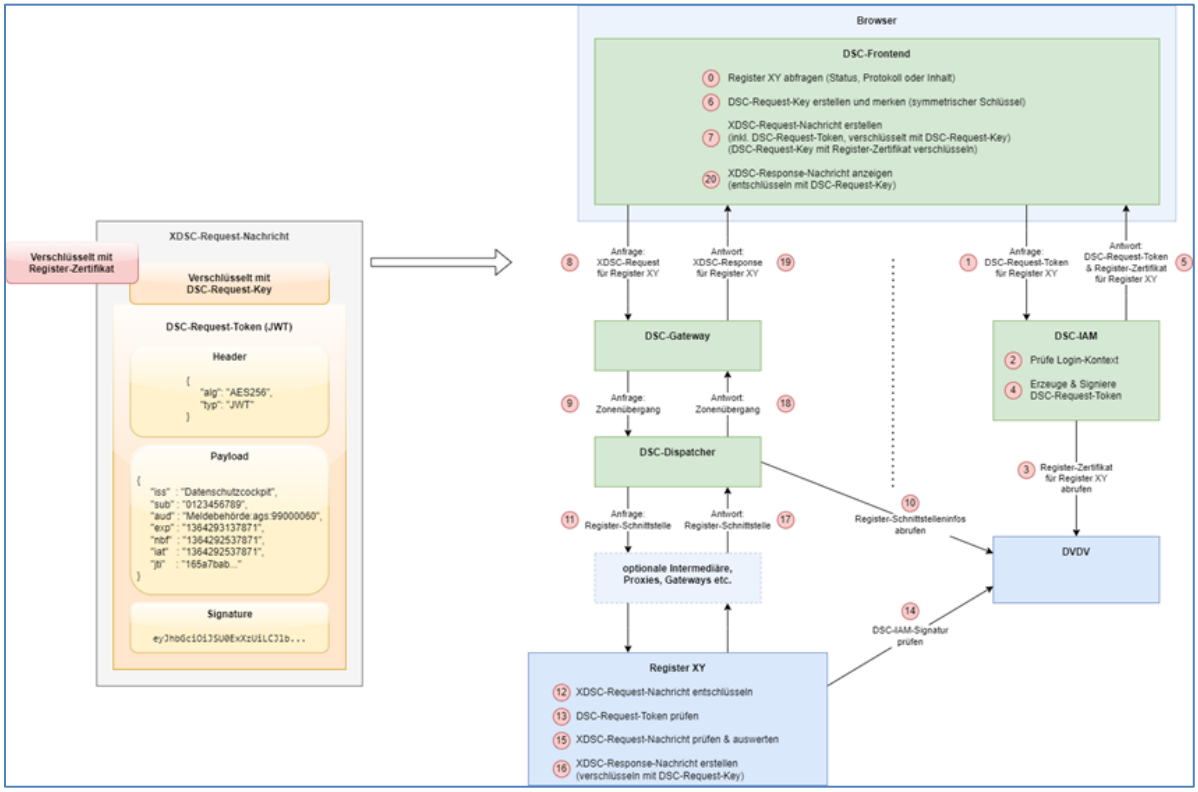

Abbildung 1: Laufzeitsicht - Registerabfrage

Schritt	Titel	Akteur	Beschreibung	Übermittelte Daten
0	Register XY abfragen	DSC-Frontend	Anfrage an Register XY wird im DSC-Frontend initiiert	
1	Anfrage: DSC-Request-Token für Register XY	DSC-Frontend	DSC-Frontend erfragt beim DSC-IAM ein DSC-Request-Token für die geplante XDSC-Anfrage an Register XY DSC-Frontend übergibt dazu ServiceUri und AuthorityKey des Registers an den IAM	IAM-Access-Token im Authorization-Header CorrelationId im Header Liste mit RegisterAccessTokenRequest im Body (enthält jeweils: registerId, authorityKey, serviceUri)
2	Prüfe Login-Kontext	DSC-IAM	DSC-IAM prüft, ob der DSC-Nutzer eine aktive Login-Session mit Vertrauensniveau hoch (eID) hat und ob die IDNr vorliegt	
3	Register-Zertifikat für Register XY abrufen	DSC-IAM	DSC-IAM ruft mit ServiceUri und AuthorityKey das öffentliche Zertifikat der XDSC-Schnittstelle des Registers XY bei DVDV ab Antwortet das erste konfigurierte DVDV-System nicht oder mit einem Fehler, wird die Anfrage an der Reihe nach alle konfigurierten DVDV-Fallback-Systeme wiederholt, bis eines erfolgreich Antwortet oder alle konfigurierten DVDV-Systeme angefragt wurden	Request an DVDV: • OrganizationKey (aka authorityKey) • serviceUri Response von DVDV: • x509-Zertifikat des Registers
4	Erzeuge & Signiere DSC-Request-Token	DSC-IAM	DSC-IAM erzeugt ein DSC-Request-Token mit folgenden Infos: • Issuer: DSC-IAM • Subject: <IDNr> • Audience: <ServiceUri>:<AuthorityKey> • Zeitstempel wie Expire etc. DSC-IAM signiert das DSC-Request-Token mit einer DSC-IAM-Signatur, wo der öffentliche Teil des Schlüssels in DVDV abrufbar ist (für spätere Prüfung im Register)	
