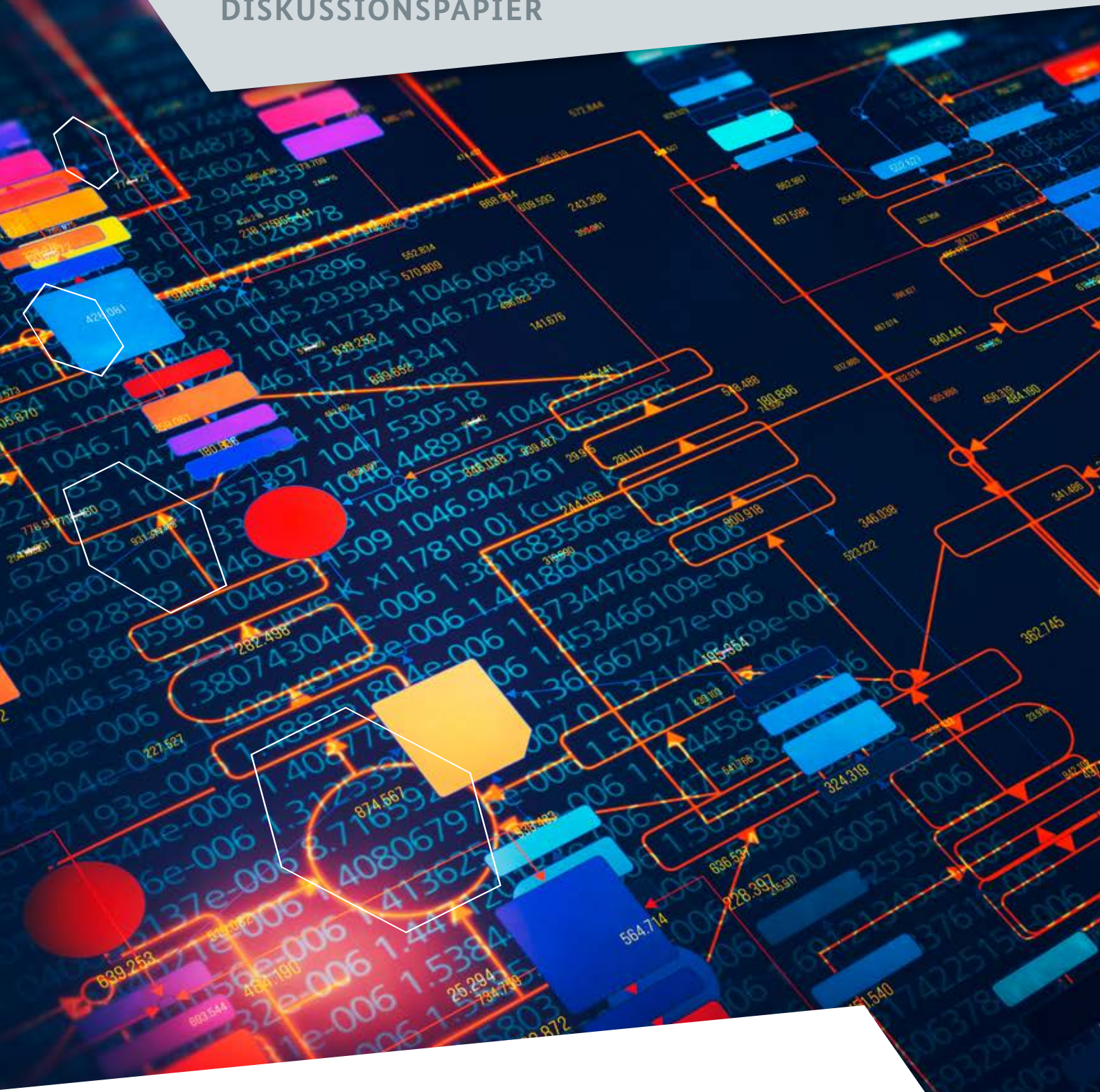
PLATTFORM INDUSTRIE4.0

DISKUSSIONSPAPIER



Sicherer Downloadservice

Impressum

Herausgeber
Bundesministerium für Wirtschaft
und Energie (BMWi)
Öffentlichkeitsarbeit
11019 Berlin
www.bmwi.de

Redaktionelle Verantwortung
Plattform Industrie 4.0
Bülowstraße 78
10783 Berlin

Gestaltung
PRpetuum GmbH, 80801 München

Stand
Oktober 2020

Bildnachweis
Plattform Industrie 4.0;
spainter_vfx – iStockphoto (Titel)

Diese und weitere Broschüren erhalten Sie bei:
Bundesministerium für Wirtschaft und Energie
Referat Öffentlichkeitsarbeit
E-Mail: publikationen@bundesregierung.de
www.bmwi.de

Zentraler Bestellservice:
Telefon: 030 182722721
Bestellfax: 030 18102722721

Inhalt

1. Einleitung

Der interoperable Bezug von Engineeringdaten ist ein Element zur Umsetzung von Industrie 4.0. Die Anforderungen an Interoperabilität sind dabei sowohl an die übertragenen Inhalte und Daten zu stellen als auch an die Formate und Protokolle. Die Interoperabilität schließt die Security-Maßnahmen mit ein, mit denen Vertraulichkeit, Integrität und Verfügbarkeit entsprechend den Anforderungen des Anwendungsszenarios umgesetzt werden.

1.1 Inhalt und Ziel dieses Diskussionspapiers

Dieses Dokument greift die Überlegungen des Diskussionspapiers „Sicherer Bezug von CAE-Daten“ (1) auf und entwickelt die Überlegungen zur Übertragung in technischer Hinsicht weiter. Insbesondere werden Kommunikationsprotokolle (HTTPS/REST), Identitätsmanagement und Authentifizierung am Beispiel von OpenID Connect diskutiert. Die Formate und Inhalte der Daten werden in anderen Dokumenten diskutiert, vornehmlich in der Serie „Verwaltungsschale im Detail“ (2). In diesem Diskussionspapier wird ein Lösungskonzept vorgestellt, das den vollautomatischen Bezug von Engineeringdaten einschließlich der Authentifikation und des Rechtemanagements unter der Weiterentwicklung existierender Standards ermöglicht. Ziel ist es, die Weiterentwicklung entsprechender Standards anzustoßen.

Die technische Tiefe dieses Dokuments ist ausreichend, um Demonstratoren für das beschriebene Anwendungsszenario aufzubauen.

Dieses Dokument richtet sich an die technisch versierte Leserin und den technisch versierten Leser.

2. Anwendungsszenario

Im Dokument „Verwaltungsschale im Detail“ (2) wird die Übertragung von Daten zwischen den beteiligten Stakeholdern diskutiert.

2.1 Überblick

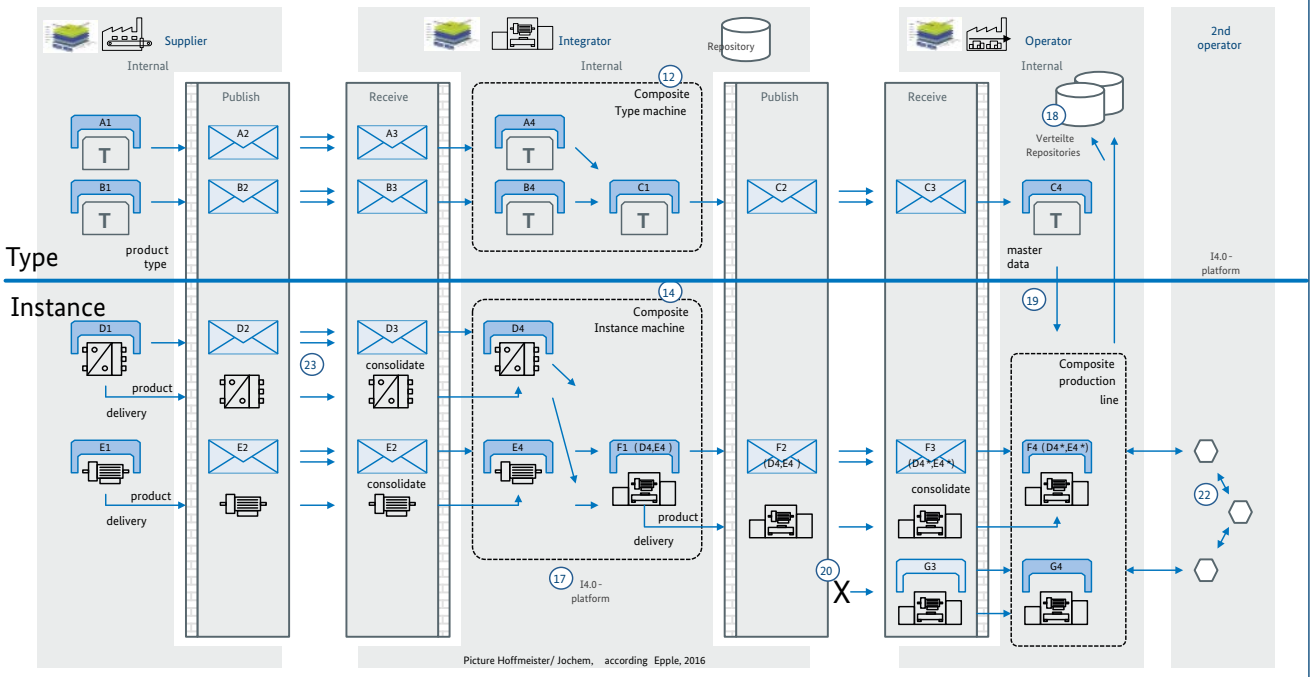
Abbildung 1 zeigt die Stationen der Typ- und der Instanzdaten, die entlang der Wertschöpfungskette, ausgehend vom Supplier über den Integrator zum Operator, übertragen werden. Die jeweiligen Typdaten dienen der Gestaltung der Automatisierungslösung und werden zum Zeitpunkt des Engineerings benötigt, bevor die Maschine oder Produktionsanlage tatsächlich aufgebaut wird.

Die Instanzdaten gelten dann für einzelne Exemplare der verwendeten Produkte bzw. der aus ihnen erstellten Maschinen oder Anlagen und können z. B. Qualitäts- oder Kalibrierdaten enthalten, um die Gesamtlösung damit zu unterstützen.

2.2 Übertragung von Engineeringdaten

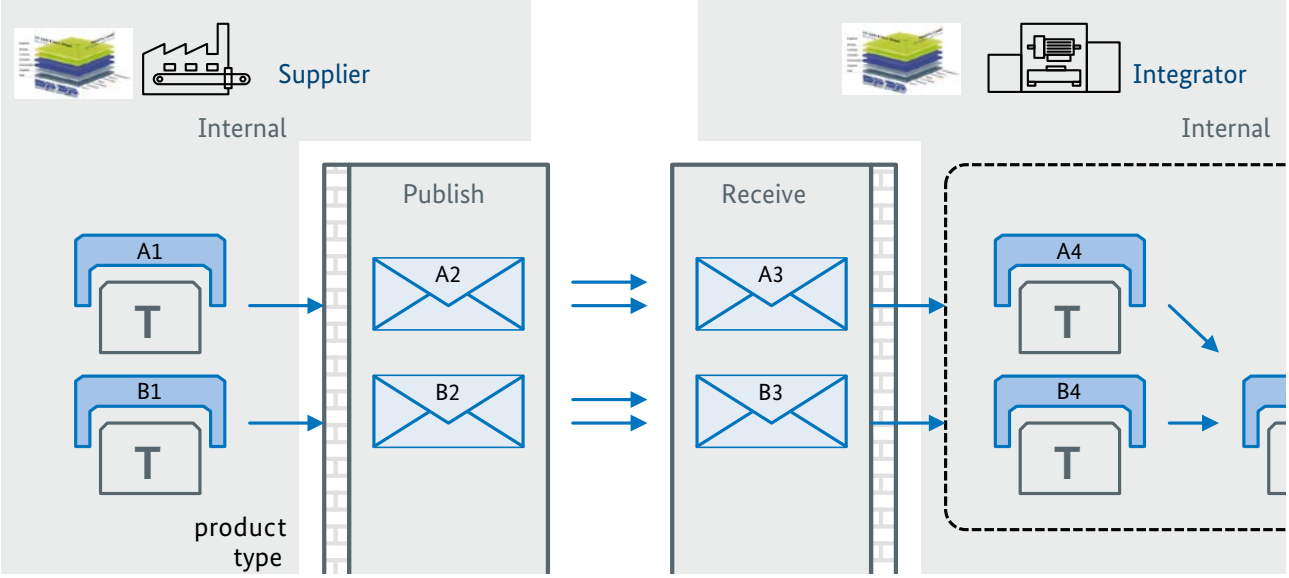
Abbildung 2 zeigt beispielhaft die Übertragung von Engineeringdaten von einem Hersteller zu einem Integrator. Dabei werden die Typdaten aus den Systemen des Herstellers entnommen, übertragen und beim Integrator wiederum in dessen Systeme geladen.

Abbildung 1: Gesamtszenario aus (2). Oben: Typinformationen; Unten: Instanzdaten



Quelle: ZVEI

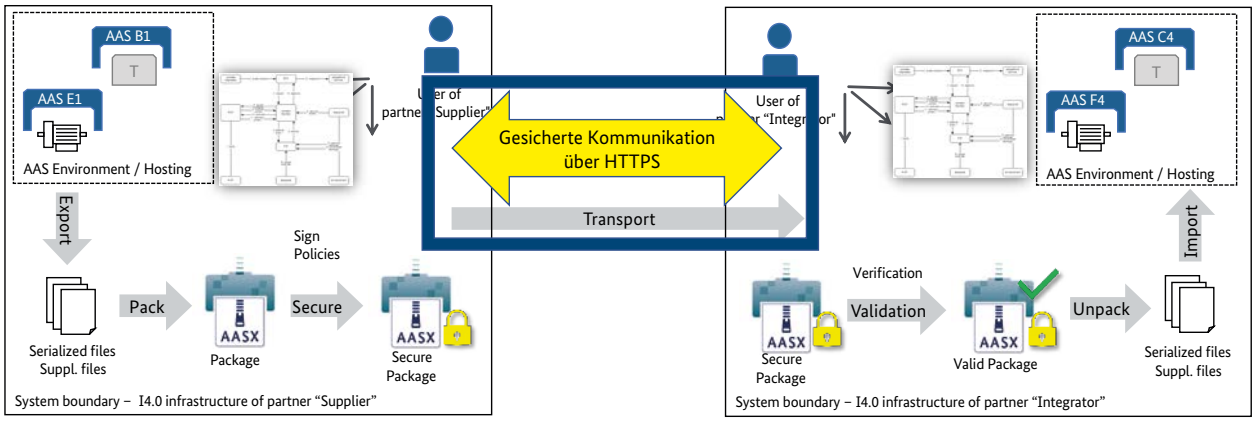
Abbildung 2: Übertragung von Typinformationen



Quelle: ZVEI

Abbildung 3 zeigt die in diesem Diskussionspapier betrachtete konkretisierte Aufgabenstellung, in der die Übertragung als Onlinevorgang realisiert ist. Der Integrator hat den Bedarf an Typinformationen und baut eine Onlineverbindung zum Hersteller (Supplier) auf. Hierzu wird davon ausgegangen, dass eine im Internetverkehr übliche HTTPS-Verbindung verwendet wird. Der Hersteller prüft die Identität des Anfragenden und stellt entsprechend seines Berechtigungskonzepts die serialisierten Typdaten in Form einer Datei zur Übertragung bereit. Aufseiten des Anfragenden wird die Echtheit der Datei geprüft und anschließend werden die Daten in die Systeme des Integrators geladen und stehen zur Benutzung bereit.

