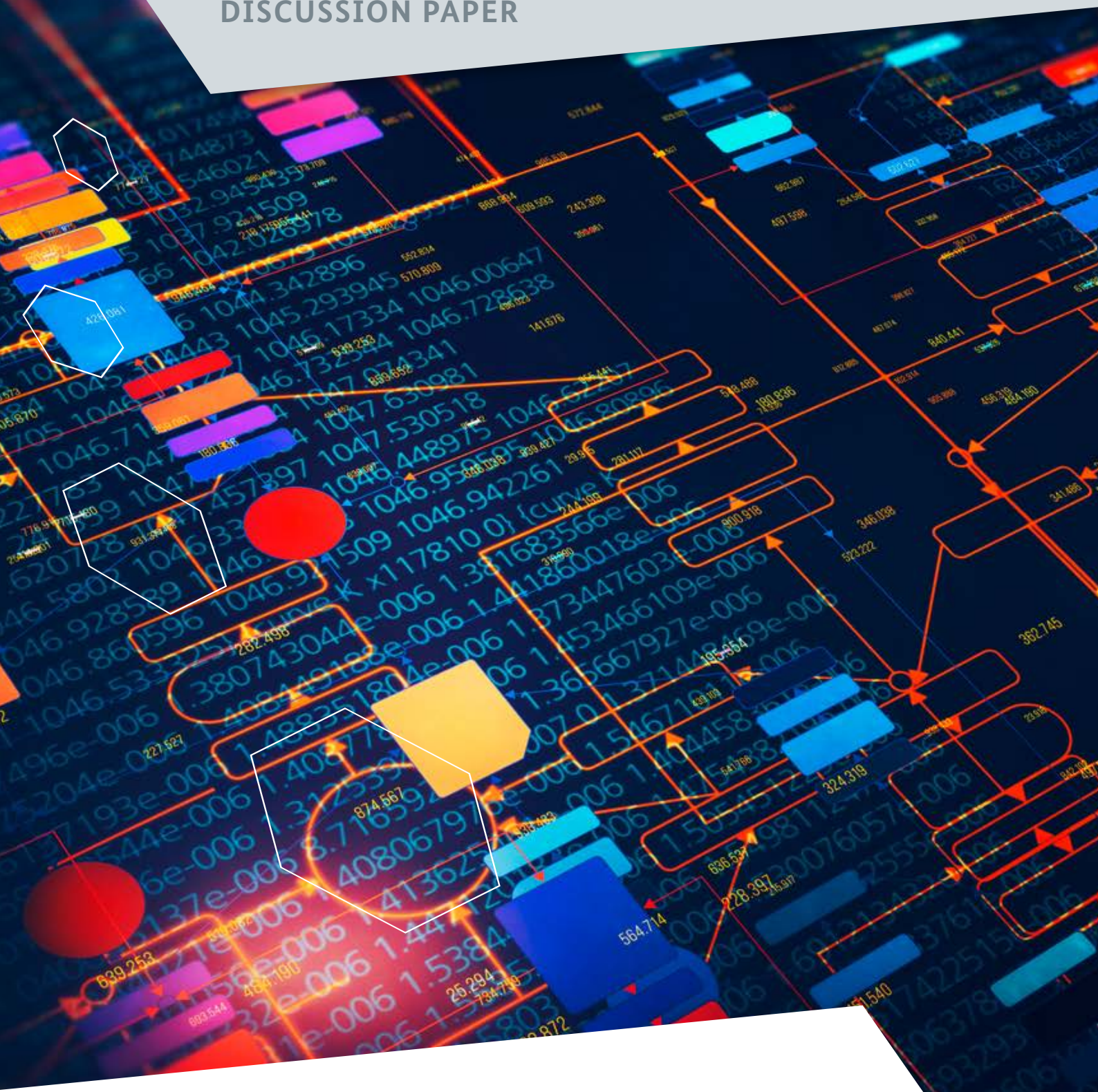
PLATTFORM INDUSTRIE4.0

DISCUSSION PAPER



Secure Download Service

Imprint

Publisher
Federal Ministry for Economic Affairs
and Energy (BMWi)
Public Relations
11019 Berlin
www.bmwi.de

Text and editing
Plattform Industrie 4.0
Bülowstraße 78
10783 Berlin

Design
PRpetuum GmbH, 80801 Munich

Status
October 2020

This publication is available for download only.

Picture credits
Plattform Industrie 4.0;
spainter_vfx – iStockphoto (Title)

Central ordering service for publications of the Federal Government:
Email: publikationen@bundesregierung.de
Tel.: +49 30 182722721
Fax: +49 30 18102722721

Content

1. Introduction

The interoperable retrieval of engineering data is crucial to the implementation of Industrie 4.0. To this end, it is imperative that the requirements for interoperability are specified for both the contents and the data that are transferred and for the formats and protocols. Interoperability also includes the security measures with which confidentiality, integrity and availability are implemented in accordance with the requirements of the application scenario.

1.1 Content and aim of this discussion paper

The present document addresses the considerations outlined in the discussion paper "Secure Retrieval of CAE Data" (1) and develops the considerations relating to their transfer further in technical terms. Communication protocols (HTTPS/REST), identity management and authentication in particular are discussed, taking OpenID Connect as an example. The formats and content of the data are discussed elsewhere in other documents, primarily in the series entitled "Details of the Asset Administration Shell" (2). In this discussion paper a solution concept is presented which provides for the fully automated retrieval of engineering data, including authentication and authorisation management based on the further development of existing standards, the ultimate aim being to bring about the further development of appropriate standards.

The technical depth of this document is sufficient to set up demonstrators for the described application scenario.

This document is aimed at the technically versed reader.

2. Application scenario

The document entitled “Details of the Asset Administration Shell” (2) discusses the transfer of data between the stakeholders involved.

2.1 Overview

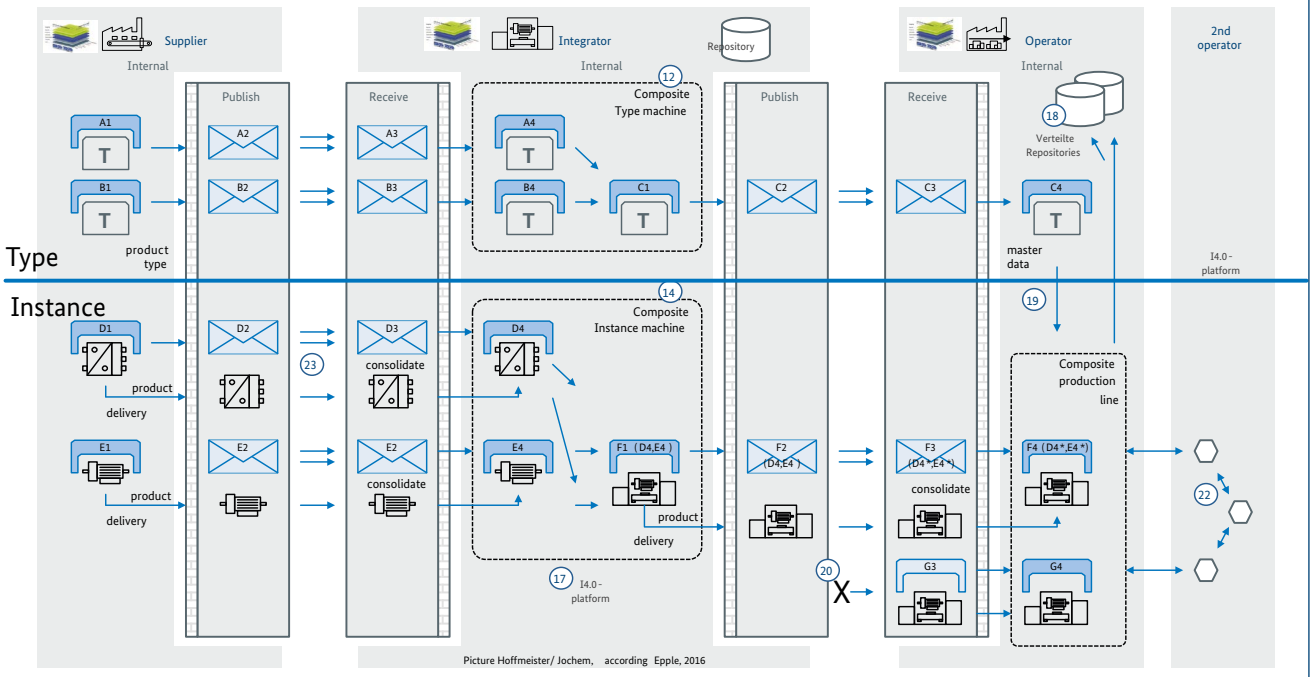
Figure 1 shows the steps involved in transferring the type and instance data along the value chain from the supplier via the integrator to the operator. The individual type data are used to develop the automated solution and are needed when the engineering process begins, before the machine or production facility is actually set up.

The instance data are then valid for individual examples of the products used or the machines or facilities assembled from them and can contain quality or calibration data, for example, to support the overall solution.