5	Antwort: DSC-Request-Token & Register-Zertifikat für Register XY	DSC-IAM	DSC-IAM liefert erzeugten DSC-Request-Token und das öffentliche Zertifikat der XDSC-Schnittstelle des Registers XY an das DSC-Frontend aus	Liste mit RegisterAccessToken (enthält jeweils: registerId, DSC-Request-Token, öffentliche Zertifikat des Registers)

Schritt	Titel	Akteur	Beschreibung	Übermittelte Daten
6	DSC-Request-Key erzeugen und merken	DSC-Frontend	DSC-Frontend erzeugt über Web Crypto API einen symmetrischen Schlüssel • XDSC-Request-Key wird mit XDSC-Anfrage an Register übertragen und dort für die Entschlüsselung von XDSC-Anfrage und Verschlüsselung der XDSC-Antwort benutzt • XDSC-Request-Key ist nur temporär für die folgende XDSC-Antwort gültig • Symmetrischer Schlüssel ausreichend, da kein öffentlicher Schlüsselteil benötigt wird und der Schlüssel verschlüsselt übertragen wird • Symmetrische Schlüssel haben keine Längenbegrenzung für zu verschlüsselnde Daten XDSC-Request-Key wird nach zugehöriger XDSC-Antwort verworfen bzw. nach Timeout (wenn keine Antwort)	
7	XDSC-Request-Nachricht erstellen	DSC-Frontend	DSC-Frontend erzeugt XDSC-Request-Nachricht für Statusdaten-, Protokolldaten- oder Inhaltsdatenabfrage Diese Nachrichten enthalten im Feld "clientsessionToken" den XDSC-Request-Token (siehe Beispielnachrichten) DSC-Frontend verschlüsselt mit dem XDSC-Request-Key über Web Crypto API den Content der XDSC-Request-Nachricht, nicht das Root-Element (inkl. DSC-Request-Token) DSC-Frontend verschlüsselt mit dem öffentlichen Zertifikat der XDSC-Schnittstelle des Registers XY über Web Crypto API den XDSC-Request-Key Der verschlüsselte XDSC-Request-Key wird innerhalb des Root-Elements, neben dem verschlüsselten Content versendet	
8	Anfrage: XDSC-Request für Register XY	DSC-Frontend	DSC-Frontend sendet verschlüsselte XDSC-Request-Nachricht (mit XDSC-Request-Key), sowie ServiceUri und AuthorityKey des Registers XY an REST-Service im DSC-Gateway	je nach Anfrage: • requestAbfragEStatus,