Abbildung 3: Schritte des Übertragungsprozesses

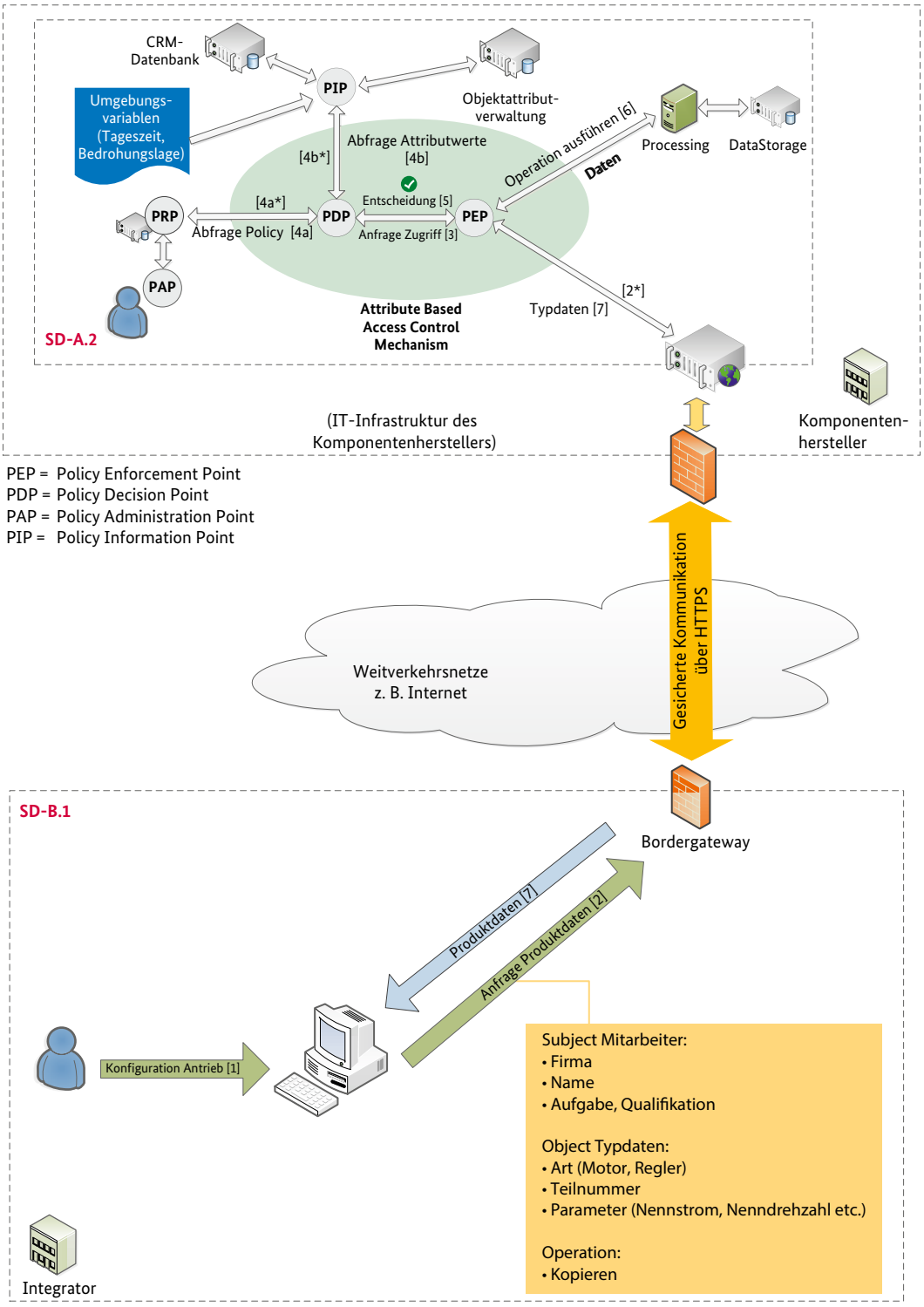


Quelle: Plattform Industrie 4.0

3. Annahmen und Anforderungen

Im Diskussionspapier „Sicherer Bezug von CAE-Daten“ (1) wurde bereits eine grundsätzliche Betrachtung des Übertragungsvorgangs durchgeführt, auf die im Folgenden zurückgegriffen werden soll, siehe auch Abbildung 4. Dargestellt wird der Bezug von einem Arbeitsplatz des Integrators aus, Informationsquellen sind die Systeme beim Komponentenhersteller. Die zu beziehenden Typdaten können dabei alles umfassen, was für die Beschreibung des Produkts und die Verwendung im Engineering benötigt wird. Dies können also technische Daten oder 3D-Modelle sein, aber auch Software und Firmware, die zum Beispiel für Simulationszwecke oder die Kommunikation mit dem Produkt (Treiber) benötigt werden könnten.

Abbildung 4: Realisierung des sicheren Downloads von Typinformationen



Quelle: Plattform Industrie 4.0

Für die Diskussion wird wie in „Sicherer Bezug von CAE-Daten“ (1) davon ausgegangen, dass die Authentizität der Daten im AASX-Format entsprechend dem verwendeten Standard „Open Packaging Convention“ (3) realisiert wird und nicht näher betrachtet werden muss. Die weitere Verarbeitung und die Sicherheit der Daten im empfangenenden System liegen ebenfalls außerhalb dieser Diskussion.

Die vorgestellte Lösung realisiert dabei einen Security-Ansatz, der von Applikationsebene (CAE-Software auf Arbeitsplatz) zu Applikationsebene (Systeme beim Komponentenhersteller) reicht.

Für einen praxistauglichen Einsatz ist es dabei wichtig, die realen Einsatzbedingungen zu berücksichtigen, die im unternehmensübergreifenden Einsatz auftreten. In Abbildung 4 sind die verschiedenen beteiligten Security-Domänen dargestellt. An den Grenzen der Security-Domänen sind typischerweise Security-Gateways installiert, die als Firewalls oder (TLS-)Proxys fungieren. Eine einheitliche Benutzerverwaltung zwischen beiden Domänen ist initial nicht anzunehmen.

3.1 Anforderungen

Moderne Designprinzipien sehen die Aufteilung der Aufgaben in jeweils eigene Dienste vor, um die unabhängige Entwicklung und Pflege zu ermöglichen: Service Oriented Architecture (SOA). Eine Ausprägung hiervon ist die Umsetzung durch miteinander agierende Microservices. Eine Lösungskonzeption muss eine entsprechende Architektur unabhängig von der tatsächlichen Umsetzung ermöglichen.

Im Folgenden werden die relevanten Dienste aus Sicht der aktuell in Er- und Überarbeitung befindlichen Dokumente zur „Verwaltungsschale im Detail“ beschrieben (2).

3.1.1 Bereitstellung der angefragten Daten

Für das Szenario der Bereitstellung von Engineeringdaten ist davon auszugehen, dass ein Komponentenhersteller über ein Portfolio von Angeboten verfügt und insofern mehr als nur die Daten eines einzelnen Produkts bereitstellen möchte. Der Downloadserver wird also ein größeres Angebot bereitstellen, das aber nicht jedem unbeschränkt zugänglich sein wird. Für die Bereitstellung von Daten zu einzelnen Produktinstanzen wird ebenso zu berücksichtigen sein, dass der Zugriff beschränkt sein wird. Auf dem Downloadserver, im Folgenden als Resource Server verallgemeinert, wird also eine Zugriffssteuerung umgesetzt sein müssen.

3.1.2 Autorisierung

Das Modell der Verwaltungsschale, wie in „Verwaltungsschale im Detail“ (2) und „Zugriffssteuerung für Industrie 4.0“ (4) beschrieben, unterstützt für jede Verwaltungsschale ein eigenes Rechtemodell. Die Autorisierung erfolgt dabei entsprechend der Attribute Based Access Control (ABAC) und benötigt für die Entscheidung über die Zugriffsrechte die Informationen (Attribute) über das Subjekt, das zugreifen möchte, sowie über die gewünschte Aktion und das Objekt, auf das zugegriffen werden soll. Die Informationen über das Subjekt werden während der Authentifizierung überprüft und bestätigt.

3.1.3 Authentifizierung

Die Authentifikation stellt die Attribute zur Beschreibung des Anfragenden (Subjekt) zur Verfügung, die die Autorisierung für die Entscheidung benötigt. Attribute könnten aus einer Benutzerverwaltung beim Komponentenhersteller stammen oder mit vom Anfragenden übertragen werden. Der Authentifizierungsserver steht für die Korrektheit der Attribute ein, nicht aber für deren Auslegung in der Zugriffssteuerung im Resource Server.

3.1.4 Skalierung

Für die Diskussion soll davon ausgegangen werden, dass der Downloadservice skalierbar ausgelegt sein soll, also Daten in großer Menge und großer Häufigkeit ausgeliefert werden müssen. Hierdurch ergeben sich die Anforderungen an die Architektur und eine Aufteilung der notwendigen Dienste.

Ein weiterer Aspekt der Skalierbarkeit ist die unternehmensübergreifende Zusammenarbeit, bei der z. B. die Nutzung filternder Proxys berücksichtigt werden muss. Es ist vorzusehen, dass Konsumenten (menschliche Benutzer oder Software-Applikationen und Systeme) aus vielen unterschiedlichen Unternehmen die Daten abrufen und dabei auch die einzelnen Konsumenten identifiziert und Zugriffsrechte basierend auf Eigenschaften (Attributen) zugeteilt werden müssen. Die Verwaltung der Eigenschaften sollte im Sinn der Skalierung dabei nicht auf der Seite des bereitstellenden Unternehmens erfolgen müssen.

4. Lösungsskizze/Diskussion

In der folgenden Lösungsskizze soll ein exemplarischer Lösungsansatz vorgestellt werden, der den eben beschriebenen Anforderungen gerecht wird. Die vorgeschlagenen Technologien, Schnittstellen und Dienste sollen die Machbarkeit zeigen und sind beispielhaft zu verstehen. In einer konkreten Umsetzung könnten die verschiedenen Dienste wie vorgeschlagen verteilt oder gemeinsam implementiert sein und andere interne Schnittstellen verwenden.

Die Lösungsskizze zielt dabei auch darauf ab, die Applikationslogik (hier: die betrachtete REST-API) und die Einbettung der Authentifizierung in die Kommunikation so zu trennen, dass beide Elemente unabhängig voneinander ausgestaltet und weiterentwickelt werden können. Im betrachteten Umfeld der Webservices stehen die notwendigen Mittel zur Verfügung, da Authentifizierung über die http-Header abgewickelt werden kann, während für die Nutzdaten die URI und der „Body“ verwendet werden.

4.1 Kommunikationsprotokoll HTTPS/REST

Wie schon im Diskussionspapier „Sicherer Bezug von CAE-Daten“ (1) beschrieben, besteht bei Verwendung des http(s)-Protokolls die größte Wahrscheinlichkeit, auch im unternehmensübergreifenden Kontext anwendbar zu sein. Dabei ist einzuplanen, dass die Kommunikation an Unternehmensgrenzen mittels TLS-Proxys gefiltert wird und das https-Protokoll daher die Fähigkeit verliert, Client-Zertifikate zur Authentifizierung zu verwenden. Dies liegt daran, dass im Protokoll die gegenseitige Authentifizierung genau dazu gedacht ist, Man-in-the-Middle-Angriffe (MitM-Angriffe) zu unterbinden und Integrität und Vertraulichkeit zwischen den beiden Endpunkten sicherzustellen. Ein inspizierender TLS-Proxy muss andererseits die Vertraulichkeit aufbrechen, um seine Aufgabe erfüllen zu können. Die Verwendung von https erlaubt es trotzdem, die Kommunikation außerhalb der Security-Domänen der beteiligten Unternehmen vor Manipulation und Mitlesen zu schützen, da durch die jeweiligen Proxys wieder TLS-Verbindungen aufgebaut werden und die Kommunikation außerhalb der Unternehmensgrenzen wieder geschützt ist. Dies schließt ausdrücklich die zur Authentifikation und Autorisation verwendeten Anmeldedaten und Token mit ein.

Schnittstellen für die automatisierte Kommunikation (Webservices) werden aufgrund der einfachen Struktur und klaren Modelle gerne mittels REST (Representional State Transfer) umgesetzt, das auf die http-Elemente GET, POST, PUT und DELETE setzt. REST ist proxyfreundlich. Eine REST-Schnittstelle für Verwaltungsschalen wird in Teil 2 der „Verwaltungsschale im Detail“ beschrieben. Sie erlaubt den Zugriff auf einzelne Datenelemente, die entweder Endpunkte, Teilmodelle oder ganze Verwaltungsschalen sein können.

Damit in der Unternehmens-IT übliche Proxys die Kommunikationsverbindung u. a. zur beschriebenen REST-Schnittstelle akzeptieren, müssen die Server-Zertifikate von einer Certification Authority (CA) ausgestellt sein, die auf dem Proxy als vertrauenswürdig hinterlegt ist. Im typischen Unternehmensbetrieb werden dies CAs aus dem Web-Umfeld sein.