2.2 Transfer of engineering data

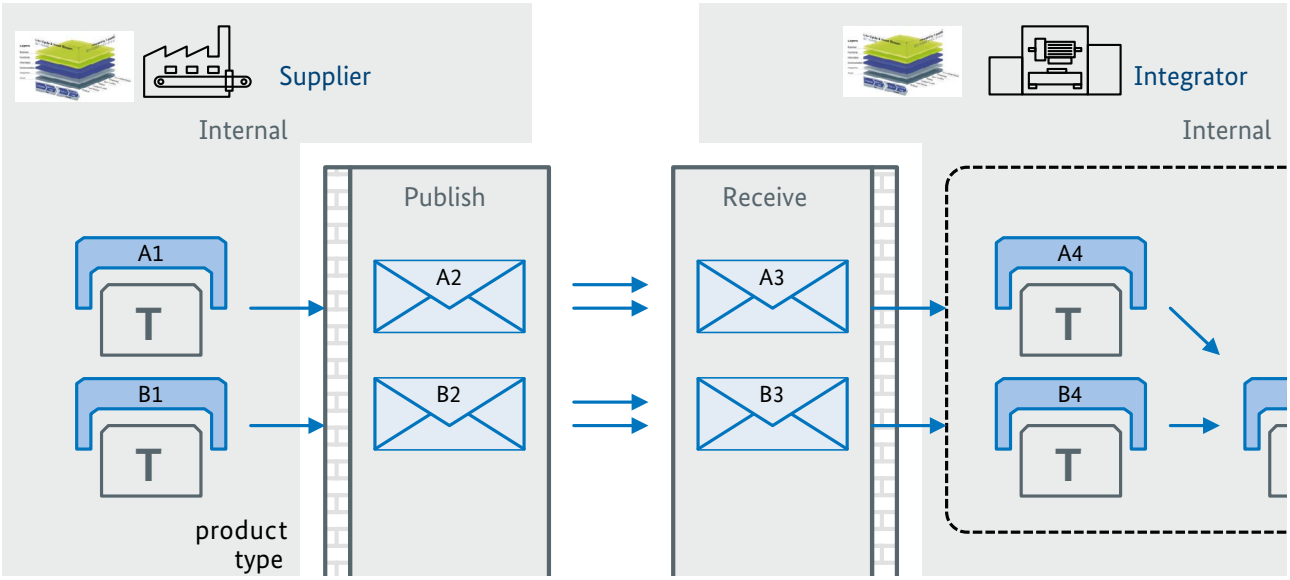
Figure 2 shows an example of the transfer of engineering data from a manufacturer to an integrator. In this process, the type data are obtained from the manufacturer’s systems, transferred to the integrator and then loaded into its systems.

Figure 1: Overall scenario taken from (2). Above: type information; Below: instance data



Source: ZVEI

Figure 2: Transfer of type information

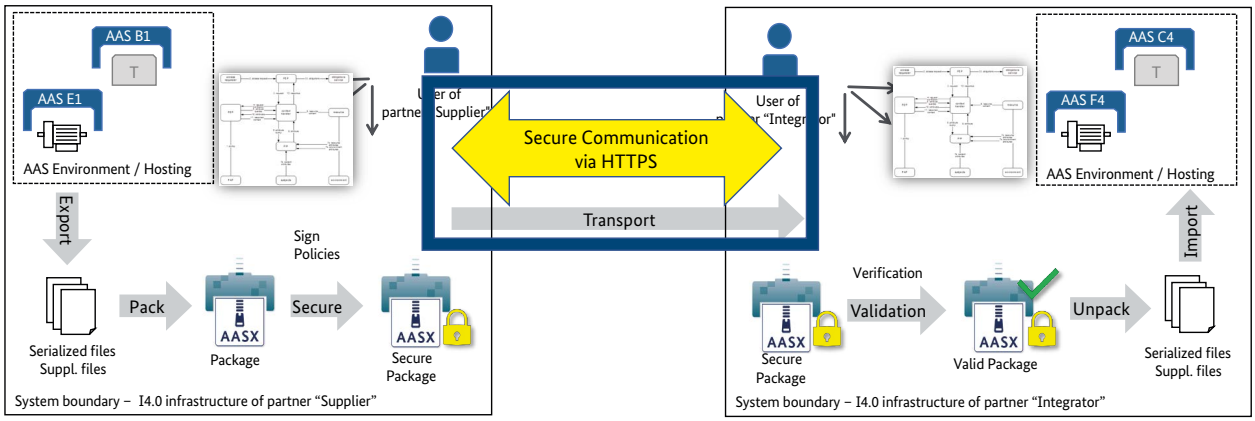


Source: ZVEI

Figure 3 shows the specific task considered in this discussion paper, in which the transfer takes place as an online operation. The integrator has a requirement for type information and establishes an online connection to the manufacturer (supplier). It is assumed in this case that an HTTPS connection is used, which is now standard technology for Internet traffic.

The manufacturer checks the identity of the requesting partner and, based on his authorisation concept, prepares the serialised type data in the form of a file for transfer. The authenticity of the file is verified by the requesting partner and then the data are loaded into the integrator's systems and are available for use.

Figure 3: Steps in the transfer process

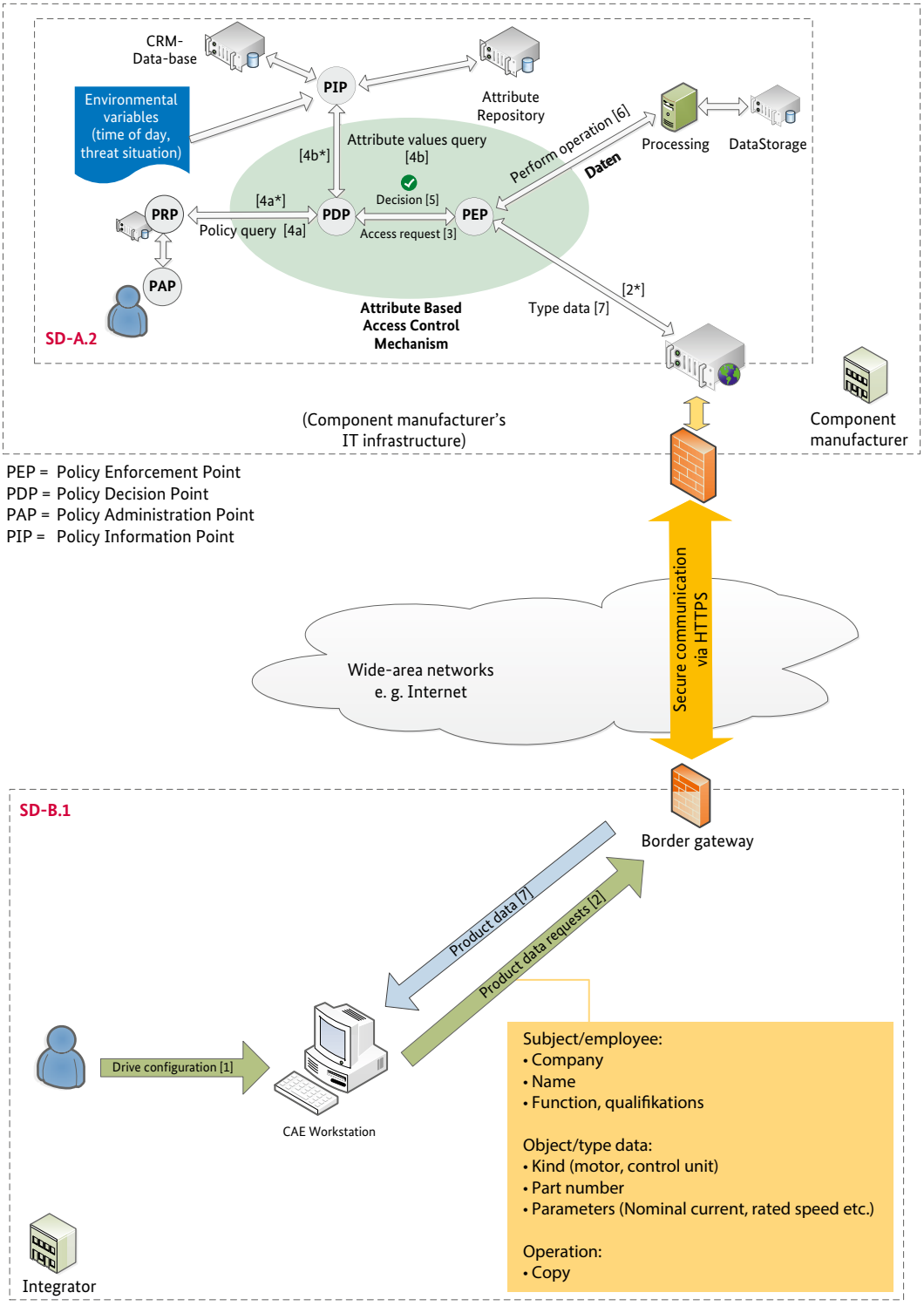


Source: Plattform Industrie 4.0

3. Assumptions and requirements

A fundamental analysis of the transfer process has already been made in the discussion paper entitled “Secure Retrieval of CAE Data” (1) and this is now referred to in the following chapters – see also Figure 4, which shows the retrieval of data from one of the integrator’s workstations, with the information sources being the systems located at the component manufacturer’s facilities. The type data to be obtained can include everything that is required for the product description and its use in engineering. This can, for example, be technical data or 3D models, but also software and firmware that could be required for simulation purposes or communication with the product (driver).

Figure 4: Implementation of the secure download of type information



Source: Plattform Industrie 4.0

For discussion purposes, it is assumed, as is the case in "Secure Retrieval of CAE Data" (1), that the authenticity of the data in AASX format has been verified in accordance with the "Open Packaging Convention" (3) standard used and does not need to be considered in more detail. The further processing and security of the data in the receiving system are also outside the scope of this discussion.

The solution presented in this paper implements a security approach that extends from one application layer (CAE software on the workstation) to another application layer (systems at the component manufacturer's facility).

For practical applications, it is important to take in to account the real-life operating conditions that occur in cross-company use. Figure 4 shows the various security domains involved. Security gateways are typically installed at the boundaries of the security domains, which act as firewalls or (TLS) proxies. A uniform user management system between both domains cannot initially be assumed.

3.1 Requirements

Modern design principles ensure that tasks are divided into separate services to provide independent development and maintenance: Service Oriented Architecture (SOA). One such arrangement is to implement the security approach with the aid of microservices that interact with each other. A solution concept must provide for a corresponding architecture regardless of the actual implementation.

In the following sections, the relevant services are described from the point of view of the documents currently being developed and revised for the "Details of the Administration Shell" (2).