Schritt	Titel	Akteur	Beschreibung	Übermittelte Daten
			DSC-Gateway ist Internet-nahes System mit REST-Services für DSC-Frontend Single-Page-App (SPA)	• requestAbfrageListeProtokolldaten • oder requestAbfrageInhaltsdaten aus dem Kosit XDSC-Standard Version 0.1.0-pilot (abgesehen vom RootElement jeweils verschlüsselt)
9	Anfrage: Zonenübergang	DSC-Gateway	DSC-Gateway sendet verschlüsselte XDSC-Request-Nachricht (mit XDSC-Request-Key), sowie ServiceUri und AuthorityKey des Registers XY an REST-Service im DSC-Dispatcher DSC-Dispatcher ist nicht direkt aus Internet aufrufbar (Entkopplung) und bildet ggf. den Zonenübergang zu Register-nahen Netzen	siehe 8
10	Register-Schnittstelleninfos abfragen	DSC-Dispatcher	DSC-Gateway ermittelt mit ServiceUri und AuthorityKey des Registers XY die entsprechenden Schnittstelleninfos (Transportkanal, Endpunkte, Zertifikate)	aktuell Abfrage an RegisterDB, nicht an DVDV Request: • registerId • correlationId Response: • BaseUrl des Registers
11	Anfrage: Register-Schnittstelle	DSC-Dispatcher	DSC-Dispatcher sendet verschlüsselte XDSC-Request-Nachricht an Register XY (ggf. indirekt über Intermediär, Gateway etc.)	siehe 8
12	XDSC-Request-Nachricht entschlüsseln	Register XY	Register XY (bzw. eine vorgeschaltete Komponente in der Registersphäre) entschlüsselt XDSC-Request-Key mit seinem privaten Schlüssel • ...der zum DVDV-eingetragenen öffentlichen Zertifikat der XDSC-Schnittstelle des Registers XY passt Register XY (bzw. eine vorgeschaltete Komponente in der Registersphäre)	

Schritt	Titel	Akteur	Beschreibung	Übermittelte Daten
			entschlüsselt XDSC-Request-Anfrage mit XDSC-Request-Key	
13	DSC-Request-Token prüfen	Register XY	Register XY prüft die Angaben im DSC-Register-Token • Ist Register der Empfänger • Sind Zeitstempel OK • Ist IdNr OK • Ist Nachricht kein Replay etc.	
14	DSC-IAM-Signatur prüfen	Register XY	Register XY prüft die Signatur des DSC-Register-Tokens über DVDV • Muss zu DVDV-Eintrag für DSC-IAM passen	siehe 3
15	XDSC-Request-Nachricht prüfen und auswerten	Register XY	Register XY validiert die Nachricht gegen ein Schema und gegen Angriffe (XML-Bomben etc.) Register XY wertet die Anfrage aus: Statusdatenabfrage / Protokolldatenanfrage / Inhaltsdatenanfrage für IdNr	
16	XDSC-Response-Nachricht erstellen und verschlüsseln	Register XY	Register XY erstellt die XDSC-Response-Nachricht (Auskunft oder Fehler) Register XY verschlüsselt die XDSC-Response-Nachricht mit dem DSC-Request-Key (temporärer Schlüssel aus Browser, der im DSC-Request mitgeliefert wurde)	
17	Antwort: Register-Schnittstelle	Register XY	Register XY antwortet mit verschlüsselter XDSC-Response-Nachricht an DSC-Dispatcher (ggf. indirekt über Intermediär, Gateway etc.) • Schritt 11 zu 17 ist synchron oder asynchron: Aber möglichst zügig im Gesamtablauf, da der Nutzer wartet	je nach Anfrage: • responseAbfrageStatus • responseAbfrageListeProtokolldaten • responseAbfrageInhaltsdaten aus dem Kosit XDSC-Standard Version 0.1.0-pilot (abgesehen vom Root-Element jeweils verschlüsselt)
18	Antwort: Zonenübergang	DSC-Dispatcher	DSC-Dispatcher antwortet mit verschlüsselter XDSC-Response-Nachricht an DSC-Gateway Zonenentkopplung Internet - Register	siehe 17

Schritt	Titel	Akteur	Beschreibung	Übermittelte Daten
19	Antwort: DSC-Response für Register XY	DSC-Gateway	DSC-Gateway antwortet mit verschlüsselter XDSC-Response-Nachricht an DSC-Frontend	siehe 17
20	XDSC-Response-Nachricht anzeigen	DSC-Frontend	DSC-Frontend entschlüsselt über Web Crypto API die XDSC-Response-Nachricht mit seinem für die Anfrage gemerkten DSC-Request-Key • Schlüssel wird danach verworfen DSC-Frontend zeigt Daten aus der Nachricht nutzerfreundlich an (Layouting, keine großen Transformationen) • ggf. nicht in einem Schritt, sondern schrittweise und interaktiv über DOM-Manipulationen • Antwortdaten werden lediglich in der angezeigten SPA-Anzeigeseite (DOM-Struktur) vorgehalten und sind nach Seitenwechsel sofort verloren	

4 Token-Authentifizierung

Die Authentifizierungs-Tokens werden durch das Datenschutzcockpit IAM im Auftrag durch den Browser der Nutzer:in erstellt, wobei die Login-Session der Nutzer:in genutzt wird, um die Tokens aus dem Security Context abzuleiten. Dazu muss Keycloak um eine Schnittstelle ergänzt werden, welche für v**iele** Register in einem Aufruf diese Tokens erzeugt.