4.1.1 Handshake zwischen den Parteien

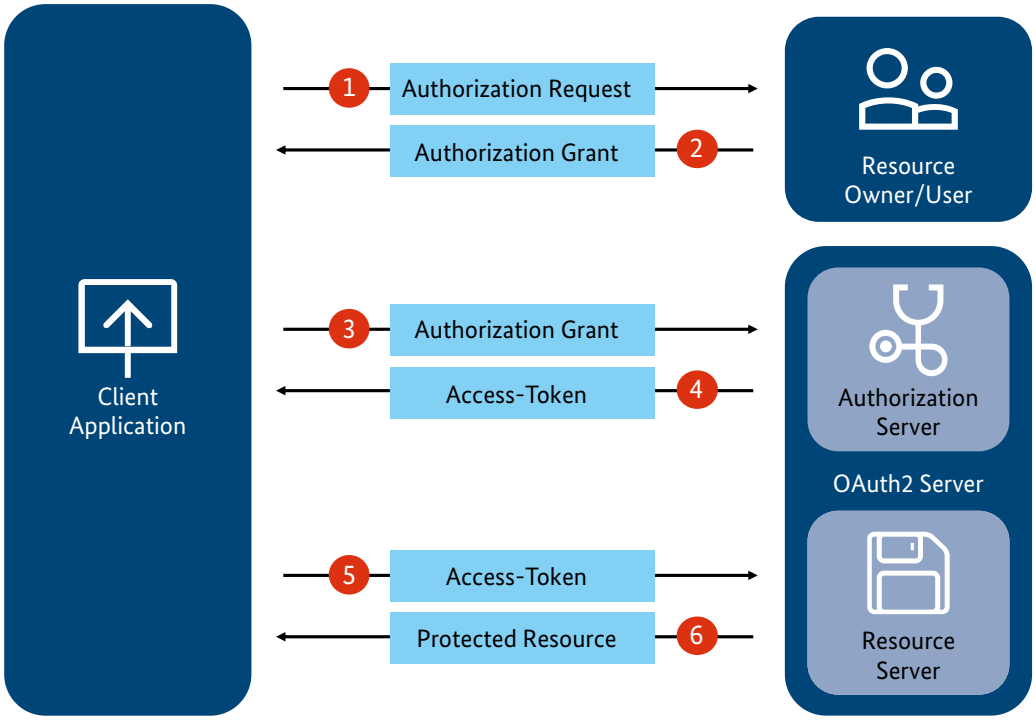
Ob die Zugriffsrechte auf die vom Server bereitgestellten Informationen für den Zugriff reichen, wird von der Zugriffssteuerung entschieden. Entsprechend den im Dokument zur Zugriffssteuerung (4) vorgesehenen Konzepten kann pro Verwaltungsschale unterschiedlich entschieden werden. Für das Protokoll ist daher vorzusehen, dass eine qualifizierte Rückmeldung an den Client gegeben werden kann. Diese Rückmeldung könnte zum Beispiel „Nur für angemeldete Benutzer“ oder „Nur für CAE-Systeme des Herstellers X“ sein, sodass der Client seine Anmeldung entsprechend aktualisieren kann. Inwieweit der Server diese qualifizierte Rückmeldung gibt oder die Anfrage ohne weitere Informationen abweist, um einem potenziellen Angreifer keine Hinweise zu geben, ist entsprechend der Security-Policy einzurichten.

4.2 Identitätsmanagement mittels OpenID Connect

Das Identitätsmanagement ist ein Thema, das im Web-Umfeld intensiv bearbeitet wird. Auf technischer Ebene hat sich der Informationstransfer mittels JSON Web Token (JWT) etabliert, der im Vergleich zu XML-Techniken kompakter abgewickelt werden kann. JSON Web Token werden dabei häufig in den HTML-Metadaten verwendet, zum Beispiel als Bearer Token. Mit JSON Web Token ist im Kontext dieses Dokuments nicht nur die Basistechnologie nach RFC 7519 „JSON Web Token (JWT)“ gemeint, sondern die gesamte Technologiefamilie wie z. B. auch RFC 7515 „JSON Web Signature (JWS)“ oder RFC 7517 „JSON Web Key (JWK)“ (5).

Ein wesentliches Konzept ist das „OpenAuth2.0 Authorization Framework“ nach RFC 8252, das die Interaktion zwischen einem Client (z. B. einer Applikation), einem „Resource Owner“ (Genehmiger) und den Ressourcen unter Einführung eines „Authorization Servers“ einführt, siehe Abbildung 5. Durch dieses Konzept kann der „Resource Server“ von der Aufgabe der Benutzerverwaltung entlastet und diese Aufgabe einem dedizierten Dienst übertragen werden.

Abbildung 5: Elemente des OAuth2 Authorization Frameworks



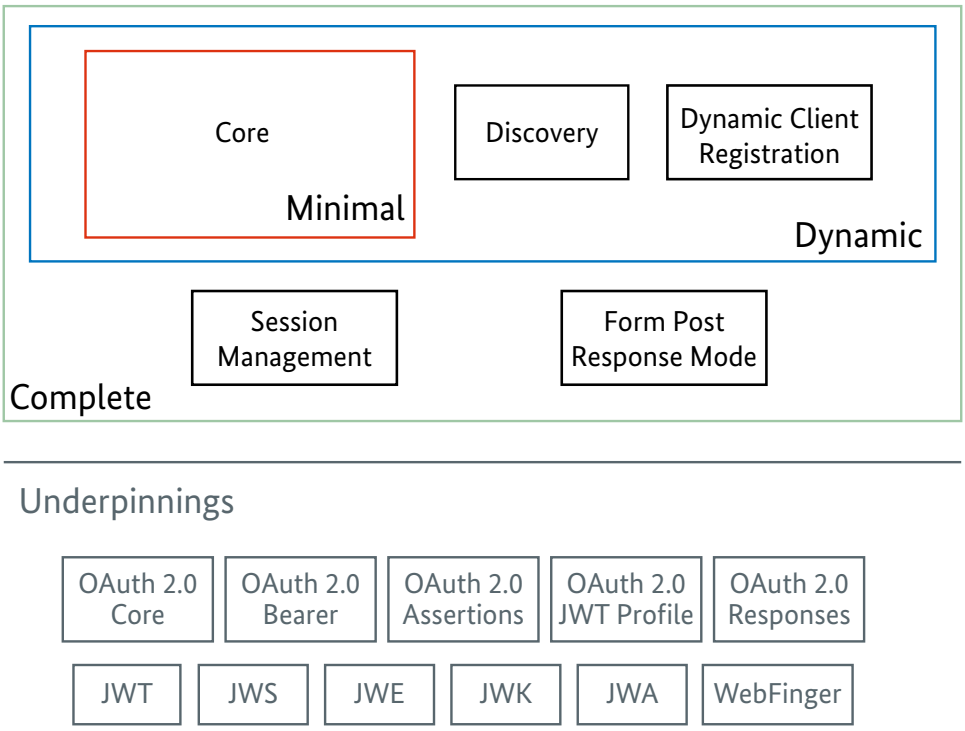
Quelle: Plattform Industrie 4.0

Wesentliche Elemente der OAuth2-Kommunikation setzen auf die vorher genannten JSON-Web-Techniken. Der Resource Server muss dazu dem Authorization Server vertrauen, wobei aus technischer Sicht eine digitale Signatur unter den erteilten Access Token im JWS-Format (JSON Web Signature) ausreichend ist.

Eine Beschränkung des OAuth2-Konzepts ist die Tatsache, dass der Authorization Server nicht nur die Authentifizierung vornimmt, wie in den Anforderungen diskutiert, sondern auch Rechte im „Access Token“ zuweist. Dies widerspricht aber den Konzepten der Zugriffssteuerung für Industrie 4.0 (4).

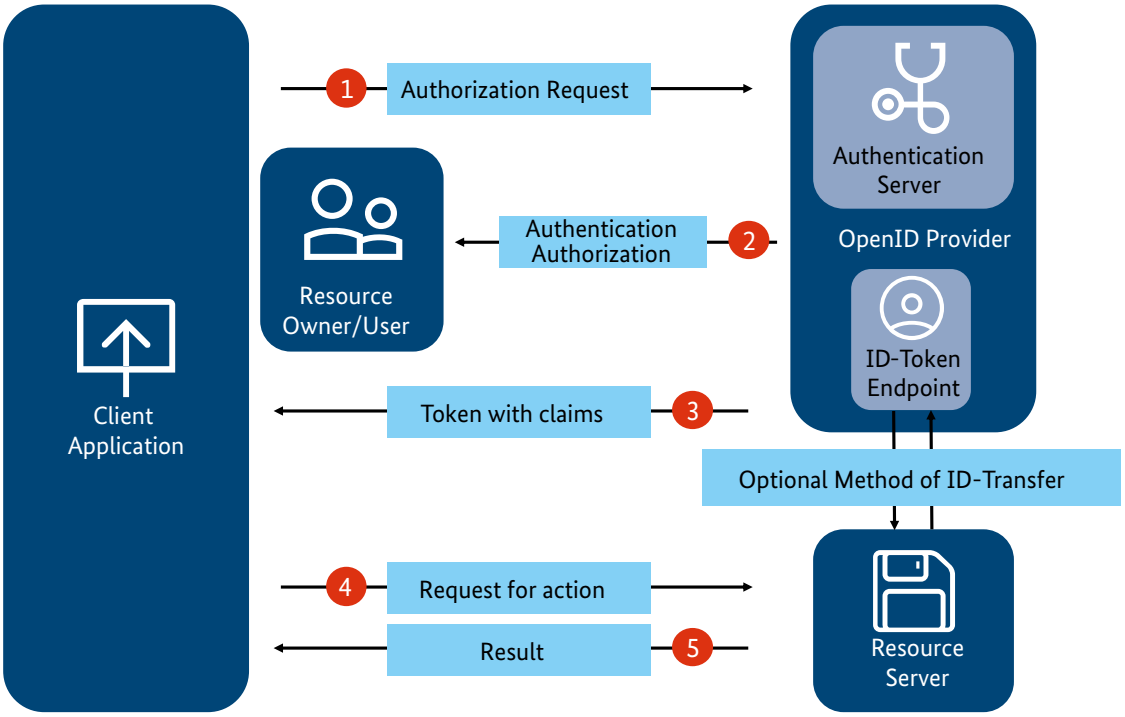
Diese Beschränkung wird im OpenID Connect-Framework, siehe Abbildung 6, aufgehoben. Das OpenID Connect-Framework baut technisch auf OAuth2 auf, trennt aber die Authentifizierung und das Rechtemanagement. Statt einer Rechtezuweisung im „Access Token“ werden zusätzliche „Identity Token“ eingeführt, die vom Client direkt verwendet werden oder alternativ etwa zur Reduktion des Bandbreitenbedarfs zwischen Resource Server und Authentifizierungsserver direkt ausgetauscht werden können, siehe Abbildung 7. Die Identity Token enthalten „Claims“ genannte Attribute über den Benutzer, wobei die in der Spezifikation beschriebenen Claims der Anwendung für Privatnutzer entsprechen und etwa Namen, Anschrift und Geburtsdatum beinhalten, nicht aber für Industrie 4.0 interessante Attribute wie eine Organisationszugehörigkeit oder Qualifikation. Die Spezifikation erlaubt jedoch eine domänenspezifische Erweiterung, die für eine Beschreibung der für Industrie 4.0 relevanten Attribute genutzt werden könnte. Vorschläge für solch zusätzliche Attribute sind allerdings nicht Teil dieses Dokuments.

Abbildung 6: OpenID Connect Protocol Suite (see: http://openid.net/connect)



Quelle: Plattform Industrie 4.0

Abbildung 7: Separater Authentication Service mit OpenID Connect



Quelle: Plattform Industrie 4.0

Die Verantwortung für die korrekte Authentifizierung des Clients und/oder weiterer beteiligter Parteien und die Bereitstellung der Claims liegt beim Authentifizierungsserver.

4.2.1 Autorisierung mittels ABAC

Basierend auf den Claims in den ID Token kann die Autorisierung für den Zugriff auf die Daten der Verwaltungsschalen wie in „Zugriffssteuerung für Industrie 4.0“ (4) und dem Dokument „Verwaltungsschale im Detail, Teil 1“ (2) beschrieben erfolgen. Die Implementierung der Zugriffssteuerung erfolgt dabei im Resource Server, der die übergebenen Attribute auswerten muss. Durch die Verwendung der Claims, die vom Authentisierungsserver bereitgestellt werden, ist die Umsetzung komplett von der Art der Authentifizierung entkoppelt und kann unabhängig weiterentwickelt werden.

4.2.2 Authentifizierung mit sicheren kryptografischen Methoden

Die meisten im Web verwendeten Modelle zur Authentifizierung richten sich an den menschlichen Benutzer und bieten daher meistens die Verwendung von Passworten an. Die davon ausgehenden Gefahren sind hinreichend diskutiert. Lösungskonzepte im Web basieren bspw. auf Zwei-Faktor-Authentifizierung (2FA), bei der zusätzlich zum normalen Login ein nur kurz gültiger Code per SMS, E-Mail o.Ä. verschickt wird[1], oder auf der Verwendung kryptografischer Schlüssel. Typischerweise sind die kryptografischen Schlüssel mit Zertifikaten gekoppelt, die bei Aufbau der sicheren Verbindung, z. B. im TLS-Handshake zur gegenseitigen Authentifizierung, verwendet werden. Bei Verbindungen über inspizierende TLS-Proxys ist ein entsprechender Verbindungsaufbau aber nicht möglich.

Die Verwendung von kryptografischen Schlüsseln zur Authentifizierung für Web-Anwendungen ist im WebAuthn-Standard der W3C beschrieben (6). Die Methode zielt auf die Anwendung in Webbrowsern und die Verwendung kryptografischer Schlüssel (ohne Zertifikate und damit ohne Informationen zur Identität) für eine zentrale Benutzerverwaltung und ist insofern für die vorliegende Anwendung nicht direkt geeignet.

Die Verwendung von WebAuthn zur Authentifizierung im OpenID Connect-Framework wird in verschiedenen Beiträgen betrachtet, unter anderem in einem Entwurf zu einer Erweiterung von OpenID Connect (7). Eine beachtenswerte Eigenschaft von WebAuthn ist es, dass es die Attestierung eines sicheren Schlüsselspeichers unterstützt.