3.1.1 Provisioning of the requested data

For the provisioning of engineering data scenario, it can be assumed that a component manufacturer has a portfolio of products and insofar wants to provide more than just the data for a single product. The download server will therefore provide quite a large range of products, but not everyone will be given unlimited access to it. For the provisioning of data on individual product instances, it will also be necessary to consider the fact that access will be limited. On the download server, described in general terms in the following as a resource server, an access control system will have to be implemented.

3.1.2 Authorisation

The Administration Shell model, as described in "Details of the Asset Administration Shell" (2) and "Access control for Industrie 4.0" (4) supports a separate access rights model for each Administration Shell. Authorisation is granted in accordance with the Attribute Based Access Control (ABAC) model and, in order to decide on access rights, it requires information (attributes) on the subject requesting access, the desired action and the object to be accessed. Information on the subject is verified and confirmed during the authentication process.

3.1.3 Authentication

Authentication provides the attributes that describe the requesting partner (subject) required by the authorisation process for the decision. Attributes may originate from a user management system at the component manufacturer's facility or be transferred by the requester. The authentication server vouches for the correctness of the attributes, not however for their interpretation in access control in the resource server.

3.1.4 Scalability

For discussion purposes, it has been assumed that the download service is designed to be scalable, i. e. data are to be supplied in large quantities and at high frequencies. This provides the basis for determining the demands placed on the architecture and how the required services are to be distributed.

Another aspect of scalability is cross-company collaboration, where, for example, the use of filtering proxies must be taken into account. It can be assumed that consumers (human users or software applications and systems) from a large number of different companies will retrieve the data. This means that the individual consumers will also have to be identified and access rights assigned to them based on specific characteristics (attributes). As far as the scalability of the download service is concerned, the management of attributes should not have to be carried out by the company supplying the data.

4. Outline solution/discussion

In the following outline solution, an example of an approach is presented, which meets the requirements described above. The proposed technologies, interfaces and services are intended to demonstrate the feasibility of the approach and should be regarded only as examples. In an actual implementation, the various services could be distributed as proposed or implemented jointly and use other internal interfaces.

The outline solution is also intended to separate the application logic (in this case: the REST-API being analysed) and the embedding of the authentication information in the communication in such a way that both elements can be expanded and further developed independently of each other. In the web services environment being analysed, the required resources are available, since authentication can be carried out using the http headers, while the URI and the "body" are used for the user data.

4.1 HTTPS/REST communication protocols

As already described in the discussion paper entitled "Secure Retrieval of CAE Data" (1), use of the HTTP(S) protocol is also most likely to be applicable in a cross-company context. Having said that, it should be taken into account that the communication at company boundaries will be filtered using TLS proxies and the HTTPS protocol will subsequently lose its ability to use client certificates for authentication purposes. This is due to the fact that mutual authentication in the protocol is designed specifically to prevent man-in-the-middle (MitM) attacks and to protect the integrity and confidentiality of data between the two endpoints. An inspecting TLS proxy, on the other hand, must break confidentiality in order to do its job. Even so, using HTTPS makes it possible to protect the communication from eavesdropping and manipulation outside the security domains of the companies involved, since TLS connections are re-established by the corresponding proxies and the communication outside the company boundaries will once again be protected. This expressly includes the logon data and tokens used for authentication and authorization purposes.

As a result of their simple structure and clear models, interfaces for automated communications (web services) are often implemented using REST (REpresentional State Transfer), which uses the http elements GET, POST, PUT and DELETE. REST is proxy friendly. A REST interface for Administration Shells is described in part 2 of "Details of the Asset Administration Shell". It permits access to single data elements, which can be either endpoints, submodels or entire Administration Shells.

To ensure that the proxies commonly used in corporate IT accept the communication connection to the REST interface described above, for example, the server certificates must be issued by a Certification Authority (CA) that is installed on the proxy as trustworthy. In typical business operations, these will be CAs from the web environment.

4.1.1 Handshake between the parties

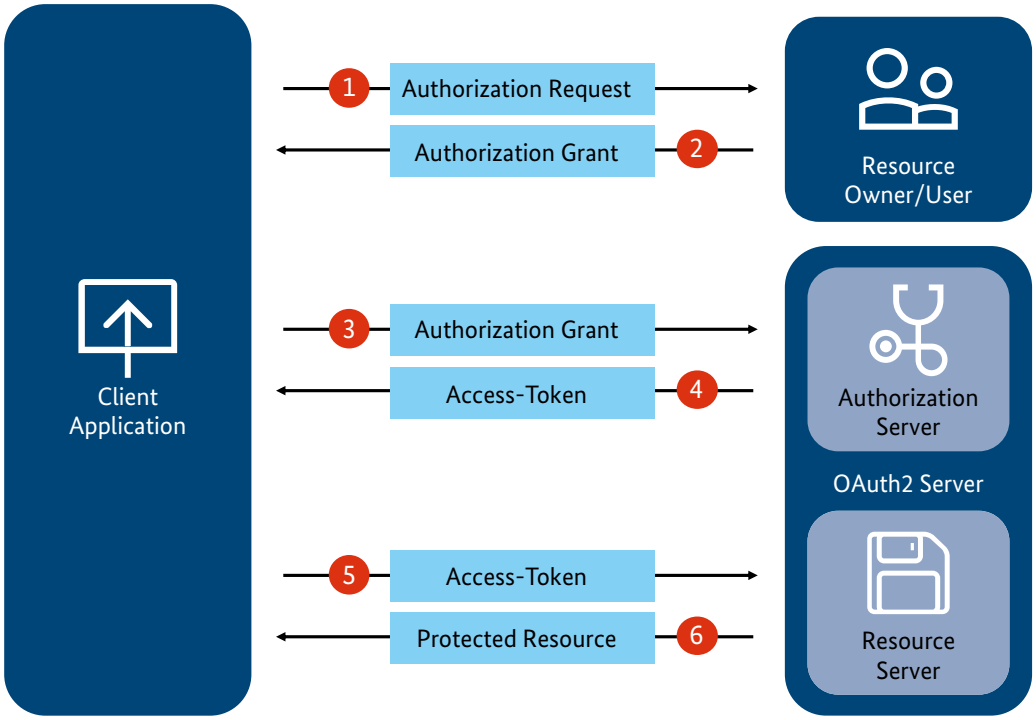
Whether the access rights to the information provisioned by the server are sufficient for access is decided by the access control system. According to the concepts provided in the document entitled "Access control for Industrie 4.0" (4), different decisions can be taken for each Administration Shell. For the protocol, therefore, it is necessary to ensure that the client can be provided with qualified feedback. This feedback could, for example, be "For logged-in users only" or "For manufacturer X's CAE systems only", so that the client can update its login accordingly. To what extent the server is to provide this qualified feedback or to reject the request without offering any further information to ensure that a potential attacker is given no information is to be determined on the basis of the security policy.

4.2 Identity management using OpenID Connect

Identity management is an issue that is dealt with intensively in the web environment. On a technical level, the transfer of information using JSON Web Tokens (JWT) is an established procedure that can be managed in a more compact way than with XML techniques. JSON Web Tokens are frequently used in HTML metadata, as Bearer Tokens, for example. In the context of this document, JSON Web Token refers not only to the basic technology in accordance with industry standard RFC 7519 "JSON Web Token (JWT)", but also to the entire technology family, including RFC 7515 "JSON Web Signature (JWS)" and RFC 7517 "JSON Web Key (JWK)" (5).

A key concept is the "OpenAuth2.0 authorization framework" in accordance with RFC 8252, which introduces the interaction between a client (e.g. an application), a "resource owner" (approver) and the resources by introducing an "authorization server", see Figure 5. By using this concept, the "resource server" can be relieved of the task of user management and is able to transfer this task to a dedicated service.

Figure 5: Elements of the OAuth2 Authorization Framework



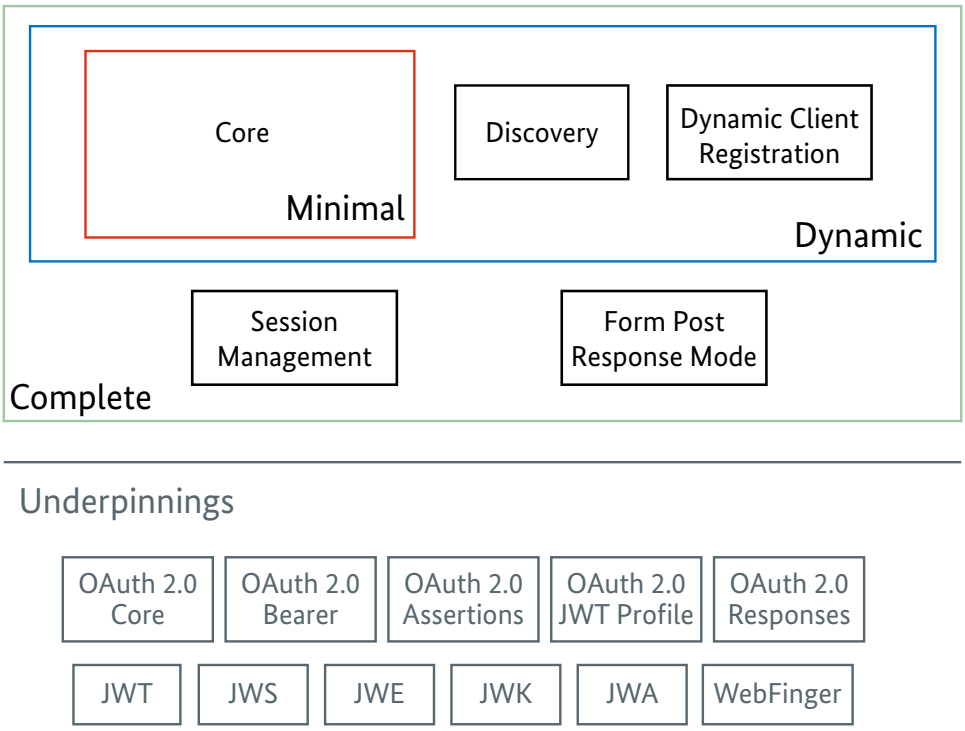
Source: Plattform Industrie 4.0

Key elements of OAuth2 communications make use of the JSON Web techniques mentioned above. For this to take place, a trust relationship must be established between the resource server and the authorization server. From a technical viewpoint, a digital signature in the JSON Web Signature (JWS) format at the bottom of the access token that was issued is sufficient for this purpose.