In den Request-Fachnachrichten von XDatenschutzcockpit wird ein Authentifizierungs-Token im Stil eines JSON Web Token (JWT) im XML-Element **<dscRequestToken>** übermittelt. Im Spezifikationsdokument *XDatenschutzcockpit Teil2-4: XML über HTTP* wird in Abschnitt 2 (*XML über HTTP)* sowie im Abschnitt 4 (*Rahmenbedingungen*) darauf Bezug genommen. In dem XML-Element ist das Token in der signierten Form *JWS Compact Serialization* gemäß RFC 7515 enthalten. Die Attributwerte im Token sind im Zeichensatz UTF-8 kodiert.

Beispiel:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c

Zur Verarbeitung von JWT gibt es Bibliotheken in verschiedenen Sprachen, z.B. hier (für Beispiel-Token oben) und hier.

Im Folgenden wird der innere Aufbau des Tokens beschrieben:

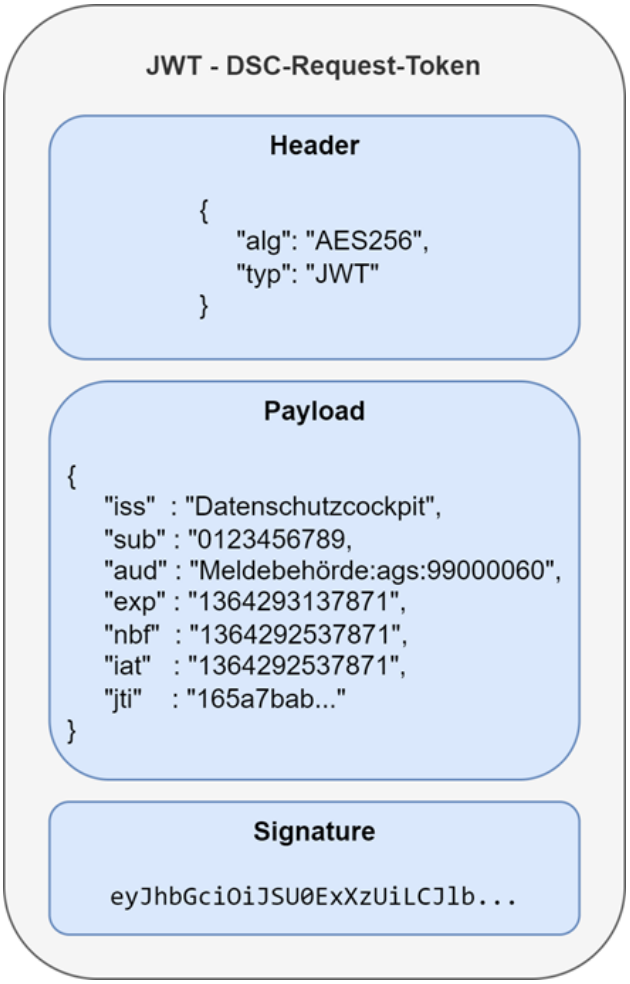

Abbildung 2: Aufbau DSC-Request-Token

Die RS256 Signatur nutzt ein RSA-Zertifikat in Kombination mit dem Hash-Algorithmus SHA-256, wobei das RSA-Zertifikat nicht mit im Token übertragen wird. Die Register müssen diese Signatur mit dem öffentlichen Datenschutzcockpit IAM-Zertifikat validieren, welches entweder bei den Registern hinterlegt ist oder über DVDV bezogen werden kann. Neben der Signatur-Validierung sind auch alle Token-Attribute (Claims) in der Register-Sphäre zu validieren, um z.B. Replay-Angriffe für die Tokens zu unterbinden. Dazu gibt es folgende Fehlerfälle:

Validierung	Fehlercode für Validierungsfehler
JSON Web Token Form validieren	J001 Das Token ist nicht valide. Es muss die Form eines JSON Web Token erfüllen.
JWT Signatur-Algorithmus prüfen	J002 Die Token-Signatur ist nicht valide. Es wird der Signatur-Algorithmus RS256 erwartet.
JWT Signatur prüfen	J003 Die Token-Signatur ist nicht valide. Die Signatur passt nicht zum hinterlegten Zertifikat im Register.
JWT Test-Aufruf auf Produktivdaten prüfen	J004 Das Token ist nicht valide. Es ist als Test-Aufruf markiert und versucht auf Echtdaten zuzugreifen.