4.2.3 Gemeinsame Verständigung zu Vertrauensankern

Die Verwendung elektronischer, digitaler Zertifikate hängt von einem gemeinsamen Verständnis und Vertrauen zu den dahinterliegenden Public-Key-Infrastrukturen (PKI) ab. Im vorliegenden Anwendungsszenario soll sich der Anfragende mit X.509-Zertifikaten authentifizieren. Entsprechend müssen die Zertifikate der ausstellenden CAs auf dem Authentifizierungsserver als vertrauenswürdig anerkannt und hinterlegt sein. Hierbei könnten öffentliche, akkreditierte CAs mit bekannten Security-Qualitäten als Aussteller infrage kommen. In Unternehmen werden für die Authentifizierung von Mitarbeitern und Rechnern üblicherweise aber keine Zertifikate öffentlicher CAs verwendet, sondern Zertifikate unternehmenseigener CAs – dies hat auch Kostengründe.

Für das vorliegende Anwendungsszenario ist daher ein vorgeschalteter Prozess vorzusehen, in dem die betreffenden Vertrauensanker ausgetauscht werden, damit später im Tagesgeschäft darauf zurückgegriffen werden kann.

4.2.3.1 Bewertung des Vertrauens in die fremde Root-CA

Für die Bewertung der Vertrauenswürdigkeit kommen zurzeit manuelle Prozesse zum Einsatz, die beim Aufbau der Geschäftsbeziehung ablaufen. Für die Bewertung werden die Policys der CA herangezogen, die in Dokumenten wie dem Certification Practice Statement (CPS) beschrieben sind. Im Dokument ETSI TS 102 042 „Electronic Signatures and Infrastructures (ESI); Policy requirements for certification authorities issuing public key certificates“ sind grundlegende Policys beschrieben, auf die referenziert werden könnte. Die European Bridge CA[2], die vom TeleTrusT organisiert wird, beschreibt ebenfalls einen gemeinsamen Mindeststandard. Durch die eIDAS-Verordnung „Electronic IDentification, Authentication and trust Services“[3] wird innerhalb der Europäischen Union ein Rechtsrahmen für die gegenseitige Anerkennung von Trust-Service-Providern für die Erzeugung digitaler Signaturen und Siegel geschaffen. Im Dokument „Vertrauensinfrastrukturen im Kontext von Industrie 4.0“ (8) werden die Möglichkeiten beschrieben, eIDAS als gemeinsamen Vertrauensanker zu nutzen. Im Dokument „IIOT Value Chain Security – The Role of Trustworthiness” (9), das die Plattform Industrie 4.0 gemeinsam mit der Robot-Revolution-Initiative erarbeitet hat, wird ein Modell beschrieben, bei dem der Aufbau einer Geschäfts-

1 Weitere Methoden ergeben sich z. B. durch besondere Apps („Authenticator“), die auf mobilen Geräten als unabhängiger Kanal fungieren.

2 https://www.ebca.de

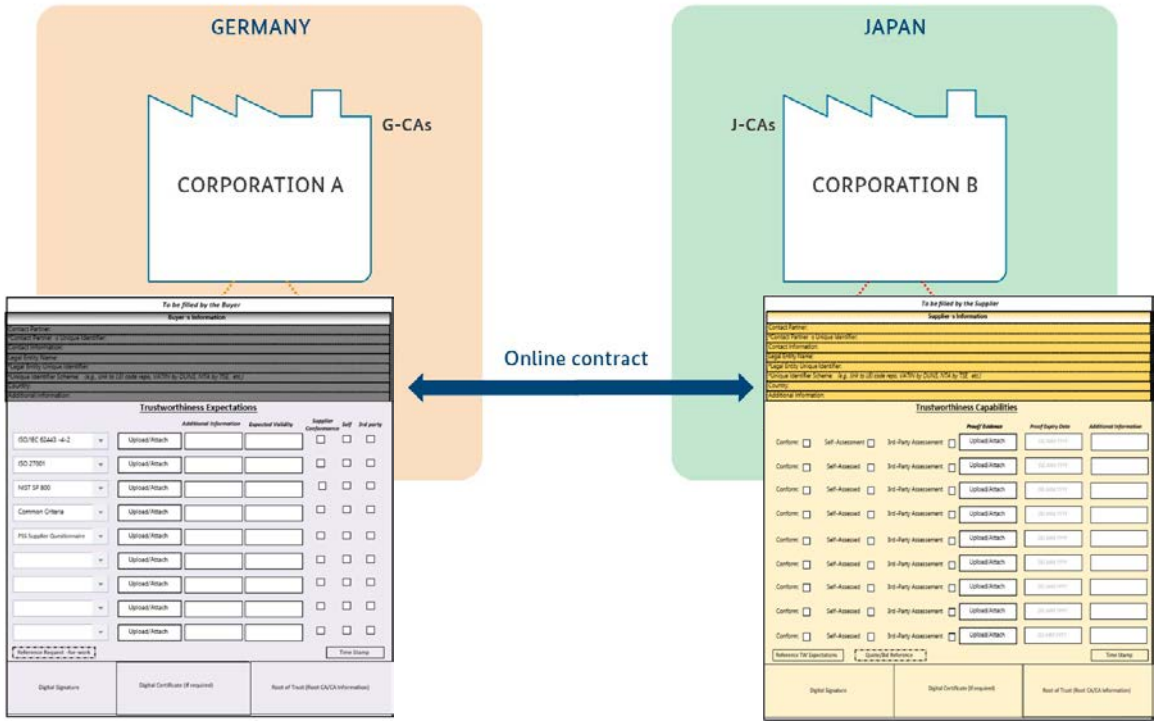
3 Referenz: EU-Verordnung Nr. 910/2014 des Europäischen Parlaments und des Rates vom 23. Juli 2014, https://eur-lex.europa.eu/legal-content/DE/TXT/PDF/?uri=CELEX:32014R0910

beziehung auf Basis eines vertrauenswürdigen Austauschs von Sicherheitsanforderungen und Sicherheitsnachweisen erfolgen würde, siehe Abbildung 8. In diesen Prozess könnte auch die Bereitstellung der Vertrauensanker integriert werden.

4.2.3.2 Sichere Bereitstellung von CA-Zertifikaten

Besteht zwischen den Parteien Einigkeit über die Security-Policys, können die Vertrauensanker ausgetauscht werden. Dabei können für die Partner auch jeweils mehrere CA-Zertifikate verwendet werden. Dies ist notwendig, weil zum Beispiel zum Zeitpunkt eines Roll-overs mehrere Zertifikate gleichzeitig unterstützt werden müssen. Auch kann es sein, dass für verschiedene Einsatzbereiche wie Mitarbeiter und Systeme im eigenen Betrieb oder für Produkte unterschiedliche CAs im Einsatz sind. Um sicherzustellen, dass die Vertrauensanker selbst authentisch sind, könnten sie durch das bereitstellende Unternehmen bestätigt werden, indem sie zum Beispiel mit einem auf das Unternehmen ausgestellten Siegel beglaubigt werden. Eine gemeinsame Vertrauensbasis für diese Siegel wäre zum Beispiel die Vertrauensinfrastruktur entsprechend der eIDAS-Verordnung.

Abbildung 8: Trustworthiness-Austauschmodell in Anlehnung an (9)



Quelle: Plattform Industrie 4.0

5. Authentifizierung mit Token

OpenID Connect macht grundsätzlich keine Vorgaben dazu, wie die Authentifizierung am Authentifizierungsserver stattfindet. Vielmehr beschreibt OpenID Connect, wie die Information über den Anfragenden vom Authentifizierungsserver der vertrauenden Instanz, hier dem Resource Server, bereitgestellt wird. Die Authentifizierung selbst verwendet unter anderem die Mechanismen, die für OAuth2 spezifiziert sind. Beispiele und Präsentationen gehen normalerweise von einer zentralen Benutzerdatenbank beim Authentifizierungsserver aus.

Für die Anwendung im skalierbaren Umfeld der Industrie 4.0 ist eine derartige zentrale Benutzerdatenbank nicht flexibel genug. Die vorliegenden Modelle sollen deshalb so erweitert werden, dass sie die Verwendung von digitalen Zertifikaten, gegebenenfalls ergänzt um weitere Attribute, z. B. durch Attributzertifikate, erlauben.

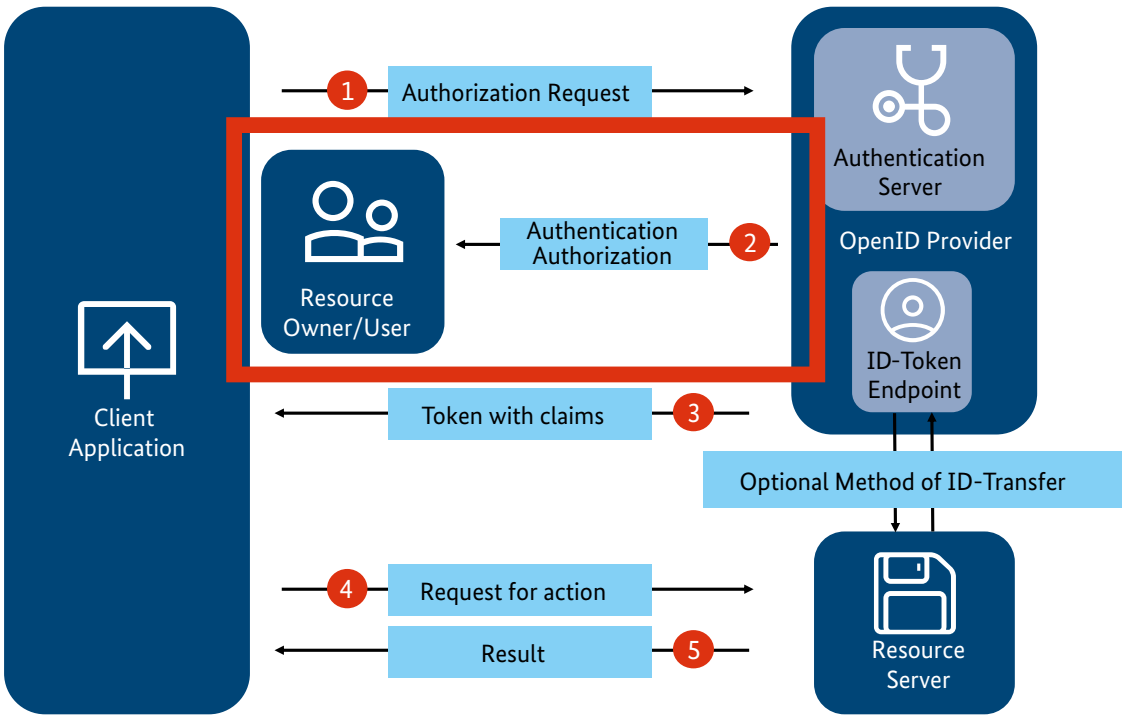
Die folgende Diskussion fokussiert sich auf den Prozessschritt der Authentifikation, siehe Abbildung 9.

5.1 Verwendung von Zertifikaten zur gegenseitigen Authentifizierung in TLS

„OAuth 2.0 Mutual-TLS Client Authentication and Certificate-Bound Access Tokens" (RFC 8705) setzt auf TLS-Verbindungen die nicht durch einen inspizierenden TLS-Proxy unterbrochen werden, siehe Abbildung 10. „Client-Zertifikat" bezeichnet im Kontext des TLS-Protokolls das digitale Zertifikat des Anfragenden. In der Terminologie von OpenID Connect wäre dies aber nicht der Client (anfragende Applikation), sondern der Resource Owner. Sehr häufig werden dabei die Client-Zertifikate von der Client-Applikation bereitgestellt. Die zugrunde liegende Verwendung von Client-Zertifikaten ist im TLS-Protokoll (RFC 8446 für TLS 1.3) enthalten. Möchte der Server Client-Zertifikate erhalten, schickt er eine „Certificate Request"-Nachricht an den Client. Eine Liste der akzeptablen Aussteller von Client-Zertifikaten kann der Server in der optionalen „Certificate Authorities"-Struktur übertragen. Der Client kann nun aus seinen zur Verfügung stehenden Zertifikaten ein Client-Zertifikat auswählen und ggf. mit seiner Zertifikatskette an den Server übertragen. Im Kontext eines menschlichen Benutzers würde dem Anwender (Resource Owner) eine Liste der anwendbaren Zertifikate vom Webbrowser zur Auswahl angezeigt werden. Bei einem automatisierten Vorgehen muss die Applikation das passende Zertifikat auswählen. Der Server kann nun entscheiden, ob er das Client-Zertifikat für die Anmeldung akzeptiert und den Verbindungsaufbau entsprechend fortsetzt oder abbricht. Die Zuverlässigkeit des Prozesses nimmt ab, wenn die Liste der akzeptablen Aussteller (Truststore) fehlt oder sonst fehlerhaft ist.

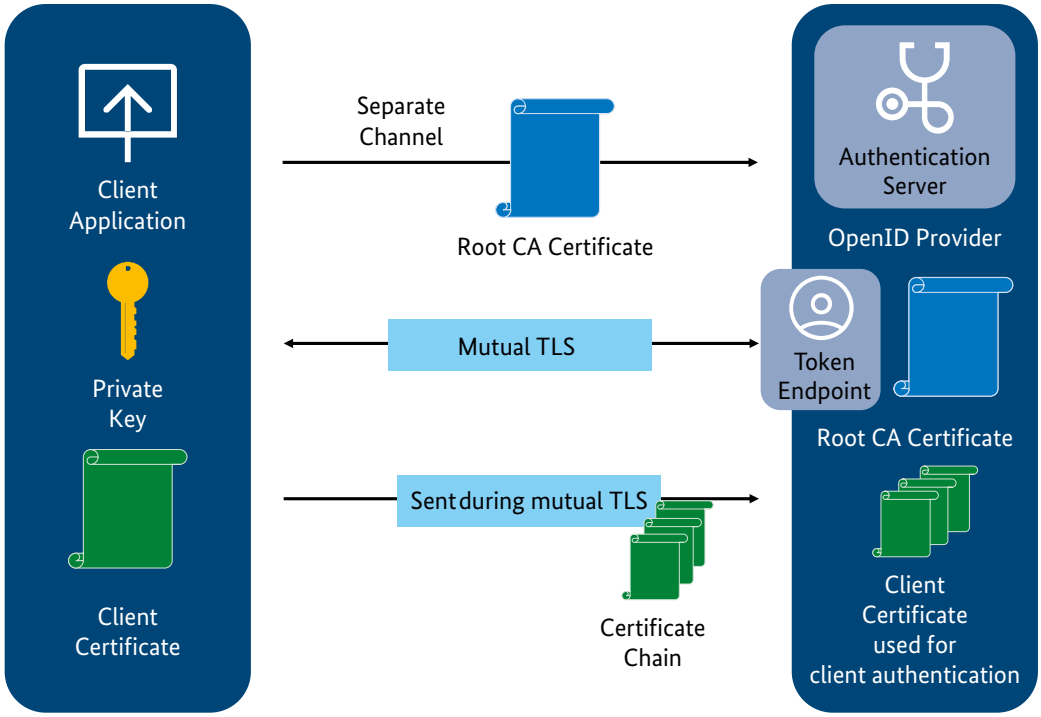
Im Fall der Verwendung von Client-Zertifikaten findet die Authentifikation entsprechend Schritt 2 in Abbildung 9 im TLS-Handshake des HTTPS-Protokolls statt, siehe Abbildung 10.