A limitation of the OAuth2 concept is the fact that the authorization server not only performs the authentication as discussed in the requirements, but also assigns rights in the "access token". This, however, contradicts the concepts of "Access control for Industrie 4.0" (4).

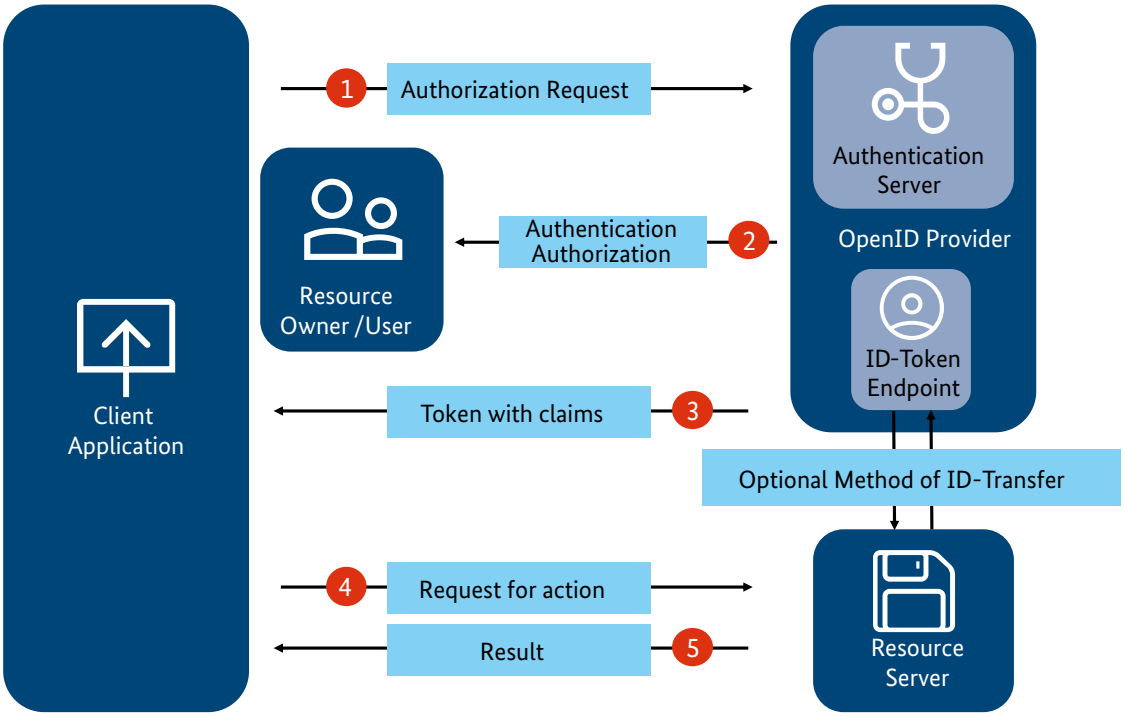
This restriction has been removed in the OpenID Connect framework, see Figure 6, The OpenID Connect framework, technically speaking, is based on OAuth2, but separates authentication and authorisation. Instead of assigning rights in the "access token", additional "identity tokens" are introduced, which are used directly by the client or alternatively can be exchanged directly between the resource server and the authentication server, for example, to reduce the bandwidth requirement, see Figure 7. The identity tokens contain attributes called "claims" about the user, whereby the claims described in the specification correspond to the application for private users and contain name, address and date of birth, for example, but do not include attributes of interest to Industrie 4.0, such as affiliation to an organization or qualifications. The specification does, however, allow a domain-specific extension to be installed that could be used to describe attributes of relevance to Industrie 4.0. Proposals for such additional attributes, however, are not included in this document.

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



Source: Plattform Industrie 4.0

Figure 7: Separate authentication service with OpenID Connect



Source: Plattform Industrie 4.0

Responsibility for the correct authentication of the client and/or other parties involved and the provisioning of the claims lies with the authentication server.

4.2.1 Authorisation using ABAC

Based on the claims in the ID Token, authorisation for access to the data of the Administration Shells can be performed as described in "Access Control for Industrie 4.0" (4) and the document "Details of the Asset Administration Shell, Part 1" (2). The implementation of access control is performed in the resource server, which has to evaluate the transferred attributes. As a result of using the claims provisioned by the authentication server, the implementation is completely decoupled from the type of authentication and can be further developed independently.

4.2.2 Authentication using secure, cryptographic methods

Most of the authentication models used on the web target the human user and, therefore, generally offer the use of passwords. The associated risks have already been discussed in sufficient detail elsewhere. Solution concepts available on the web are based, for example, on two-factor authentication (2FA), in which, in addition to the normal login, a code that is valid for a short time only is sent by SMS, e-mail or any other application that can send text[1], or on the use of cryptographic keys. Typically, the cryptographic keys are coupled with certificates that are used to establish the secure connection, e.g. in the TLS handshake for mutual authentication. In the case of connections via inspecting TLS proxies, however, it is not possible to establish such a connection.

The use of cryptographic keys for authentication in web applications is described in the W3C's WebAuthn standard (6). The method is designed to be used in web browsers and for the use of cryptographic keys (without certificates and thus without providing information on the user's identity) for a centralised user management system and is thus not really suitable for the present application.

The use of WebAuthn for authentication in the OpenID Connect framework is analysed in various papers, including a draft proposal for an extension of OpenID Connect (7). A significant feature of WebAuthn is that it supports attestation of a secure key store.

4.2.3 Mutual agreement on trust anchors

The use of electronic, digital certificates depends on mutual agreement and trust in the underlying Public Key Infrastructures (PKI). In the present application scenario, the requesting party is required to authenticate itself with X.509 certificates. Accordingly, the certificates of the issuing CAs must be recognised and stored as trustworthy on the authentication server. Accredited public CAs with known security qualities could be considered as certificate issuers. Companies do not usually use certificates from public CAs for the authentication of their employees and computers, however, but prefer to use certificates from company owned CAs – there are also cost reasons for doing this.

For the present application scenario, therefore, an upstream process in which the relevant trust anchors are exchanged is included to enable them to be referred to during day-to-day business operations.

4.2.3.1 Assessment of the trust in the foreign root CAs

For the assessment of trustworthiness, manual processes are currently in place that are initiated when the business relationship is established. The actual assessment is carried out by referring to the policies of the CA, which are described in documents such as the Certification Practice Statement (CPS). Document ETSI TS 102 042 entitled "Electronic Signatures and Infrastructures (ESI); Policy requirements for certification authorities issuing public key certificates", for example, describes fundamental policies that may be used as a reference. The European Bridge CA[2], which is organised by TeleTrusT, also describes a common minimum security standard. As a result of the eIDAS Regulation "Electronic IDentification, Authentication and trust Services"[3], a legal framework has been established within the European Union for the mutual recognition of trust service providers for the creation of digital signatures and seals. The document entitled 'Trust infrastructures in the context of Industrie 4.0' (8) describes ways of using eIDAS as a common trust anchor. The document entitled "IIOT Value Chain Security – The Role of Trustworthiness" (9), which was prepared by Plattform

1 Other methods result from the availability of special ("authenticator") apps, for example, which act as independent channels on mobile devices.

2 https://www.ebca.de

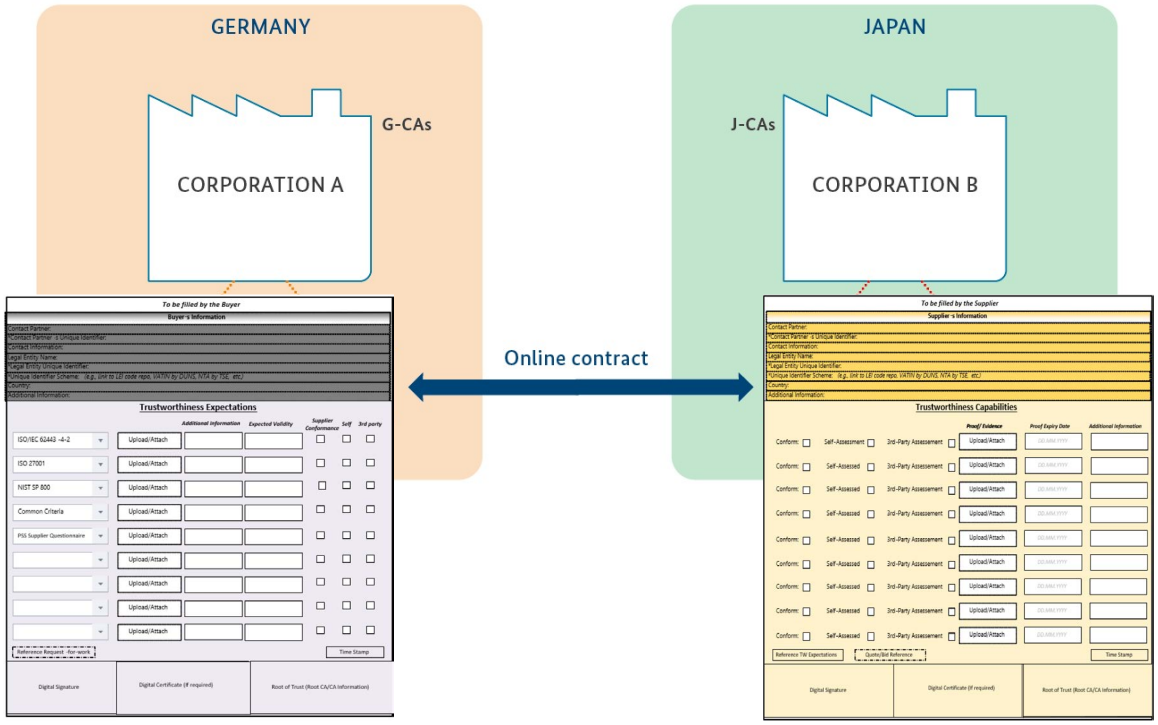
3 Reference: REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 23 July 2014, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32014R0910

Industrie 4.0 in collaboration with the Robot Revolution Initiative, describes a model in which the establishment of a business relationship would be based on a trustworthy exchange of security requirements and proofs, see Figure 8. The provisioning of trust anchors could also be integrated into this process.