Die im Token zu verwendenden Standard-Claims sind hier beschrieben und diese Token-Attribute müssen ebenfalls auf Register-Seite validiert werden:

Feld	Name	Beschreibung	Validierung auf Register-Seite und ggf. Fehlertyp	Fehlercode für Validierungsfehler
iss	Issuer	Der Aussteller des Tokens. Für XDatenschutzcockpit ist hier der feste Wert "Datenschutzcockpit" eingetragen.	Das Attribut muss auf Register-Seite validiert werden. Der Wert ist gültig, wenn er den Wert "Datenschutzcockpit" enthält.	J011 Das Token-Attribut 'Issuer' ist nicht valide. Es muss gleich 'Datenschutzcockpit' sein.
sub	Subject	Definiert für welches Subjekt die Claims gelten. Das `sub`-Feld definiert also für wen oder was die Claims getätigt werden. Für XDatenschutzcockpit ist hier die Identifikationsnummer (IDNr) der Person eingetragen, für welche die	Das Attribut muss auf Register-Seite nur bzgl. der Form validiert werden. Eine Inhaltsvalidierung ist hier nicht möglich, entweder existiert die	J012 Das Token-Attribut 'Subject' ist nicht valide. Es muss die Form einer Identifikationsnummer haben.

Feld	Name	Beschreibung	Validierung auf Register-Seite und ggf. Fehlertyp	Fehlercode für Validierungsfehler
		XDatenschutzcockpit-Anfrage gestellt wird.	Person mit der Identifikationsnummer in den Daten oder eben nicht.	
aud	Audience	Die Zieldomäne, für die das Token ausgestellt wurde. Für XDatenschutzcockpit ist hier eine konkrete ID des Registers bzw. Register-Mandanten (also inkl. Zuständigkeits-Kodierung, wenn vorhanden) eingetragen, an welches die XDatenschutzcockpit-Anfrage gestellt wird. Sie setzt sich so zusammen: <Fachdomäne>:<Präfix>:<Code> Beispiele: "Meldebehörde:ags:99000060" oder "Bzst:dbs:490010010000" Präfix und Code entsprechen dabei der Systematik aus DVDV und XInneres-Kopfdaten.	Das Attribut muss auf Register-Seite validiert werden, damit keine fremden Register Tokens durch Replay-Angriffe für andere Register missbraucht werden können. Der Wert ist gültig, wenn er der eigenen Register-ID entspricht.	J013 Das Token-Attribut 'Audience' ist nicht valide. Es muss der Register-Id entsprechen.
exp	Expiration Time	Das Ablaufdatum des Tokens in Unixzeit, also der Anzahl der Sekunden seit `1970-01-01T00:00:00Z`.	Das Attribut muss auf Register-Seite validiert werden, damit die Tokens nicht für Replay-Angriffe missbraucht werden können. Der Wert ist gültig, wenn er in der Zukunft liegt.	J014 Das Token-Attribut 'Expiration Time' ist nicht valide. Es muss die Form einer Unixzeit haben und in der Zukunft liegen.
nbf	Not Before	Die Unixzeit, ab der das Token gültig ist.	Das Attribut muss auf Register-Seite validiert werden, damit die Tokens nicht für Replay-	J015 Das Token-Attribut 'Not Before' ist nicht valide. Es muss die Form einer Unixzeit