Abbildung 9: Prozessschritt der Authentifikation



Quelle: Plattform Industrie 4.0

Abbildung 10: Mutual-TLS Client Authentication nach RFC 8705



Quelle: Plattform Industrie 4.0

Es sei hier bemerkt, dass die gegenseitige Authentifizierung mit Zertifikaten in diesem Fall zwischen dem Client und dem Authentifizierungsserver stattfindet. Für den eigentlichen Datenaustausch mit dem Resource Server findet die Authentifizierung des Servers in einer separaten TLS-Verbindung zwischen Client und Resource Server statt.

5.2 Verwendung von Token für die Authentifizierung mit kryptografischen Schlüsseln

Um eine entsprechende Authentifizierung mit kryptografischen Schlüsseln auch bei einer durch inspizierenden TLS-Proxy unterbrochenen TLS-Verbindung durchführen zu können, muss die Authentifizierung von der TLS-Ebene auf die Applikationsebene übertragen werden. Hierzu verwenden OpenID Connect/OAuth2 JSON Web Token (JWT, RFC 7519) mit den entsprechenden Erweiterungen wie JSON Web Signature (JWS, RFC 7515).

5.2.1 Kryptografisch sichere Authentifizierung mit der private_key_jwt-Methode

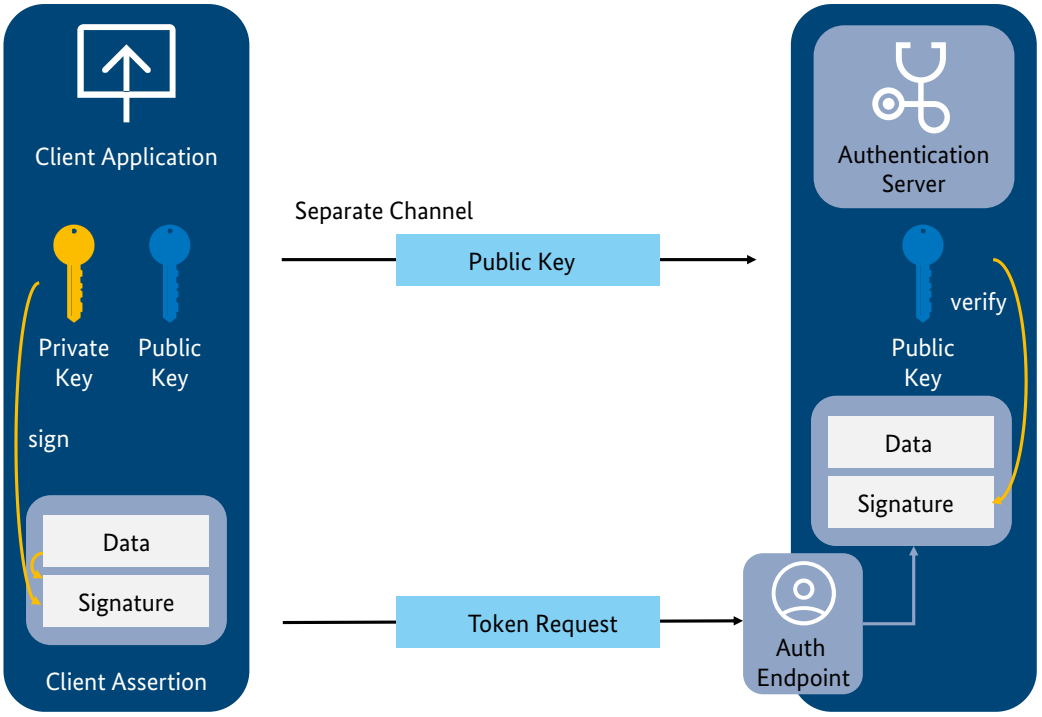
Abbildung 11 zeigt, wie die in OpenID Connect Core spezifizierte Authentifizierung einer Client-Applikation mit der private_key_jwt-Methode funktioniert. Die Client-Applikation signiert die Daten im Token mit dem vorliegenden privaten Schlüssel (Private Key); der Server kann die Authentizität des Anfragenden nun prüfen, indem er die Signatur gegen den öffentlichen Schlüssel (Public Key) des Anfragenden prüft.

Für diese Prüfung muss der Server den öffentlichen Schlüssel (Public Key) des Anfragenden verfügbar haben. Dies ist jedoch im vorliegenden Kontext unvorteilhaft, da:

- die öffentlichen Schlüssel auf dem Server verwaltet werden müssen (welcher Anfragende verbirgt sich hinter dem Schlüssel?),
- die öffentlichen Schlüssel dem Server in einem sicheren Verfahren verfügbar gemacht werden müssen und
- die Information dem Server bei Erneuerung der Schlüssel erneut sicher bekannt gemacht werden muss.

Die genannten Probleme lassen sich durch die Verwendung des Konzepts der Public Key Infrastructure (PKI) lösen. Dabei wird der öffentliche Schlüssel (Public Key) zusammen mit Attributen zur Beschreibung der Identität in ein Zertifikat eingebettet und so werden Schüssel und Identität miteinander verbunden (Verwaltung der Identitäten). Durch digitale Signatur einer vertrauenswürdigen Stelle wird das Zertifikat beglaubigt (Schlüsselverwaltung). Diese Vorteile wären bei Verwendung direkter TLS-Authentifizierung, wie in Abschnitt 5.1 beschrieben, anwendbar.

Abbildung 11: Authentifizierung mit private_key_ jwt

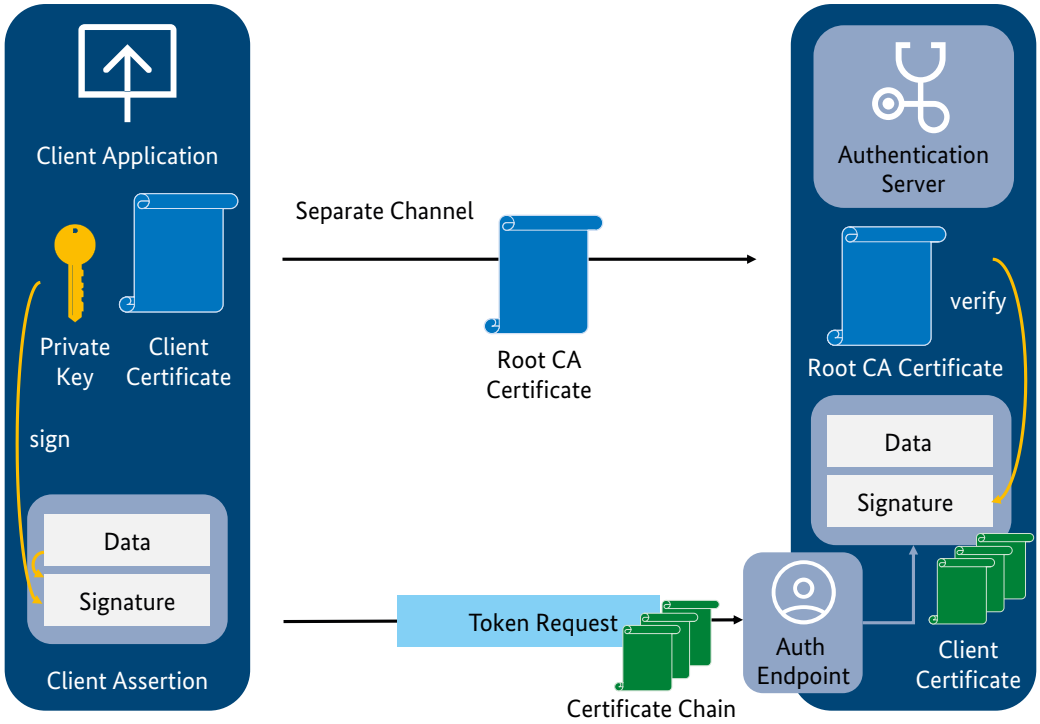


Quelle: Plattform Industrie 4.0

5.2.2 Vorgeschlagene Methode private_key_certchain_jwt

Um die Vorteile der PKI auch mit JSON Web Token nutzen zu können, wird hier eine neue Methode „private_key_ certchain_jwt“ vorgeschlagen. Abbildung 12 zeigt dieses Konzept, das eine Erweiterung von private_key_jwt durch eine Zertifikatskette ist. Zusätzlich zu den zu übertragenden Daten wird das Client-Zertifikat mit der ergänzenden Zertifikatskette im JSON Web Token übertragen. Für Zertifikat und Kette wird die „x5c“-Datenstruktur verwendet, wie in RFC 7515 (JSON Web Signature) beschrieben. Der Server kann nun das Client-Zertifikat gegen seine Liste akzeptabler Aussteller prüfen und basierend auf den Attributen im Zertifikat die weitere Authentifizierung durchführen. Eine beispielhafte Implementierung ist in Abschnitt 11 beschrieben.

Abbildung 12: Methode private_key_certchain_jwt



Quelle: Plattform Industrie 4.0

Die Umsetzung weiterer Security-Anforderungen, wie zum Beispiel der Schutz vor Replay-Attacken, entspricht der Anwendung der „private_key_jwt“-Methode. Hier wäre zu prüfen, ob die notwendigen Sicherheitsmaßnahmen bereits berücksichtigt sind.

Die Vertrauenswürdigkeit der Claims beruht auf dem Vertrauen, das in die Prüfung der Signatur und der Zertifikatskette gesetzt wird. Insofern sind die Prozesse zur Akzeptanz und Pflege der verwendeten Root-CA-Zertifikate ein wichtiger Bestandteil des Security-Konzepts. Weiterhin ist die Sicherheit des privaten Schlüssels für die Signatur des Token Requests eine Basis der Vertrauenswürdigkeit. Insofern sollte die Bestätigung, dass der private Schlüssel etwa durch ein Sicherheitselement geschützt ist, durch die Prozesskette unterstützt werden, etwa durch Attestation oder Attribute in den Zertifikaten. Die sichere Speicherung von Schlüsseln könnte als Claim in den ID Token abgebildet werden.

6. Beispielhafter Ablauf einer Download-Sitzung

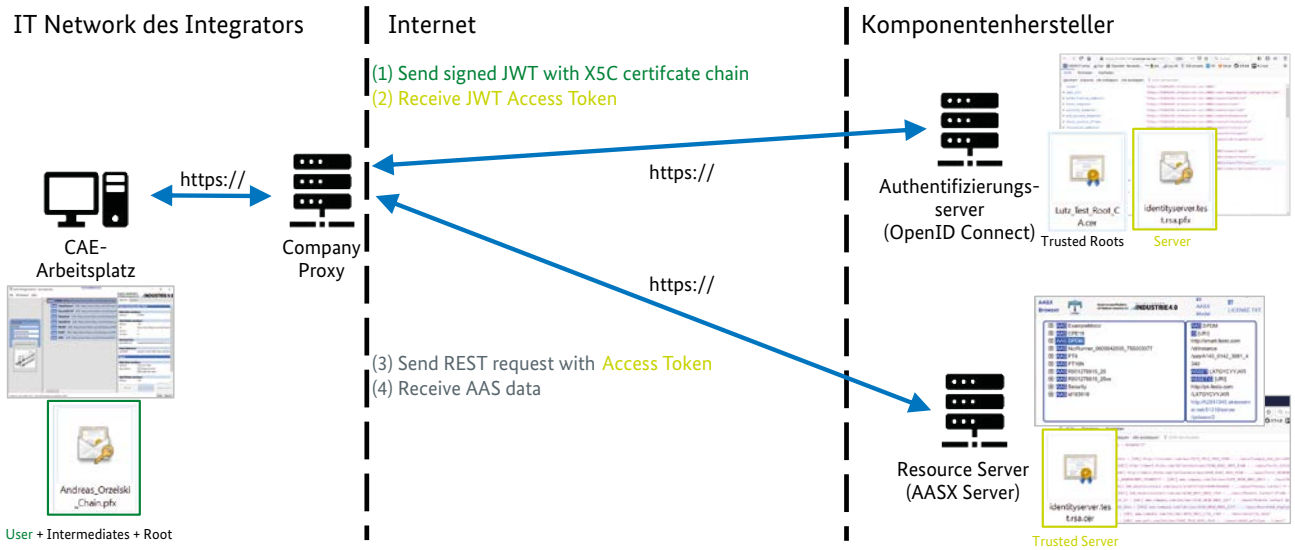
Für das oben vorgeschlagene Konzept wurde ein Demonstrator, wie in Abbildung 13 dargestellt, basierend auf dem AASX-Explorer (10) implementiert. Dafür wurde ein als Open Source zur Verfügung stehender OpenID Connect Server als Authentifizierungsserver um die vorgeschlagene „private_key_certchain_jwt"-Methode erweitert, die entsprechend in den AASX-Explorer implementiert wurde. Der AASX-Server als Resource Server muss dem Authentifizierungsserver vertrauen und die nach der Authentifizierung ausgestellten Token mit den entsprechenden Attributen (Claims) auswerten. Die Attribute werden dann nach den Regeln des ABAC, wie in „Verwaltungsschale im Detail" (2) beschrieben, für die Zugriffssteuerung verwendet.

6.1 Handshake für den automatischen Zugriff

Sind die Endpunkte und notwendigen Attribute zum Zeitpunkt des Verbindungsaufbaus bereits bekannt, weil sie vorab festgelegt wurden, kann der Abruf der gewünschten Daten wie in Abbildung 13 beschrieben erfolgen. Viel wahrscheinlicher ist es jedoch, dass zwar der Endpunkt für den Zugriff auf die Daten auf dem Resource Server bekannt ist, der Endpunkt für die Authentifizierung aber zur Laufzeit ausgehandelt wird, da er mit der eigentlichen Anwendung nicht verknüpft ist.