4.2.3.2 Secure provisioning of CA certificates

If the parties agree on the security policies, then the trust anchors can be exchanged. More than one CA certificate can also be used for each of the partners. This is necessary because several certificates must be supported simultaneously at the time of a rollover, for example. It can also happen that different CAs are used for different areas of application, such as employees and systems in the same company, or for products. To ensure that the trust anchors themselves are authentic, they could be confirmed by the provisioning company by authenticating them with a seal that has been issued to the company. A basis for mutual trust in the case of such seals, for example, would be the trust infrastructure meeting the requirements of the eIDAS Regulation.

Figure 8: Trustworthiness exchange model based on (9)



Source: Plattform Industrie 4.0

5. Token-based authentication

OpenID Connect provides no guidelines whatsoever as to which authentication method is to be used on the authentication server. Instead, OpenID Connect describes how the information on the requesting party is provisioned by the authentication server of the trusting instance, in this case the resource server. The authentication itself, among other things, makes use of the mechanisms specified for OAuth2. Examples and presentations normally originate in a central user database on the authentication server.

A central user database of this kind, however, is not flexible enough to be used in the scalable environment of Industrie 4.0. The present models are therefore to be extended in such a way that they allow digital certificates to be used, supplemented as and when necessary by additional attributes, e.g. attribute certificates.

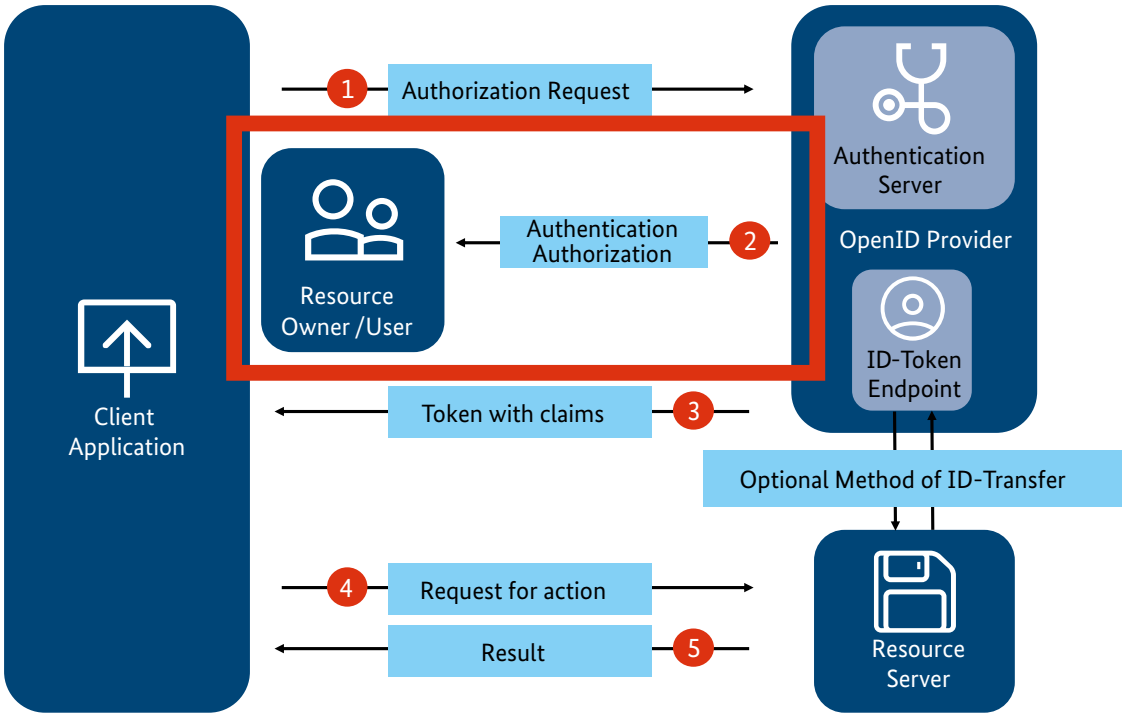
The following discussion focuses on the authentication step in the process, see Figure 9.

5.1 Use of certificates for mutual authentication in TLS

"OAuth 2.0 Mutual TLS Client Authentication and Certificate-Bound Access Tokens" (RFC 8705) uses TLS connections that are not interrupted by an inspecting TLS proxy, see Figure 10. In the context of the TLS protocol, "client certificate" refers to the digital certificate of the requesting party. In OpenID Connect terminology, this would not be the client (requesting application), however, but the resource owner. It is very frequently the case that client certificates are provided by the client application. The underlying use of client certificates is specified in the TLS protocol (RFC 8446 for TLS 1.3). If the server would like to receive client certificates, it sends a "certificate request" message to the client. The server can transmit a list of acceptable issuers of client certificates in the optional "Certificate Authorities" structure. The client can now select a client certificate from the certificates made available to it and, if necessary, transmit it to the server with its certificate chain. In the context of a human user, the end-user (resource owner) would be shown a list of applicable certificates to choose from by the web browser. In an automated procedure, the application must choose the appropriate certificate. The server can now decide whether to accept the client certificate for login and then either continue establishing the connection or cancel as the case may be. The reliability of the process tends to decline if no list of acceptable issuers (trust store) is sent or is otherwise defective.

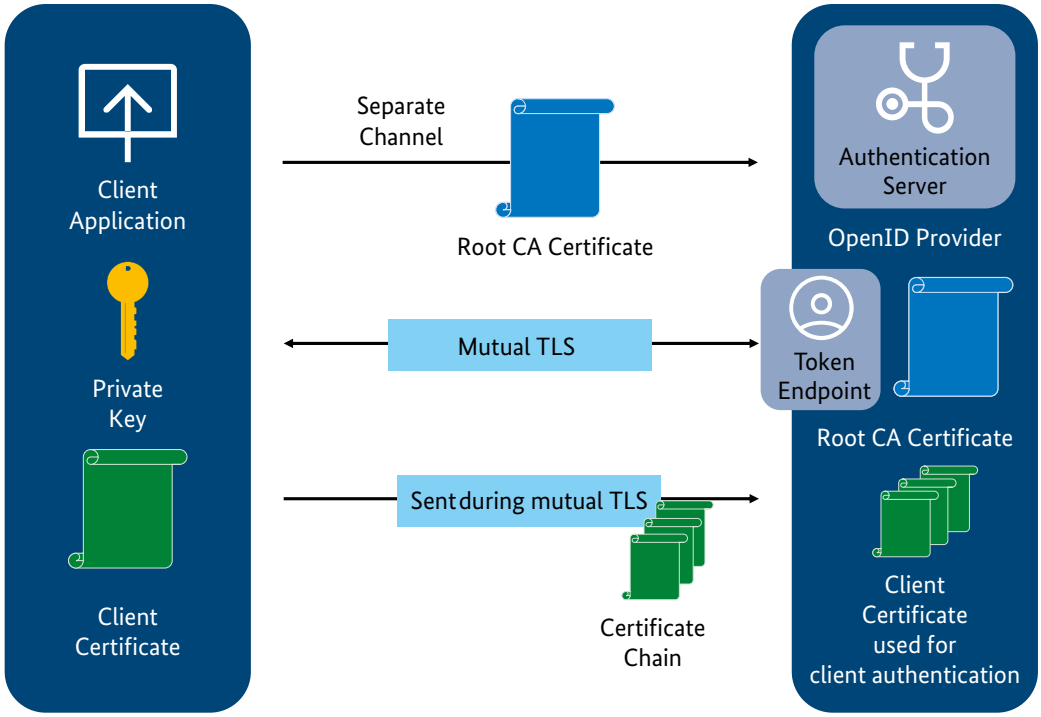
In the event that client certificates are used, authentication takes place as shown in Step 2 in Figure 9 in the TLS handshake of the HTTPS protocol, see Figure 10.

Figure 9: Authentication step in the process



Source: Plattform Industrie 4.0

Figure 10: Mutual TLS client authentication in accordance with RFC 8705



Source: Plattform Industrie 4.0

It should be noted that mutual authentication with certificates in this case takes place between the client and the authentication server. For the actual data exchange with the resource server, the authentication of the server takes place in a separate TLS connection between client and resource server.

5.2 Use of tokens for authentication using cryptographic keys

To be able to perform the appropriate authentication with cryptographic keys, even if the TLS connection is interrupted by an inspecting TLS proxy, the authentication must be transferred from TLS layer to the application layer. For this purpose, OpenID Connect/OAuth2 use JSON Web Tokens (JWT, RFC 7519) with the appropriate extensions, such as JSON Web Signature (JWS, RFC 7515).