Feld	Name	Beschreibung	Validierung auf Register-Seite und ggf. Fehlertyp	Fehlercode für Validierungsfehler
			Angriffe missbraucht werden können. Der Wert ist gültig, wenn er in der Vergangenheit liegt.	haben und in der Vergangenheit liegen.
iat	Issued At	Die Unixzeit, zu der das Token ausgestellt wurde.	Das Attribut muss auf Register-Seite nur bzgl. der Form validiert werden. Über exp, nbf und iat kann festgestellt werden, ob Datenschutzcockpit und Register in etwa eine synchrone Zeit nutzen. Die Werte exp und nbf werden durch das Datenschutzcockpit nicht zu strikt gesetzt, z.B. 3 Minuten in der Vergangenheit und 3 Minuten in der Zukunft, um hier bei Uhren-Gangunterschieden nicht zu schnell Fehler zu erhalten.	J016 Das Token-Attribut 'Issued At' ist nicht valide. Es muss die Form einer Unixzeit haben und zwischen 'Expiration Time' und 'Not Before' liegen.
jti	JWT ID	Eine eindeutige case-sensitive Zeichenfolge, welche das Token eindeutig identifiziert. Hiermit kann verhindert werden, dass das Token repliziert wird. Hierbei kann es sich etwa um eine durchgezählte Nummer, einen GUID oder einen Hashwert handeln. Falls der Token-Empfänger von mehreren	Das Attribut muss auf Register-Seite validiert werden, damit die Tokens nicht für Replay-Angriffe missbraucht werden können. Die ID darf nur einmal verwendet werden.	J017 Das Token-Attribut 'JWT ID' ist nicht valide. Es muss die Form einer ID haben und darf nicht mehrfach benutzt werden.

Feld	Name	Beschreibung	Validierung auf Register-Seite und ggf. Fehlertyp	Fehlercode für Validierungsfehler
		Ausstellern einen Token empfängt, kann es sein, dass die JWT ID nicht eindeutig ist. Durch die Kombination des Ausstellers (iss) und der JWT ID (jti) kann diese wieder eindeutig werden. Für das Datenschutzcockpit wird hier eine ID hochgezählt.		

Die Register erhalten über dieses Token die Nutzer-Identifikation und die Nutzer-Zustimmung (über ePerso-Login), so dass durch die Art der Anfrage (z.B. Inhaltsdatenanfrage) klar wird, welche personenbezogene Daten zu welcher Nutzer:in ausgeliefert werden sollen und dürfen.

5 Ende-zu-Ende-Verschlüsselung

Die Ende-zu-Ende-Verschlüsselung zwischen dem Browser der Datenschutzcockpit-Nutzer:in und dem jeweiligen Register erfordert eine zusätzliche Verschlüsselungs-Ebene, welche über die durch OSCI / XTA etc. bereitgestellten Konzepte hinausgeht. Die serverseitigen Komponenten des Datenschutzcockpits dürfen weder in die Anfragen an die Register eingreifen, noch in die Antworten von den Registern hineinsehen können, da das Datenschutzcockpit register- und bereichsübergreifend agiert. Siehe hierzu auch § 10 Abs.2 Satz 5 OZG.

Die XDatenschutzcockpit-Fachdaten (ohne den Nachrichtenkopf) werden mittels **XML Encryption** verschlüsselt (zusätzlich zu weiteren Verschlüsselungen mittels OSCI / XTA / SSL etc.).

Das folgende Beispiel zeigt eine unverschlüsselte Anfrage des Datenschutzcockpits an ein Register auf Basis einer XDatenschutzcockpit-Fachnachricht:

```
<?xml version="1.0" encoding="UTF-8"?>
<abfrageStatus
xmlns:xdsc="http://xoev.de/datenschutzcockpit/xdatenschutzcockpit/datentypen_xml
/1.0"
 ...
 correlationID="ea801222-f160-4ced-b1cb-792db7375660">

    <xdsc:requestAbfrageStatus>

<xdsc:DSCRequestToken>eyJhbGciOiNIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3O
DkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf3
6POk6yJV_adQssw5c</xdsc:DSCRequestToken>
    </xdsc:requestAbfrageStatus>

</abfrageStatus>
```

Code Block 1 Unverschlüsselte Request-Nachricht "abfrageStatus"