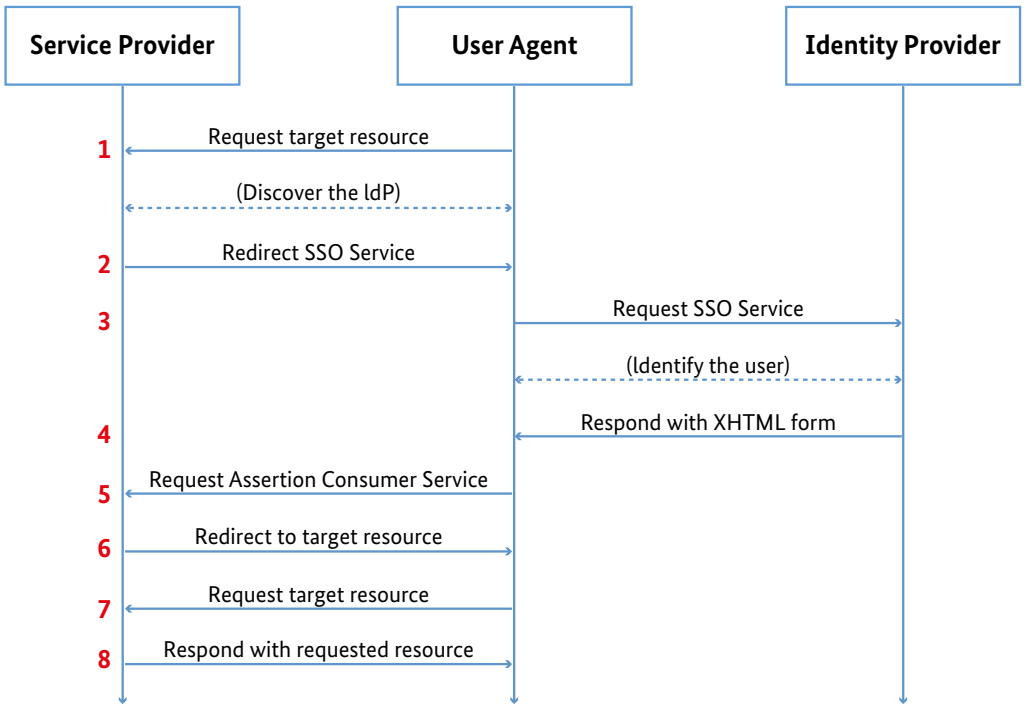
Im Kontext der Security Assertion Markup Language 2.0 (SAML 2.0) sind entsprechende Mechanismen vorhanden, um eine entsprechende Interaktion zu realisieren, siehe Abbildung 14. Die Übertragbarkeit auf die Verwendung von OpenID Connect ist zu prüfen. Hierbei muss der User Agent in der Lage sein, mit den verwendeten Redirect-Nachrichten umzugehen.

Abbildung 13: Implementierter Demonstrator



Quelle: Plattform Industrie 4.0

Abbildung 14: Interaktion im Kontext SAML 2.0[4]



Quelle: Plattform Industrie 4.0

Im Folgenden wird diskutiert, wie die generischen http-Mechanismen eingesetzt werden können, um die Verwendung einer möglichst großen Auswahl an Bibliotheken für REST über http zu ermöglichen.

6.1.1 Verweis auf den Authentifizierungsserver

Bei Webdiensten, insbesondere für menschliche Benutzer, ist es gängige Praxis, vor dem Zugriff auf nicht öffentliche Ressourcen auf Authentifizierungsserver zu verweisen. Hier ist eine Schwäche im http-Protokoll vorhanden, da dieses keinen Redirect für Authentifizierungszwecke vorsieht. Die Antwort „401 Authentication Required“ kann laut Standard nicht mit einem „Location“-Header kombiniert werden, da im http-Standard selbst ursprünglich nur „Basic Authentication“ vorgesehen war. Für Redirects sind im http-Protokoll die Antwortcodes der Klasse „300“ vorgesehen. Für den hier vorgesehenen Zweck, bei dem der Verweis auf den Authentifizierungsserver nur für die Authentifizierung gelten soll, wäre die Anwendung einer entsprechenden Erweiterung des http-Standards[5] die richtige Antwort. Mit dem Redirect wird im „Location“-Header die URL des Authentifizierungsservers inklusive der notwendigen Hinweise für die Authentifizierung und den Rückverweis übermittelt.

Ein beispielhafter Location-Header könnte wie folgt aussehen:

```
Location: https://auth.server/authenticate?some_
info=info&redirect_uri=https://resource.server/
resource&some_more_info=more
```

Dabei wird die Information dem Client zurückgegeben, wie er den Authentifizierungsserver erreichen kann, wobei in der URI auch alle Informationen für den Authentifizierungsserver enthalten sind.

6.1.2 Verhalten des Clients

Der Client muss nun auf den Redirect geeignet reagieren. Für den Zugriff auf den Authentifizierungsserver muss der Client die eigentlich geplante Anfrage zurückstellen und auf den Authentifizierungsserver zugreifen. Durch die Übertragung aller Informationen im Location-Header erhält der Authentifizierungsserver die notwendigen Zusatzinformationen, sodass er mit dem Client alle Attribute/Eigenschaften (Claims) aushandeln kann.

4 CC BY-SA 3.0, https://en.wikipedia.org/w/index.php?curid=32521419

5 https://tools.ietf.org/html/draft-williams-http-accept-auth-and-redirect-02

Nachdem die Authentifizierung erfolgreich abgeschlossen wurde, übernimmt der Client die vom Authentifizierungsserver bereitgestellten JSON Web Token (Access Token) als Bearer Token für den anschließenden Zugriff auf den Resource Server.

6.1.3 Verhalten des Authentifizierungsservers

Der Authentifizierungsserver erhält über die URI die Hilfsinformationen, die für den Erfolg des Handshakes notwendig sind. Dies könnte zum Beispiel die Menge der bereitgestellten Attribute (Claims) sein, die für den Zugriff auf die gewünschte Ressource notwendig sind. Sind diese Informationen nicht vorhanden oder nicht durch die Implementierung des Authentifizierungsservers unterstützt, liefert dieser die verfügbaren Standardattribute zurück.

Optional kann der Authentifizierungsserver auf die Bereitstellung des ID Tokens für den Client verzichten und die ID Token nur dem Resource Server über den ID-Token-Endpunkt bereitstellen. Dies könnte z. B. die notwendige Bandbreite reduzieren oder gegenüber dem Client verbergen, welche Claims dem Resource Server bereitgestellt werden.

6.1.4 Verhalten des Resource Servers

Nach der erfolgreichen Authentifizierung greift der Client mit dem erhaltenen Bearer Token auf den Resource Server zu. Im Resource Server erfolgt die Zugriffssteuerung basierend auf den Claims, die im ID Token enthalten und vom Authentifizierungsserver beglaubigt wurden. Gegebenenfalls muss der Resource Server das ID Token vom entsprechenden ID-Token-Endpunkt des Authentifizierungsservers beziehen.

6.1.5 Demonstrator

Abschnitt 11 beschreibt die technischen Details des umgesetzten Demonstrators. Mit den im Lösungskonzept beschriebenen Elementen soll er als Diskussionsbasis für eine Weiterentwicklung des Konzepts und betroffener Standards dienen sowie zu Implementierungen zur produktiven Nutzung führen.

7. Zusammenfassung und Ausblick

Im vorliegenden Diskussionspapier wird ein Konzept für einen sicheren Downloadservice beschrieben, der die Anforderungen von Industrie 4.0 berücksichtigt. Die Kommunikation findet online statt und ist unternehmensübergreifend. Für die Skalierbarkeit und Sicherheit wird das Konzept kryptographischer Schlüssel und Zertifikate verwendet. Gleichzeitig werden die Anforderungen an sichere Unternehmensprozesse erfüllt, die die Kontrolle über die ein- und ausgehende Kommunikation erfordern und zum Einsatz inspizierender Proxys an den Unternehmensgrenzen führen. Insofern wird die Authentifizierung in die Applikationsschicht verlagert und mittels JSON Web Token umgesetzt. Dabei kommen soweit möglich Industriestandards wie OAuth2 und OpenID Connect zum Einsatz. Für die Verwendung von Client-Zertifikaten wird eine neue Authentifizierungs-Methode „private_key_certchain_jwt“ vorgeschlagen und in Demonstratoren erprobt. Die Konzepte orientieren sich am Anwendungsszenario des Downloads von Engineeringdaten, indem sie IT-Techniken wie HTTPS und REST voraussetzen und auf die Skalierbarkeit „nach oben“ zielen. Die Konzepte sind jedoch nicht auf dieses Anwendungsszenario beschränkt.

Basierend auf den Erkenntnissen dieses Diskussionspapiers und der Demonstratoren sollte eine Ausentwicklung und Standardisierung erfolgen. Hierzu ist Kontakt zur OpenID Foundation bezüglich der technischen Standardisierung aufzunehmen. Die vorgeschlagenen Konzepte können nur dann erfolgreich angewendet werden, wenn sie in der Breite von den Anbietern entsprechender Technologien unterstützt und von den Anwendern akzeptiert werden.

8. Glossar

ABAC	Attribute Based Access Control
Authentifizierung	bezeichnet den Nachweis oder die Überprüfung der Authentizität
Authentifizierungsserver	Dedizierter Dienst/Endpunkt für die Authentifizierung
Autorisierung	Berechtigung erteilen, basiert auf Authentifzierung
Bearer Token	(Überbringer-)Token, dessen Inhalt ohne weitere Nachprüfung vertraut wird.
CAE	Computer Aided Engineering
Claim	(Beanspruchtes) Identitätsattribut, z. B. Name, Adresse, …
Client/Client Application	Anfragende Applikation
HTTPS	HyperText Transfer Protocol (S: secured via TLS)
Identity Provider	Dienst, der Identitäten prüft und ID-Informationen bereitstellt
ID Token	Token mit Attributen, die die Identität des Subjekts beschreiben
JWT, JWS	JSON Web Token (RFC 7519), JSON Web Signature (RFC 7515)
OAUTH2	Protokoll zur delegierten Autorisierung (RFC 6749)
OpenID Connect	Konzept/Protokoll zur delegierten Authentifizierung
Resource Owner	Genehmiger, im Webumfeld kann dies der User sein, der auf seine eigenen, auf einem Resource Server liegenden Daten zugreift und seiner Applikation dies erlaubt
Resource Server	Server, der relevante Ressourcen bereitstellt
SD	Sicherheitsdomäne
SOA	Service Oriented Architechture
TLS	Transportation Layer Security

9. Abbildungsverzeichnis

10. Literaturverzeichnis

1. **Diskussionspapier „Sicherer Bezug von CAE-Daten“. Berlin:** Plattform Industrie 4.0, 2018.

2. **Details of the Asset Administration Shell. Part 1: The exchange of information between partners in the value chain of Industrie 4.0 (v2.0.1). Berlin:** Federal Ministry for Economic Affairs and Energy (BMWi), 2020.

3. **Information technology – Document description and processing languages – Office Open XML File Formats – Part 2: Open Packaging Conventions.** ISO/IEC 29500-2:2012.

4. **Diskussionspapier „Zugriffssteuerung für Industrie 4.0-Komponenten zur Anwendung von Herstellern, Betreibern und Integratoren“. Berlin:** Plattform Industrie 4.0, 2018.

5. **Peyrott, Sebastián E. The JWT Handbook.:** Auth0 Inc., 2016–2018.

6. **Web Authentication: An API for accessing Public Key Credentials Level 1.:** W3C, 2019.

7. **OpenID Connect Extended Authentication Profile (EAP) ACR Values 1.0 – draft 00.:** OpenID Foundation, 2016.

8. **Vertrauensinfrastrukturen im Kontext von Industrie 4.0.:** Plattform Industrie 4.0, 2020 (in Vorbereitung).

9. **IIoT Value Chain Security – The Role of Trustworthiness.:** Plattform Industrie 4.0, 2020.

10. **aasx-package-explorer.** https://github.com/admin-shell-io/aasx-package-explorer.

11. Technische Details zum vorgeschlagenen Lösungskonzept

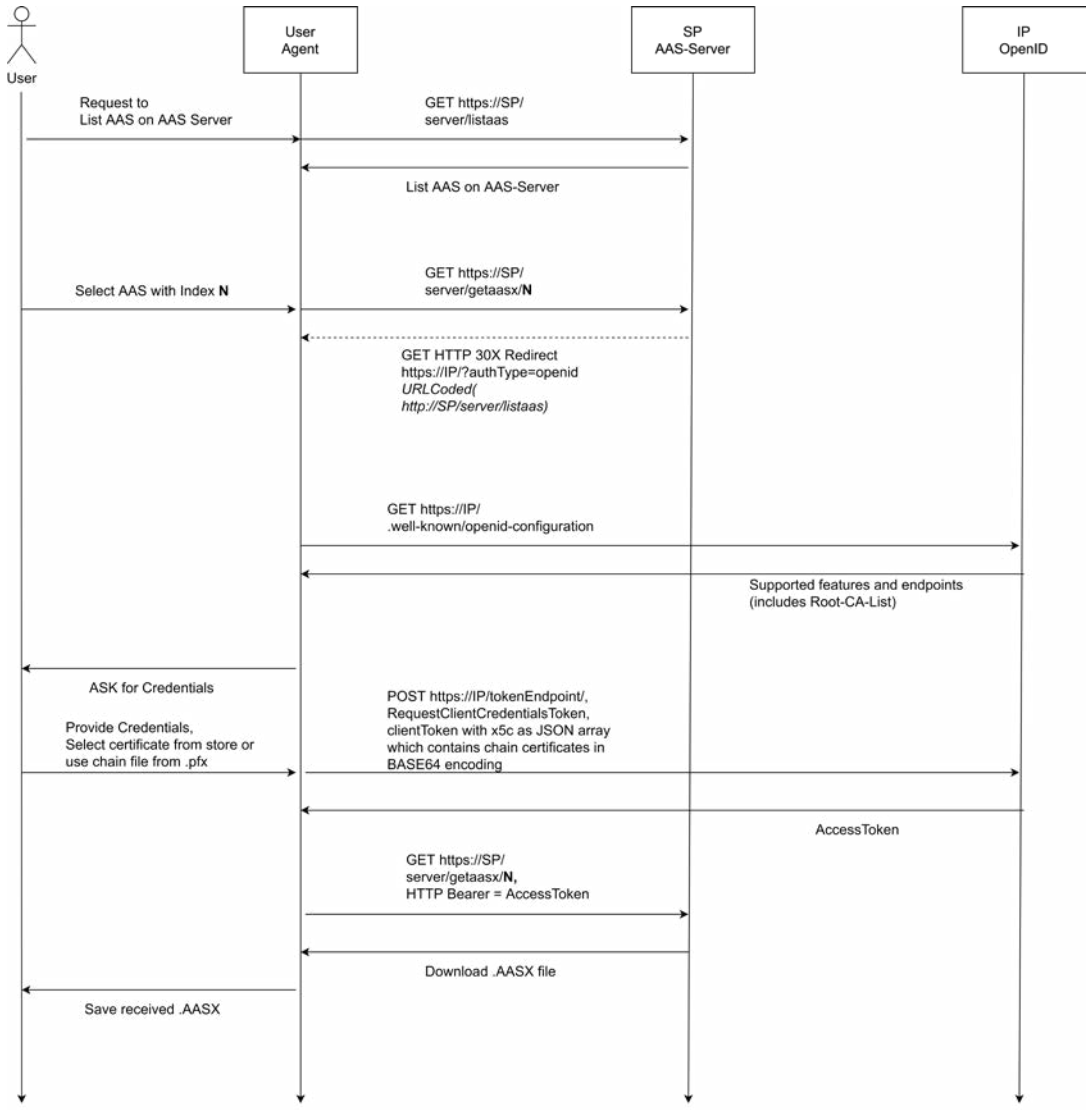
Im Folgenden sollen die oben beschriebenen Konzepte anhand eines Demonstrators dargestellt werden. Informationen zum Demonstrator und Quellcode können unter einer OpenSource-Lizenz unter https://github.com/admin-shell-io bezogen werden.