5.2.1 Cryptographically secure authentication using the private_key_jwt method

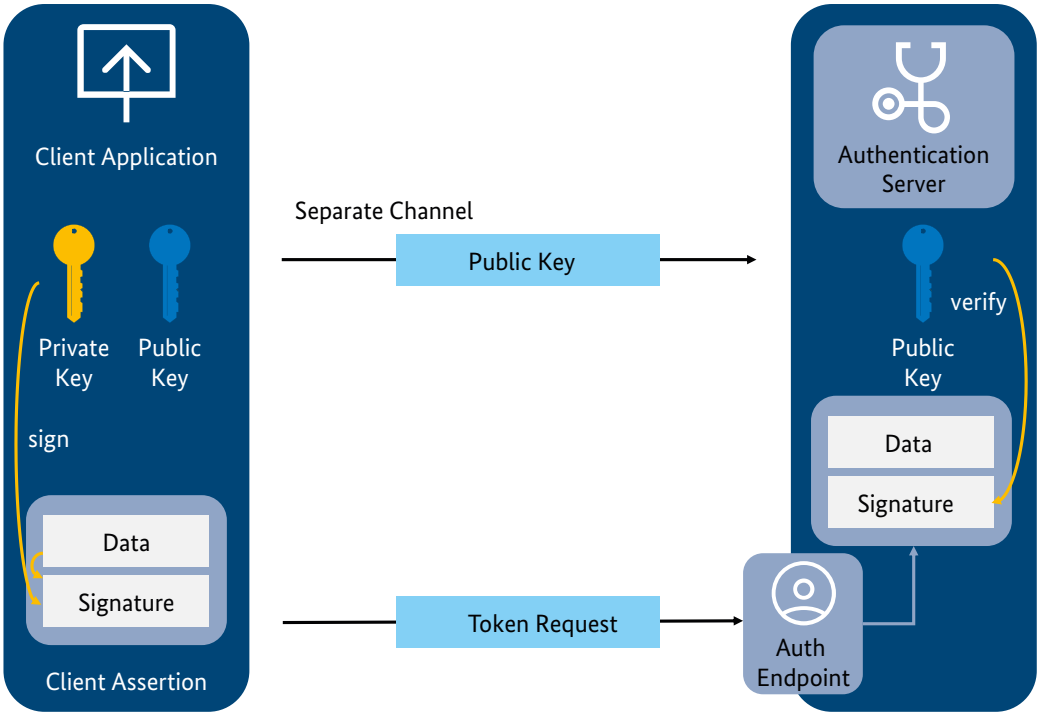
Figure 11 shows how the authentication of a client application specified in OpenID Connect Core works using the private_key_jwt method. The client application signs the data contained in the token with the present private key; the server can now verify the authenticity of the requestor by checking the signature against the public key of the requesting party.

For this verification procedure, the server must have access to the public key of the requesting party. This is of no benefit in the present context, however, since

- the public keys must be managed on the server (which requesting party is concealed behind the key?)
- the public keys must be made accessible to the server in a secure procedure
- when the keys are renewed, the information must be made known to the server again in a secure manner.

The problems mentioned above can be solved using the concept of the public key infrastructure (PKI), in which the public key is embedded in a certificate together with attributes describing the identity so that the key and the identity are bound together (identity management). The certificate is authenticated using a digital signature from a trusted authority (key management). These benefits would be applicable when direct TLS authentication is used, as described in Section 5.1.

Figure 11: Authentication using the private_key_jwt method

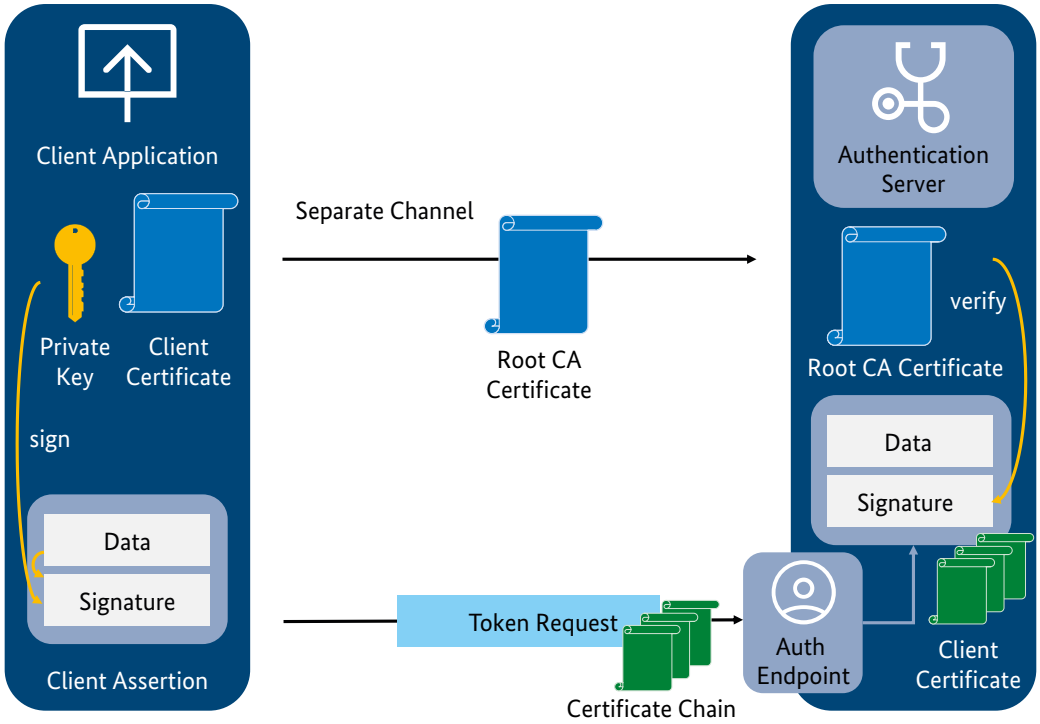


Source: Plattform Industrie 4.0

5.2.2 Proposed private_key_certchain_jwt method

To be able to enjoy the benefits of PKI also when using JSON Web Tokens, a new method "private_key_certchain_jwt" is now proposed. Figure 12 shows this concept, which is an extension of private_key_jwt by adding a certificate chain. In addition to the data to be transferred, the client certificate is transferred together with the supplementary certificate chain in the JSON Web Token. As described in RFC 7515 (JSON Web Signature), the "x5c" data structure is used for the certificate and chain. The server can now verify the client certificate against its list of acceptable issuers and perform further authentication based on the attributes in the certificate. An example of such an implementation is described in Chapter 11.

Figure 12: The private_key_certchain_jwt method



Source: Plattform Industrie 4.0

The implementation of additional security requirements, such as protection against replay attacks, is equivalent to using the "private_key_jwt" method. In this case, it would be helpful to determine whether the necessary security measures have already been taken into account.

The trustworthiness of the claims is based on the trust placed in the verification of the signature and the certificate chain. In this respect, the processes required for the acceptance and updating of the root CA certificates are an important part of the security concept. Furthermore, the security provided by the private key for the signature of the token request is a basic principle of its trustworthiness. In this respect, confirmation that the private key is protected by a security element should be supported by the process chain, for example by attestation or attributes in the certificates. The secure storage of keys could be shown as a claim in the ID token.

6. Example of the structure of a download session

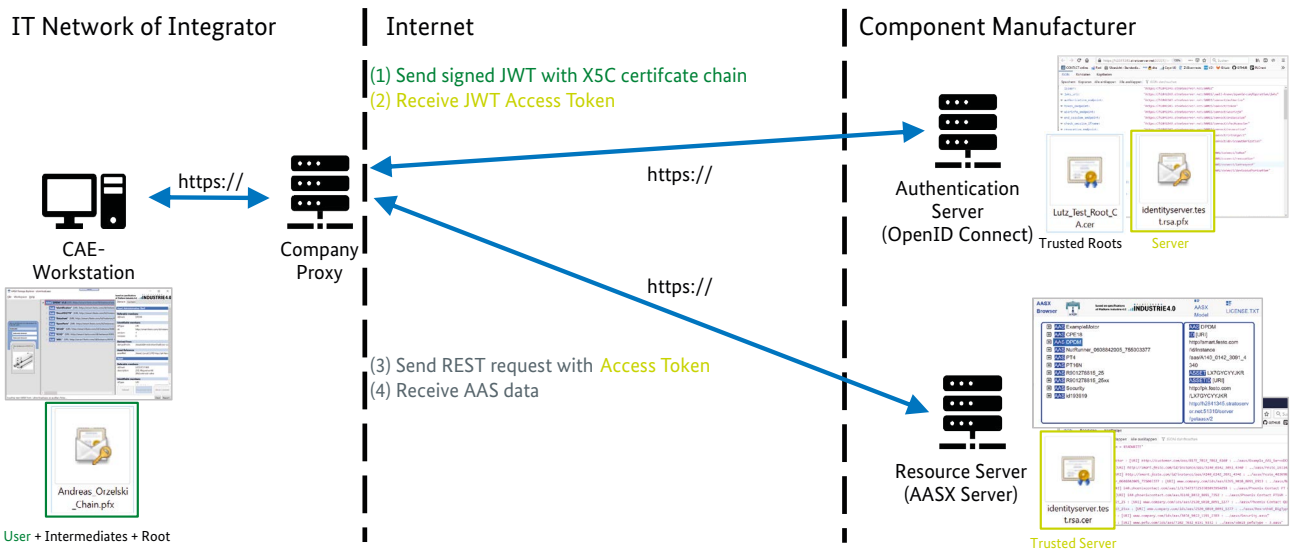
For the concept proposed above, a demonstrator was implemented, as shown in Figure 13, based on the AASX Explorer (10). For this purpose, an OpenID Connect server that had been made available as an open source was upgraded to an authentication server by adding the proposed "private_key_certchain_jwt" method, which was correspondingly implemented in the AASX Explorer. As a resource server, the AASX server must trust the authentication server and evaluate the tokens issued following authentication with the corresponding attributes (claims). The attributes are then used for access control in accordance with ABAC rules as described in "Details of the Asset Administration Shell" (2).