Das nachfolgende Beispiel zeigt die XDatenschutzcockpit-Nachricht in verschlüsselter Form. Hierbei werden im Beispiel lediglich die Elemente *<xdsc:requestAbfrageStatus>* sowie *<xdsc:DSCRequestToken>* im Browser der Datenschutzcockpit-Nutzer:in verschlüsselt. Die Kopfdaten, insbesondere die *correlationID*, werden von den Backend-Komponente des Datenschutzcockpits in unverschlüsselter Form ergänzt.

```
<?xml version="1.0" encoding="UTF-8"?>
<abfrageStatus
xmlns:xdsc="http://xoev.de/datenschutzcockpit/xdatenschutzcockpit/datentypen_xml
/1.0"
 ...
 correlationID="ea801222-f160-4ced-b1cb-792db7375660">

   <xenc:EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2009/xmlenc11#aes256-
gcm" />
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <xenc:EncryptedKey>
          <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p">
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
/>
          </xenc:EncryptionMethod>
          <ds:KeyInfo>
            <ds:X509Data><ds:X509Certificate>MIIEDzCCAveg... öffentliches DVDV-
Zertifikat des Registers... q3uaLvlAUo=</ds:X509Certificate></ds:X509Data>
          </ds:KeyInfo>
          <xenc:CipherData>
            <xenc:CipherValue>sGH0hhzkjmLWYYY0gyQMampDM... verschlüsselter
symmetrischer Schlüssel ...gewHMbtZafk1MHh9A==</xenc:CipherValue>
          </xenc:CipherData>
        </xenc:EncryptedKey>
      </ds:KeyInfo>
      <xenc:CipherData>
          <xenc:CipherValue>V3Vb1vDl055Lp92zvK..... verschlüsselte Fachdaten....
kNzP6xTu7/L9EMAeU</xenc:CipherValue>
      </xenc:CipherData>
  </xenc:EncryptedData>

</abfrageStatus>
```

Code Block 2 Verschlüsselte Request-Nachricht "abfrageStatus"

Bei der XML Encryption wird eine Hybridverschlüsselung aus rsa-oaep-mgf1p (asymmetrisch) und aes256-gcm (symmetrisch) verwendet (siehe Beispiel-Nachricht oben), die gegen Padding-Oracle-Angriffe abgesichert ist. Der Hash-Algorithmus sha1 wird hierbei als Parameter des OAEP-Algorithmus verwendet und gilt in diesem Kontext als sicher (siehe RFC 3447 unter B.1 und XML Encryption unter 5.4.2).

Der **symmetrische AES 256 Schlüssel** wird pro Register-Anfrage **im Browser des Datenschutzcockpit-Nutzers erzeugt** und im Element *<xenc:EncryptedKey>/<xenc:CipherData>/<xenc:CipherValue>* an das Register übermittelt. .

Der **asymmetrische RSA Schlüssel** zur Verschlüsselung des symmetrischen Schlüssels ist der des jeweilig angefragten Registers. Dazu wird das **öffentliche DVDV-Inhaltsdatenzertifilat des angefragten Registers** verwendet und im Element *ds:KeyInfo>/<ds:X509Data>* übermittelt. Das Register kann dazu seinen passenden privaten Schlüssel ermitteln.

Das folgende Beispiel zeigt eine Antwort des Registers an das Datenschutzcockpit ohne Verschlüsselung.

```
<?xml version="1.0" encoding="UTF-8"?>
<responseAbfrageStatus
xmlns:xdsc="http://xoev.de/datenschutzcockpit/xdatenschutzcockpit/datentypen_xml
/1.0"
 ...
 correlationId="ea801222-f160-4ced-b1cb-792db7375660">

    <istUebermittlungErfolgt>true</istUebermittlungErfolgt>
    <zeitpunktLetzteUebermittlung>2023-04-
26T00:00:00</zeitpunktLetzteUebermittlung>
    <zeitraum>
        <startZeitpunkt>2021-07-01T00:00:00</startZeitpunkt>
        <endZeitpunkt>2023-06-30T00:00:00</endZeitpunkt>
    </zeitraum>

</responseAbfrageStatus>
```

Code Block 3 Unverschlüsselte Response-Nachricht "responseAntwortStatus"

Im nachfolgenden Beispiel wurde die Antwortnachricht durch das Register verschlüsselt. Dabei wird derselbe **symmetrische AES 256 Schlüssel** aus dem Request für die Verschlüsselung der Fachdaten verwendet! Der KeyInfo-Teil kann in der XDatenschutzcockpit-Antwort entfallen, da der Browser der Nutzer:in den Key bereits kennt.