11.1 Beispielhafter Ablauf der Authentifizierung und Autorisierung

Abbildung 15 zeigt einen beispielhaften Ablauf eines Zugriffs, wie im Demonstrator umgesetzt. Für dieses Beispiel wird angenommen, dass der User Agent, z. B. ein AASX-Explorer, für den Benutzer zuerst eine Liste der verfügbaren Verwaltungsschalen (Asset Administration Shells, AAS) anfragt. Diese Liste würde im vorliegenden Beispiel keiner Zugriffsbeschränkung unterliegen und daher auch ohne Authentifizierung bereitgestellt werden.

Anschließend soll die (AASX)-Datei der Verwaltungsschale N bezogen werden. Da der User Agent zu diesem Zeitpunkt entweder kein Token hat oder das Token abgelaufen oder unzureichend ist, verweigert der AAS-Server den Zugriff und leitet den AAS-Client mittels eines http-Redirect zum Identity Provider IP um. Dabei liefert der AAS-Server die Information mit, dass der Identity Provider nach OpenID

Abbildung 15: Implementierter Demonstrator



Quelle: Plattform Industrie 4.0

Connect arbeitet. Es könnten weitere Informationen mitgegeben werden, die der Identity Provider benötigt, um den Authentifizierungsprozess durchzuführen. Im Beispiel ist dies nur die URL des angesprochenen Endpunkts des Identity Providers.

Der Zugriff des User Agents auf den Identity Provider mit den ergänzenden Informationen erlaubt diesem, ein entsprechendes Authentifizierungsangebot an den User Agent zusammenzustellen, das die unterstützten Funktionen und Endpunkte auflistet. Entsprechend des OpenID-Connect-Konzepts könnten dabei verschiedene Authentifizierungsmechanismen unterstützt werden, die im Handshake ausgehandelt werden müssten. Für den vorgeschlagenen Fall des „private_key_certchain_jwt“ listet der Identity Provider die akzeptierten CAs auf, indem er deren Subject Distinguished Names (DN) analog zur Implementierung von Mutual-TLS überträgt. Der User Agent kann nun den User zur Eingabe oder Freigabe von Credentials auffordern. Im vorliegenden Beispiel wäre dies die Freigabe einer Zertifikatskette für die vorgeschlagene Methode „private_key_certchain_jwt“ und die Erlaubnis, den zugehörigen privaten Schlüssel für die Signierung des Tokens zu verwenden. Der User Agent schickt daraufhin, wie im Lösungskonzept vorgesehen, seine Authentifizierungsdaten in Form eines „private_key_certchain_jwt“-Tokens an den Identity Provider. Dieser prüft die Korrektheit des Tokens inklusive der enthaltenen Zertifikatskette.

Der Identity Provider stellt nun dem User Agent ein Access Token aus, dass dieser als Bearer Token für den Zugriff auf die gewünschte Ressource verwenden kann. Hierzu wiederholt der User Agent den Zugriff, der nun erfolgreich mit dem Herunterladen der AASX-Datei abgeschlossen werden kann. Der User Agent kann das Token auch für weitere Zugriffe auf die gleiche API benutzen, ohne sich erneut authentifizieren zu müssen, bis entweder das Token abgelaufen ist oder die Rechte für eine bestimmte Operation zusätzliche Attribute entsprechend den Vorgaben der Zugriffssteuerung erfordern.

11.2 Format eines „private_key_certchain_jwt“-Tokens

Für die Umsetzung des vorgeschlagenen Konzepts ist das „private_key_certchain_jwt“-Token entsprechend Abschnitt 5.2.2 zu definieren. Das Token basiert auf dem „private_key_jwt“-Token, das um die notwendigen X.509-Zertifikate ergänzt wird. Da die Verwendung von Zertifikatshierarchien unterschiedlicher Beteiligter unterstützt werden soll, wird ein x5c-Element mit der vollständigen Zertifikatskette vorgesehen. Die Zertifikatskette wird entsprechend dem Konzept eines JSON Web Key (JWK) mit in die Header eingebaut, wo auch die anderen Elemente wie Algorithmus und Key Type integriert sind, siehe Tabelle 1.

Tabelle 1: Header des „private_key_certchain_jwt“ Tokens[6]

Parameter	Description
alg	Algorithm used. For digitally signed JWT It should be hashed and signed like RSA signed with 256bit Hash “RS256” or EC signed with 256bit Hash “ES256”
kty	Key Type, here “RSA” or “EC”
use	Intended use of the key, here Signature (of the token) “sig”
x5c	**[REQUIRED] Additional Element in “private_key_certchain_jwt” Array containing X.509 certificate chain in BASE64 encoded format, list starting with the certificate used to sign and ending with the root CA certificate**
[other parameters]	Not significant for this discussion

6 Darstellung nach https://kb.authlete.com/en/s/oauth-and-openid-connect/a/client-auth-private-key-jwt

In der „client_assertion“, die zusammen mit dem Header digital signiert sein muss, sind die weiteren Parameter bezüglich der Kommunikationspartner enthalten, siehe Tabelle 2.

Tabelle 2: Payload des „private_key_certchain_jwt“

Parameter	Description
iss	[REQUIRED] Issuer. This must contain the client_id of the OAuth client.
sub	[REQUIRED] Subject. This must contain the client_id of the OAuth client.
aud	[REQUIRED] Audience. A value that identifies the authorization server as an intended audience. The authorization server must verify that it is an intended audience for the token. The audience should be the URL of the authorization server's token endpoint.
jti	[REQUIRED] JWT ID. A unique identifier for the token, which can be used to prevent reuse of the token. These tokens must only be used once unless conditions for reuse were negotiated between the parties; any such negotiation is beyond the scope of this specification.
exp	[REQUIRED] Expiration time on or after which the JWT must not be accepted for processing.
iat	[OPTIONAL] Time at which the JWT was issued.
user	[OPTIONAL] Claim to be confirmed by Identity Provider (may also be learned from certificate chain included).

Schließlich wird die Signatur entsprechend der folgenden Formel berechnet:

```
Signature=
  RSASHA256(base64UrlEncode(header) + "." +base64UrlEncode(payload), PrivateKey)
```

11.2.1 Beispieltoken

Im Folgenden wird ein Beispieltoken mit den Elementen
Header und Payload dargestellt:

```
{
  "alg": "RS256",
  "kid": "5A136803068C7CCAF232DCE3DB6FD8ADAFCE59C7",
  "x5t": "WhNoAwaMfMryMtzj22_Yra_OWcc",
  "typ": "JWT",
  "x5c": [
    "MIIDPjCCAiagAwIBAgIIMUVO//pDBPIwDQYJKoZIhvcNAQELBQAwTz
     …
     DYYKKseV80HOj",
    "MIIDkjCCAnqgAwIBAgIICmtpapy3fswwDQYJKoZIhvcNAQELBQAwTz
     …
     OkkOplB6G0DuAo147RTlLr7pxgagjiZ/+o3cAvz3iIuLz"
  ]
}
{
  "jti": "d77d180a-d6ff-4c53-bd49-ad5825a16c49",
  "sub": "client.jwt",
  "iat": 1586241438,
  "email": "aorzelski@phoenixcontact.com",
  "nbf": 1586241438,
  "exp": 1586241498,
  "iss": "client.jwt",
  "aud": "https://localhost:5001/connect/token"
}
{Digitale Signatur}
```