6.1 Handshake for automatic access

If, at the time the connection is established, the endpoints and the required attributes are known, because they were defined beforehand, then the desired data can be retrieved, as described in Figure 13. It is much more likely, however, that, although the endpoint for accessing the data is known on the resource server, the endpoint for authentication will be negotiated at runtime, since it is not linked to the actual application.

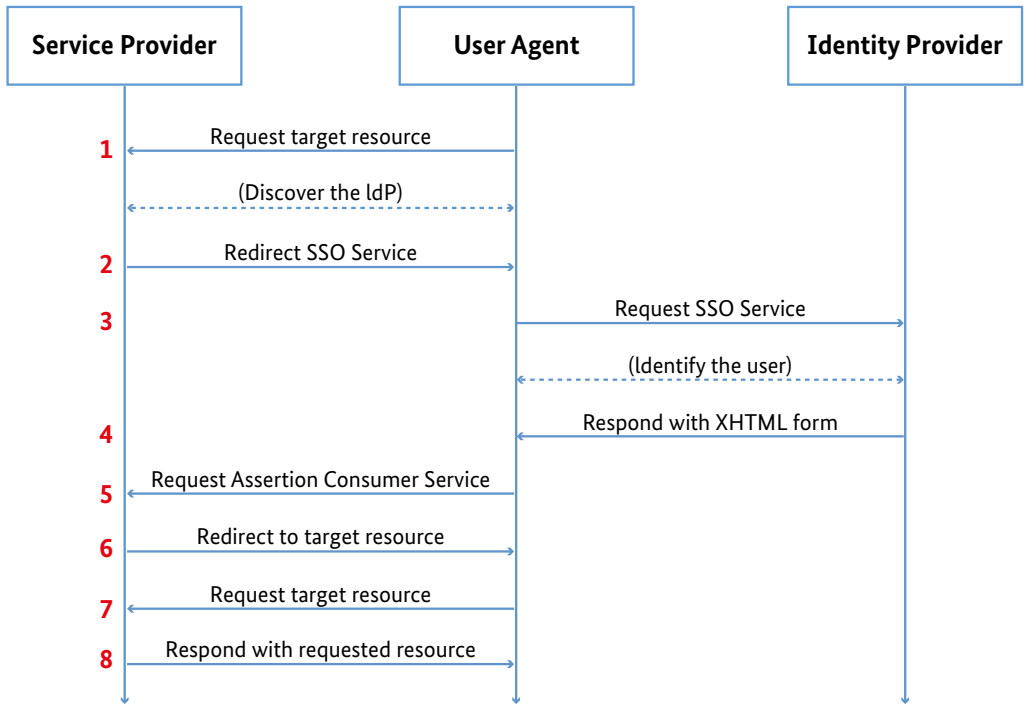
In the context of the Security Assertion Markup Language 2.0 (SAML 2.0), appropriate mechanisms are available to implement a corresponding interaction, see Figure 14. Its transferability to OpenID Connect needs to be examined. The user agent in this case would have be able to handle the redirect messages used.

Figure 13: Implemented demonstrator



Source: Plattform Industrie 4.0

Figure 14: Interaction in the context of SAML 2.0[4]



Source: Plattform Industrie 4.0

The following sections discuss how generic HTTP mechanisms can be used to enable REST to make use of the widest possible range of libraries via HTTP.

6.1.1 Reference to the authentication server

In the case of web services, especially for human users, it is common practice to refer to authentication servers before accessing non-public resources. There is a weakness here in the HTTP protocol, since it offers no redirect functionality for authentication purposes. According to the standard, the "401 Authentication Required" response cannot be combined with a "Location" header because the HTTP standard itself originally only provided for "Basic Authentication". For redirects, class "300" response status codes are included in the HTTP protocol. For the purpose intended here, where reference to the authentication server is for authentication purposes only, the use of a corresponding extension of the HTTP standard[5] would be the correct response. When a redirect response is returned, both the URL of the authentication server and the information required for authentication and the redirect, are sent in the "Location" header.

An example of a Location header could look as follows:

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

Information is returned to the client on how to reach the authentication server, while the URI also contains all the information required for the authentication server.

6.1.2 Client response

The client must now respond to the redirect in the appropriate way. To access the authentication server, the client must put the request that had been planned earlier on hold and access the authentication server. Since all the information was sent in the location header, the authentication server receives the additional information it requires to negotiate all attributes/properties (claims) with the client.

After authentication has been successfully completed, the client accepts the JSON Web Token (access token) provided by the authentication server as a bearer token for subsequent access to the resource server.

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

6.1.3 Authentication server response

The authentication server receives the additional information required for the handshake to succeed via the URI. This could be the set of provisioned attributes (claims), for example, that are necessary to access the desired resource. If this information is not available or is not supported by the implementation of the authentication server, the authentication server returns the standard attributes that are available.

The authentication server has the option of being able to decide not to provide the client with the ID token, but to provide the ID token to the resource server only via the ID token endpoint. This could reduce the required bandwidth, for example, or prevent the client from seeing which claims are provisioned for the resource server.

6.1.4 Resource server response

Once authentication has been successfully completed, the client accesses the resource server with the bearer token it has received. Access control in the resource server is based on the claims contained in the ID token and authenticated by the authentication server. If necessary, the resource server must obtain the ID token from the corresponding ID token endpoint of the authentication server.

6.1.5 Demonstrator

Chapter 11 describes the technical details of the implemented demonstrator. With the elements described in the solution concept, the intention is that it serve as a basis for discussion for the further development of the concept and the standards affected, in addition to leading to the implementation of solutions for productive use.

7. Summary and outlook

This present discussion paper describes a concept for a secure download service that takes into account the requirements of Industrie 4.0. Communication takes place online and on a cross-company basis. For scalability and security reasons, the concept of cryptographic keys and certificates is used. At the same time, the requirements are met for secure company processes, for which it is necessary to have control over incoming and outgoing communications and result in the use of inspecting proxies at company boundaries. This has resulted in authentication being shifted to the application layer and implemented by means of JSON Web Tokens. Wherever possible, industry standards such as OAuth2 and OpenID Connect are used. When client certificates are to be used, a new authentication method known as “private_key_certchain_jwt” has been proposed and is being tested in demonstrators. The concepts presented in this paper are based on the application scenario of downloading engineering data in that IT technologies such as HTTPS and REST are a prerequisite and the aim is upward scalability. Nevertheless, the concepts are not limited to this particular application scenario.

Based on the conclusions drawn from this discussion paper and the demonstrators, we recommend that in-depth development and standardisation should take place. To this end, the OpenID Foundation should be contacted with regard to technical standardisation. The proposed concepts can only be successfully applied if they receive widespread support from the providers of the relevant technologies and are accepted by the users.

8. Glossary

ABAC	Attribute Based Access Control
Authentication	Refers to proof or verification of authenticity
Authentication server	Dedicated service/endpoint for authentication
Authorisation	Issue of access permission based on authentication
Bearer Token	Its contents are trusted without further verification.
CAE	Computer Aided Engineering
Claim	Identity attribute, e.g. name, address, ...
Client/Client Application	Requesting application
HTTPS	Hypertext Transfer Protocol (S: secured via TLS)
Identity Provider	Service that verifies user identities and supplies ID information
ID Token	Token with attributes that describe the identity of the subject
JWT, JWS	JSON Web Token (RFC 7519), JSON Web Signature (RFC 7515)
OAUTH2	Protocol for delegated authorisation (RFC 6749)
OpenID Connect	Concept/protocol for delegated authentication
Resource Owner	Approver. In the web environment, this can be a user that accesses its own data stored on a resource server and permits its application to do so
Resource Server	Server that provisions relevant resources
SD	Security domain
SOA	Service oriented architecture
TLS	Transportation layer security

9. List of figures

10. References

1. **Discussion paper "Secure Retrieval of CAE Data". 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. **Discussion paper "Access control for Industrie 4.0 components for application by manufacturers, operators and integrators". 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. **Trust Infrastructures in the Context of Industrie 4.0.:** Plattform Industrie 4.0, 2020 (in preparation).

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. Technical details of the proposed solution concept