Auf Seiten des Datenschutzcockpits wird nicht die gesamte Nachricht bis zum Browser der Datenschutzcockpit-Nutzer:in übertragen. Die Nachrichten-Kopfdaten, konkret: die *correlationID* werden vor Weitergabe an den Browser entfernt.

```
<?xml version="1.0" encoding="UTF-8"?>
<responseAbfrageStatus
xmlns:xdsc="http://xoev.de/datenschutzcockpit/xdatenschutzcockpit/datentypen_xml
/1.0"
 ...
 correlationId="ea801222-f160-4ced-b1cb-792db7375660">

 <xenc:EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2009/xmlenc11#aes256-
gcm" />
      <xenc:CipherData>
          <xenc:CipherValue>V3Vb1vDl055Lp92zvK..... Verschlüsselte Fachdaten...
kNzP6xTu7/L9EMAeU</xenc:CipherValue>
      </xenc:CipherData>
  </xenc:EncryptedData>

</responseAbfrageStatus>
```

Code Block 4 Verschlüsselte Response-Nachricht "responseAntwortStatus"

6 Weitere Hinweise zur Verarbeitung der XDSC-Fachnachrichten

Im Fachstandard wird aktuell (Stand: Januar 2023) zwischen drei Nachrichtentypen unterschieden:

Requests (Anfragen des DSC an das Register)

- Es ist zwingend erforderlich, dass Requests auf Register-Seite gegen das XSD-Schema validiert werden, denn diese Nachrichten werden teilweise im Browser der Datenschutzcockpit-Nutzers zusammengesetzt und könnten manipuliertes XML enthalten. Dabei sollte vor einer Schema-Validierung die erwartete Maximallänge der Nachricht geprüft werden und im XML-Parser die vorhandenen Sicherheitsschalter gesetzt werden, um z.B. DTDs, Entity Expansion und External DTD bzw. Stylesheet Loading zu unterbinden.
- Bei einer Inhaltsdatenabfrage entspricht die uebermittlungId der uebermittlungId aus den der Antwortnachricht des Registers zur Protokolldatenanfrage. Das Register bestimmt die Form der uebermittlungId um eine Datenaustausch-Nachricht bzw. deren Protokolldaten eindeutig zuordnen zu können. Das Register muss bei der Abfrage prüfen, ob die Nutzer:in aus dem Authentifizierungs-Token für diese Übermittlungs-ID auskunftsberechtigt ist.

Responses (Antworten der Register an das DSC)

- Es ist die Aufgabe der Register, nur solche Daten aus der Register-Sphäre auszuspielen, für deren Einsicht der identifizierte Nutzer aus dem Authentifizierungs-Token berechtigt ist. Protokolldaten, die nicht zur Anzeige im Datenschutzcockpit bestimmt sind, dürfen nicht an das Datenschutzcockpit übermittelt werden. Sollten in einem Datenaustausch mehrere Personen enthalten sein, z.B. bei einer Geburtsmeldung Mutter, Vater und Kind, so ist es die Aufgabe des Registers, diese Daten für die Datenschutzcockpit-Antwort ggf. aufzusplitten und zu filtern, so dass z.B. die Mutter keine Daten des Vaters sieht und umgekehrt.
- Es gilt, dass nicht zu allen Datenkategorien, zu den Protokolldaten gemeldet wurden, auch Inhaltsdaten übermittelt werden müssen. Sollte es beispielsweise Attribute mit Sperrvermerk geben, so würden diese nicht an das Datenschutzcockpit übermittelt werden. Die Inhaltsdatenbereitstellung darf also weniger Attribute zurückmelden als die zugehörige Protokolldatenbereitstellung.
- Das Register steht in der Verantwortung, dass die übermittelten Inhalte für Bürger:innen verständlich sind. Die Protokoll- und Inhaltsdaten werden ohne weitere Verarbeitung oder Anreicherung im Datenschutzcockpit zur Anzeige gebracht, es

erfolgt lediglich ein visuelles Styling (Layout, Fonts, Farben, Datums-Formatierung etc.).

- Das Register ist für die halte der Daten im Datenschutzcockpit verantwortlich. Das Datenschutzcockpit tritt nur als vermittelnder Dienst auf. Dazu zählt beispielsweise die Sicherstellung der Virenfreiheit von übermittelten Dateien.

Error (Fachlicher Fehler bei der Abfrage des DSC am Register)

- Der Inhalt von Errornachrichten ist unverschlüsselt, um diesen an Fehlerlogging der Backend-Komponenten auswerten zu können.
- Error-Nachrichten dürfen keine personenbezogenen Daten enthalten, da diese unverschlüsselt übermittelt werden.