Das vollständige Token ist im Folgenden abgedruckt und kann per Copy & Paste z. B. bei https://jwt.io eingegeben und analysiert werden. Für die Analyse der eingebetteten Zertifikate kann z. B. einer der online verfügbaren X.509-Zertifikats-Decoder oder das „openssl x509"-Kommandozeilentool verwendet werden. Die drei Bestandteile **Header**, **Payload** und **Signatur** sind durch „." getrennt und farblich abgehoben.


```
eyJhbGciOiJSUzI1NiIsImtpZCI6IkY3RUMwMTlCNDI5OTM4NTkzNzA3MUJGM0VFOUY1RTcwRDQ3RUEwRTAiLCJ4NXQiOiI5LXdCbTBLWk9GazNCeHZ6N3A5ZWNOU
i1vT0EiLCJ0eXAiOiJKV1QiLCJ4NWMiOlsiTUlJRGZEQ0NBbVNnQXdJQkFnSUlUUTNiQkRnaGNxWXdEUVlKS29aSWh2Y05BUUVMQlFBd1R6RUxNQWtHQTFVRUJoTU
NSRVV4R0RBV0JnTlZCQW9URDFCb2IyVnVhWGdnUTI5dWRHRmpkREVOTUFzR0ExVUVDeE1FUTFSV1F6RVhNQlVHQTFVRUF4TU9VM1ZpSUVOQklFeGxkbVZzSURJd0h
oY05NakF3T1RJeElURXhOakF3V2hjTklqRXdPVEl4TVRFeE5qQXdXakNCaURFTE1Ba0dBMVVFQmhNQ1JFVXhKakFrQmdOVkJBb01IVkJvYjJWdWFYZ2dRMjllZEdG
amRDQkhiV0pJSUNZZ1EyOHVJRXRITVNRd0lnWURWUVFEREJ0TWRYUjZJRXJEcEc1cFkydGxJQ2gyYjI0Z1ZHVnpkQ0JEUVNreEt6QXBCZ2txaGtpRzl3MEJDUUVXS
Ed4cVlXVnVhV05yWlVCd2FHOWxibWw0WTI5dWRHRmpkQzVqYjIwd2dnRWlNQTBHQ1NxR1NJYjNEUUVCQVFVQUE0SUJEd0F3Z2dFS0FvSUJBUUN2ZUUzTUpFQ1grS1
FiWE9SQVl5eEtDTkNtKzRRTi9RaW1DOFFqUWZGV2gxVHI1TUdxaHE3TGdDVE9wVW9NV1JSbnlhUmVZQll0aUxOUVZEV256V3FWNWtYM0tvc2g5SkZ5M2NrNDlLUHM
zRTNBVzk2VWZBYnhWNzIyakRpS1k5UTJ1U1JWdlB0SERyd2pSK3pOQms5T2hueE1WQURHdE1nLzFqTWxFY2pxOVZlODBkOWJTa2Vwd3hKZGd0R0hkbzJoMlNyWWRv
V3FaYXJYbGVHT2VydGZ5OG1wMEF5QzVJMjh1aThMdFdHRmFkN0Myc3pJRWwzL0V0c1BlaWdTd0RJSFFORjJjSWNndkoyd0pWTktXdTlyL0JRZnRBVEdYMEJuK1BNT
is5Q0xvWTBHcUVrS1F6amNKdkpzbG9oWm5kbTh4eXdZQ2dwM3pEYk9GclNHb3ZFTkFnTUJBQUdqSWpBZO1CNEdDV0NHU0FHRytFSUJEUVFSRmc5NFkyRWdZMlZ5ZE
dsbWFXTmhkR1V3RFFZSktvWklodmNOQVFFTEJRQURnZ0VCQUNxeWF6bStqZ01Ob0ZnK05nNjhXM3FRbHFUbm1MRXFBdDZTWHlsVkUvOWQ3M1VZeEg3aHh6UGJzR3N
zV2NaclV5WG50T0FMYVRyQ3RnVS9BWGNTNCtRd0xaYk1DZFlqNWprS25QYTJlSmxtS203RHNSVkFlV3ZoU1hWSFJMT0Z2Nm9qY0JRQnVOYWhxdHdrL052RmlLeHN5
YTYwZWxtYnFGeHh0WitQMWlHYzJkTnF4aWUrM08wM2tMUEEyYk1qQ2lPM1RnRlpBYVVIdTNYS1g2eHJJZHpPYWhVaERmbjZ4aDJjREhwUzJZSzFjdTg2U3VkNmhLW
mxpZkxnWG9xTkw3TE1sMDBsczZ2UHpCNjRzNDBVcERscmVOMTltWHZKc1QwWUJHeGFPN1R0ZS9uZWE0VG9xZmR1M211dENBVjAxQ25kdmQyTWtUWE9PWmpMLzIvRz
dRZz0iLCJNSUlEa2pDQ0FucWdBd0lCQWdJSUNtdHBhcHkzZnN3d0RRWUpLb1pJaHZjTkFRRUxCUUF3VHpFTE1Ba0dBMVVFQmhNQ1JFVXhHREFXQmdOVkJBb1REMUJ
vYjJWdWFYZ2dRMjl1ZEdGamRERU5NQXNHQTFVRUN4TUVRMVJXUXpFWE1CVUdBMVVFQXhNT1UzVmlJRU5CSUV4bGRtVnNJREV3SGhjTk1qQXdNakUzTVRjMU1EQXdX
aGNOTWprd09ESXpNVFV6TWpBd1dqQlBNUXN3Q1FZRFZRUUdFd0pFUlFWU1CWUdBMVVFQXhNT1UzVmlJRU5CSUV4bGRtVnNJREV3SGhjTk1qQXdNakUzTVRjMU1EQXdX
aGNOTWprd09ESXpNVFV6TWpBd1dqQlBNUXN3Q1FZRFZRUUdFd0pFUlFWU1CWUdBMVVFQ2hNUFVHaHZaVzVwZUNCRGIyNTBZV04wTVEwd0N3WURWUVFMRXdSRFZGW
kRNUmN3RlFZRFZRUURFdzVUZFdJZ1EwRWdUR1YyWld3Z01qQ0NBU0l3RFFZSktvWklodmNOQVFFQkJRQURnZ0VQQURDQ0FRb0NnZ0VCQUw3NE5ENUVWUWJIZ0doRW
NIOGg1Q09CN25BYTRITS9NNjBPa3dFNC9ybzhtckRPRW1YaloyVm82TFMzREtuaTRnTE10VC95aTh3bHhIVE8wOEt5UTR6WXYyTHpYSHEwbHkreW1uMzZTYWF6bVV
MenRKRloxY2Y2c2UrQW5WWVh2cTNhcHI2RzQvU05jNTdzOUZYVFhVSHFheVZUWmY1SEZPMDI3MzV5ZzF5MWxxbllVOUFPa3JzUGF1WG0rSnhQYnRTdVV0R3pRSDlk
c0Z4MHlvcmVkYnJmV283Qkp0NkVCd1pPRHFHekdVME96ZW04WTR1RlZoRmVRL1diTi9udDg0Z1RMNjhCSEdZb0NpT3hOOThXc3dBNm8yeG9LTkJWLzRvYVF3clpMc
DRWTm1lZ2VRenVsY0EremtTNDlHR1ZKT0ZESldnK1BIUDlQdzA1WFNHNUwxTUNBd0VBQWFOeU1IQXdEd1lEVlIwVEFRSC9CQVV3QXdFQi96QWRCZ05WSFE0RUZnUV
VFZEhLUy9rVmZ0ZWMrSlZqQ2d1UFhLcVF6b2d3Q3dZRFZSMFBCQVFEQWdFR01CRUdDV0NHU0FHRytFSUJBUVFFQXdJQUJ6QWVCZ2xnaGtnQmh2aENBUTBFRVJZUGV
HTmhJR05sY25ScFptGpZWFJsTUEwR0NTcUdTSWIzRFFFQkN3VUFBNElCQVFCMHU3VGJZQXlaekQydlpxTEdoeVgxelQweXQvTkxkallHZURQckskvcWE4K1RPdytv
OWdXTGVVeXF4bFRXb3JLU0RJZGVHVG9SQXNpY1Fyek9sTmRCc0ZpeXRhRkpwdm9jTDBOQUc5RWV5WEw1c1JSVnE5dk95N2MySmpxWVpuNG14ZU5rZ3QxeU8xeVJKQ
21JaDNkcXU5WVE0UFRYQlZxdmt2cUVHTnc2cWl6MmdPcXBGaUZWeU9zNWRHSC9ZcjR4UVVuSFM0ci9qd3FQTjZjZ0NUd3Iza3JYWHZVZXhSUWVzVHYrSkxxZ0dWen
VRQjNhcjg3WWpnclFncFF6cFp0KzRXM1pkZlJiek5hTXRwV1hsV2w4ZFA0TTVsRHNVWDJDQ2tFUG5tOXE1UlhKdmV3bFZtUm9aYXQ0cHlHY0gyV3AxRGxkYmVSdFZ
UZVJEbVVydlRRbnEiLCJNSUlEZERDQ0FseWdBd0lCQWdJSU1jZmdneG02N0Fvd0RRWUpLb1pJaHZjTkFRRUxCUUF3TVRFVE1CRUdBMVVFQ2hNS1VHeGhlV2R5YjNW
dVpERWFNQmdHQTFVRUF4TVJUSFYwZWlCVVpYTjBJRkp2YjNRZ1EwRXdIaGNOTWpBd01qRTNNVGMwT1RBd1doY05Namt3T0RJek1UVXpNakF3V2pCUE1Rc3dDUVlEV
lFRR0V3SkVSVEVZTUJZR0ExVUVDaE1QVUdodlpXNXBlQ0JEYjI1MFlXTjBNUTB3Q3dZRFZRUUxFd1JEVkZaRE1SY3dGUVlEVlFRREV3NVRkV0lnUTBFZ1RHVjJaV3
dnTVRDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBTHZpUTY5dDdxMVQyNDhsZ01sdnZsbHhmQTBKOHRkb3d2SUdxT3h2L0lJeEpGTFJ
zcDlYcjBncEwreFZOS0VSTVZGZnJVQ3FHdHpUbU1aYWFPb1NSalB6b2xmSHdPMGJ3aXM2dTIzWnZrZlpDRlFJU2RFbGFBbUE5Y2tUdGhUSVRjbS9LY045U1p4cmpv
RElTQlJVQVMvelNpdmlOTXp4ZXppSDBQell0STdWRkpFRVBHdExOWGdWRlp3UUhTMGlHUXVFSVQ2cGxnS09GcVd3eGdlRGQvR29KYlIrdXkzNjR5MkZhZWVXNW9nQ
1pBRk55M1paYStkUFI1TGpPOENDVTBtZ3JYUXRmVjJRRVV1cjR6V21KYVc2M3dNWm42ZXNvclowT1gxQTVOK1lwN2NpYnpPR1pEbmxGOThad3ZoM1dBTVptVDY1TV
hOZ29QQmZkL3ZvQ3NVQ0F3RUFBYU55TUhBd0R3WURWUjBUQVFIL0JBVXdBd0VCL3pBZEJnTlZIUTRFRmdRVVlzMmczM2RNbExwb1ZxN0lGMW92TFNTS0ZGWXdDd1l
EVlIwUEJBUURBZ0VHTUJFR0NXQ0dTQUdHK0VJQkFRUUVBd01BQnpBZUJnbGdoa2dCaHZoQ0FRMEVFUllQZUdOaElHTmxjblJwWm1sallYUmxNQTBHQ1NxR1NJYjNE
UUVCQ3dVQUE0SUJBUUNnbFp0VE5USys3S3JwZThFem0xUW9zcWNlaURRR3l1YnJLZlNjSlkzSklsZFhwTm0zNWQzaWE4akJSL0lnZVZoOTRzRUdDZmt1SEZOcEVKT
3ZWQ2p0Q2hSb2UyVi9DdE5ydzB6MDZDVGpRNnFPZlAwb3JHcGUrQ3JkM09sbzFtOFY2RVBUZ1FWazRSUzdSdjd0b3NjaVQwNjlGanA3TmdIa2dQdzBNWFBnTFpBRn
BhZUFsOTY3L1MrTlpLNjhIb2J5N3ZWRGtrZ1FoaEhaYlY0Rmc3OTNnWXdRK1MxT3J1eEQ4SUJ1ZHRaRmpDSElGKzFESFlzWFVuR3BrWldJODFGMXRmOUVHWVpOWkh
NWkJYaVBCa3p0K0VoTzZzb3RNWUJYdWx6VDBIR29yUDVFSXpHOWRMa0JFaGdMcUhDUnlnQldYeFloVDdCWEo3Um5NVWQwV0hOYSIsIk1JSURWakNDQWo2Z0F3SUJB
Z0lJUHN6c1NTS09ZcUl3RFFZSktvWklodmNOQVFFTEJRQXdNVEVUTUJFR0ExVUVDaE1LVUd4aGVXZHliM1ZlWkRFYU1CZ0dBMVVFQXhNUlRIVjBlaUJVWlhOMElGS
nZiM1FnUTBFd0hoY05VGt3T0RJek1UVXpNakF3V2hjTk1qa3dPREl6TVRVek1qQXdXakF4TVJNd0VRWURWUVFLRXdwUWJHRjVaM0p2ZFc1a01Sb3dHQVlEVlFRRE
V4Rk1kWFI2SUZSbGMzUWdVbTl2ZENCRFFUQ0NBU0l3RFFZSktvWklodmNOQVFFQkJRQURnZ0VQQURDQ0FRb0NnZ0VCQU1SK1Bwdjg2MGthWUJRSXFpQUMzK243QzA
0bEZ6SVlzSlY2N2xPeS9LVW9STmJBWSsxeFdsK281emNCL3hZcnBNc0RVNUtBb1BrSXhvWmN0cWJ0OStYblRyekN3WDExSG1LWjA2WjZka2hMYTRhS3RWak85dFFX
U1lOQzhIbXVmYW5yZEhxNVk5U21JWjRFQnhJYWJFaklHRGhCVGM4QkFiWkVnTVpXYnJRZ2k1cVc0bDErZHgrN0pNWXJ0ZUJSbG4rSUhLOGRsSitnakx6eVpSWkFXN
GhLSUFyRTh0dnJHZDdqd2tXV1FlbTVCaVFBcHlpV3JVQXltNTVGd3RUclZid2VsVDVOMDJQZ3YzREtjRDBJR3Rqd24ydGMzc2NPNmpmZDE4cW4vT2d3YnBQbnlxaU
w0NGpFbnhOS2xYTy9telpldkxPejk5V1kvOGFQa0Rua3RSVUNBd0VBQWFOeU1IQXdEd1lEVlIwVEFRSC9CQVV3QXdFQi96QWRCZ05WSFE0RUZnUVVBN0hXWWdnUkg
2dGZUOVBCeXVUdmJUbWNEWll3Q3dZRFZSMFBCQVFEQWdFR01CRUdDV0NHU0FHRytFSUJBUVFFQXdJQUJ6QWVCZ2xnaGtnQmh2aENBUTBFRVJZUGVHTmhJR05sY25S
cFptbGpZWFJsTUEwR0NTcUdTSWIzRFFFQkN3VUFBNElCQVFDR0cySjBSOGI2ZU1GRWlMeCtTbGFIbGlQclU5M3FRMGxTcmV5RlNGMDdBbG5zczYxZVZ1T2o2dzQ5N
EcrYXgrbGR6N2loc2JmWmV6eVloTUJONWlZSjFvbW15ZGU4VHNQSkhaWE1LVU54ZEdOSVM3S3BJSWJjNWtNNGQ2OFhWc0dFb2E1YjRUa2V2ZGI3TEZRRzVhVktYem
Zac2ZweEU1QzNlclAvVzdYTkJsYVhPK0VVemZTUmZTTjNOTWQ1K1VCeUVNVU9MdVZGeWFzTnRFaFRYa0hDOEdWcDllcHJ2Rk03dTJlNlBxd1pwbkxTSDhnVkYyOGh
yZEVGR0VmYTIwcjRQWmd3KzRvbkpZZzBqb3NJbDZyTWVpaXAxM3J5Zl1saWs0czg0WUpPa2tPcGxCNkcwRHVBbzE0N1JUbExyN3B4Z2FnamlaLytvM2NBdnozaUl1
THoiXX0.eyJqdGkiOiIyM2M1NTRmNS02MGU3LTQ3YzItOTQ3Yy1kZmQ2ODNjMzU1ZGYiLCJzdWIiOiJjbGllbnQuand0IiwiaWF0IjoxNjAxMDM1MDMzLCJlbWFpb
CI6ImxqYWVuaWNrZUBwaG9lbml4Y29udGFjdC5jb20iLCJuYmYiOjE2MDEwMzUwMzMsImV4cCI6MTYwMTAzNTA5MywiaXNzIjoiY2xpZW50Lmp3dCIsImF1ZCI6Im
h0dHBzOi8vYWRtaW4tc2hlbGwtaW8uY29tOjUwMDAxL2Nvbm5lY3QvdG9rZW4ifQ.NDk2nfIiv4-
Jk1ZL7yL6rvNgPhU6J36YZsfr2COf1LIOf7wTwPJeZ9dUAXQ77NF4GV49PYS-Nx5NBfMoy5Qy4dg3eJqzllCw1VGasjbbVYpX-hx3o6fQA6P2R-
Xpq0vHNcbL0dlubh8DZvwSB6OtkRnEiOy3i_fTwR99ZLVBclFnQ-
uh2FdzpFl8MmSDzBaR5FcEhMbuyXIJyZYAKPn8noHzWsftNCQeAz477P2aLWW4zN0MDuPcjM8DQuQ1IBs0U8W4v4bW0gnr-
OGVrneHJ94nBgIBsjkUtLqV4WCitkWWHX3CERRvvUKPvG-ITvoOLwdUYHR54orCBqjf5nZBQQ
```

AUTOREN

André Braunmandl, Bundesamt für Sicherheit in der Informationstechnik | Vanessa Bellinghausen, Bundesamt für Sicherheit in der Informationstechnik | Holger Blasum, SYSGO GmbH | Dr. Birgit Boss, Robert Bosch GmbH | Dr. Gerd Brost, Fraunhofer AISEC | Sebastian Fandrich, SICK AG | Kai Fischer, Siemens AG | Björn Flubacher, Bundesamt für Sicherheit in der Informationstechnik | Kai Garrels, ABB STOTZ-KONTAKT GmbH | Markus Heintel, Siemens AG | Dr. Michael Hoffmeister, Festo SE & Co. KG | Dr. Detlef Houdeau, Infineon AG | Dr. Lutz Jänicke (Leitung), PHOENIX CONTACT GmbH & Co. KG | Michael Jochem, Robert Bosch GmbH | Thomas Lantermann, Mitsubishi Electric B.V. | Jens Mehrfeld, Bundesamt für Sicherheit in der Informationstechnik | Andreas Orzelski, PHOENIX CONTACT GmbH & Co. KG | Andreas Pfaff, Mitsubishi Electric B.V. | Dr. Mehran Roshandel, T-Systems International GmbH | Markus Ruppert, KOBIL Systems GmbH | Detlef Tenhagen, HARTING Stiftung & Co. KG | Jens Vialkowitsch, Robert Bosch GmbH | Thomas Walloschke, Industrie KI GmbH | Tianzhe Yu, Institut für Automation und Kommunikation e.V. |

Diese Publikation ist ein gemeinsames Ergebnis der Arbeitsgruppen „Sicherheit vernetzter Systeme“ und „Referenzarchitekturen, Standards und Normung“ (Plattform Industrie 4.0).

www.plattform-i40.de