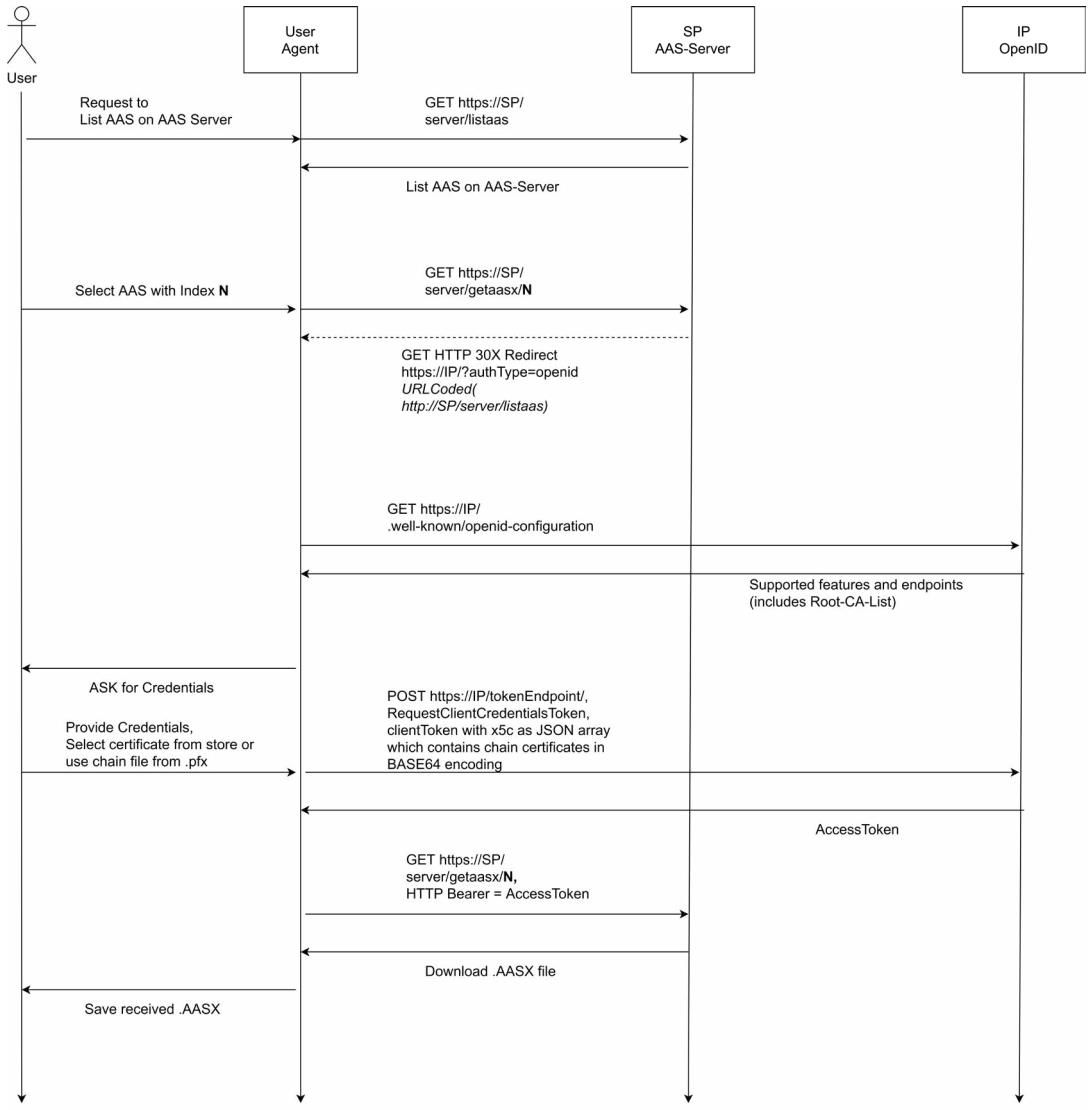
In the following sections, the concepts described above are illustrated with the aid of a demonstrator. Information on the demonstrator and source code can be obtained under an open source licence from "https://github.com/admin-shell-io".

11.1 Example of a process of authentication and authorisation

Figure 15 shows an example of an access sequence as implemented in the demonstrator. For this example, it is assumed that the user agent, e.g. an AASX explorer, initially requests a list of available Asset Administration Shells (AAS) for the user. In the example shown, this list would not be subject to any access restrictions and would therefore be provided without authentication..

When this has been done, the (AASX) file is to be downloaded from Asset Administration Shell N. However, since the User Agent at this time has either no token, or the token has expired, or is insufficient, the AAS server denies access and redirects the AAS client to the Identity Provider IP using an HTTP redirect. The AAS server provides the information that the Identity Provider uses OpenID Connect. Additional information which the Identity Provider needs in order to carry out the authentication process could be

Figure 15: Example of a demonstrator handshake



Source: Plattform Industrie 4.0

provided. In the example given, this is merely the URL of the Identity Provider's intended endpoint.

The User Agent's access to the identity provider using the supplementary information enables the Identity Provider to compile a corresponding authentication package for the User Agent that lists the supported functions and endpoints. On the basis of the OpenID Connect concept, a variety of authentication mechanisms could be supported that would have to be negotiated by a handshake. For the proposed "private_key_certchain_jwt" case, the Identity Provider lists the accepted CAs by sending their Subject Distinguished Names (DN) in much the same way Mutual TLS is implemented. The User Agent can now prompt the user to enter or approve credentials. In this example this would be the release of a certificate chain for the proposed "private_key_certchain_jwt" method and permission to use the corresponding private key to sign the token. The User Agent then sends its authentication data in the form of a "private_key_certchain_jwt" token to the Identity Provider, as specified in the solution concept. The Identity Provider verifies the token is correct, including the certificate chain it contains.

The Identity Provider now issues an access token to the User Agent which he can use as a bearer token to access the desired resource. For this purpose, the User Agent repeats the access procedure, which can now be successfully completed by downloading the AASX file. The User Agent can also use the token for access to the same API at a later date, without having to reauthenticate until either the token has expired or the rights for a specific operation require additional attributes in accordance with the access control specifications.

11.2 Format of a "private_key_certchain_jwt" token

For the implementation of the proposed concept, the "private_key_certchain_jwt" token must be defined as specified in Section 5.2.2. The token is based on the "private_key_jwt" token, which is supplemented by the required X.509 certificates. Since the use of certificate hierarchies of different participants is to be supported, an x5c element is included with the complete certificate chain. The certificate chain is built into the headers in accordance with the concept of a JSON Web Key (JWK), where the other elements, such as algorithm and key type, are also included, see Table 1.

Table 1: Header of "private_key_certchain_jwt" token[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 Description at https://kb.authlete.com/en/s/oauth-and-openid-connect/a/client-auth-private-key-jwt

The “client_assertion”, which must be digitally signed together with the header, contains the other parameters relating to the communication partners, see Table 2.

Table 2: Payload of ”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).

Finally, the signature is calculated in accordance with the following formula:

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

11.2.1 Sample tokens

An example of a token including the header and payload elements is shown below:

```
{
  "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"
}
{Digital signature}
```

The full token is printed out below and can be entered and analysed using copy&paste e.g. at https://jwt.io/. For the analysis of the embedded certificates, file:///C:\\Laufwerk_L\\AG3\\DP_SKI40_Downloadservice\\z.B.%20einer%20der%20online%20verfügbaren%20X.509 certificate decoders available online, for example, or the „openssl x509“ command line tool can be used. The three components: **Header**, **Payload** and **Signature** are separated by „.“ and highlighted in colour.

eyJhbGciOiJSUzI1NiIsImtpZCI6IkY3RUMwMTlCNDI5OTM4NTkzNzA3MUJGM0VFOUY1RTcwRDQ3RUEwRTAiLCJ4NXQiOiI5LXdCbTBLWk9GazNCeHZ6N3A5ZWNOU
i1vT0EiLCJ0eXAiOiJKV1QiLCJ4NWMiOlsiTUlJRGZEQ0NBbVNnQXdJQkFnSUlUUTNiQkRnaGNxWXdEUVlKS29aSWh2Y05BUUVMQlFBd1R6RUxNQWtHQTFVRUJoTU
NSRVV4R0RBV0JnTlZCQW9URDFCb2IyVnVhWGdnUTI5dWRHRmpkREVOTUFzR0ExVUVDeE1FUTFSV1F6RVhNQlVHQTFVRUF4TU9VM1ZpSUVOQklFeGxkbVZzSURJd0h
oY05NakF3T1RJeElURXhOakF3V2hjTklqRXdPVEl4TVRFeE5qQXdXakNCaURFTE1Ba0dBMVVFQmhNQ1JFVXhKakFrQmdOVkJBb01IVkJvYjJWdWFYZ2dRMjllZEdG
amRDQkhiV0pJSUNZZ1EyOHVJRXRITVNRd0lnWURWUVFEREJ0TWRYUjZJRXJEcEc1cFkydGxJQ2gyYjI0Z1ZHVnpkQ0JEUVNreEt6QXBCZ2txaGtpRzl3MEJDUUVXS
Ed4cVlXVnVhV05yWlVCd2FHOWxibWw0WTI5dWRHRmpkQzVqYjIwd2dnRWlNQTBHQ1NxR1NJYjNEUUVCQVFVQUE0SUJEd0F3Z2dFS0FvSUJBUUN2ZUUzTUpFQ1grS1
FiWE9SQVl5eEtDTkNtKzRRTi9RaW1DOFFqUWZGV2gxVHI1TUdxaHE3TGdDVE9wVW9NV1JSbnlhUmVZQll0aUxOUVZEV256V3FWNWtYM0tvc2g5Sk5M2NrNDlLUHM
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
EVlIwUEJBUURBZ0VHTUJFR0NXQ0dTQUdHK0VJQkFRUUVBd0lBQnpBZUJnbGdoa2dCaHZoQ0FRMEVFUllQZUdOaElHTmxjblJwWm1sallYUmxNQTBHQ1NxR1NJYjNE
UUVCQ3dVQUE0SUJBUUNnbFp0VE5USys3S3JwZThFem0xUW9zcWNlaURRR3l1YnJLZlNjSlkzSklsZFhwTm0zNWQzaWE4akJSL0lnZVZoOTRzRUdDZmt1SEZOcEVKT
3ZWQ2p0Q2hSb2UyVi9DdE5ydzB6MDZDVGpRNnFPZlAwb3JHcGUrQ3JkM09sbzFtOFY2RVBUZ1FWazRSUzdSdjd0b3NjaVQwNjlGanA3TmdIa2dQdzBNWFBnTFpBRn
BhZUFsOTY3L1MrTlpLNjhIb2J5N3ZWRGtrZ1FoaEhaYlY0Rmc3OTNnWXdRK1MxT3J1eEQ4SUJ1ZHRaRmpDSElGKzFESFlzWFVuR3BrWldJODFGMXRmOUVHWVpOWkh
NWkJYaVBCa3p0K0VoTzZzb3RNWUJYdWx6VDBIR29yUDVFSXpHOWRMa0JFaGdMcUhDUnlnQldYeFloVDdCWEo3Um5NVWQwV0hOYSIsIk1JSURWakNDQWo2Z0F3SUJB
Z0lJUHN6c1NTS09ZcUl3RFFZSktvWklodmNOQVFFTEJRQXdNVEVUTUJFR0ExVUVDaE1LVUd4aGVXZHliM1Z1WkRFYU1CZ0dBMVVFQXhNUlRIVjBlaUJVWlhOMElGS
nZiM1FnUTBFd0hoY05VGt3T0RJek1UVXpNakF3V2hjTk1qa3dPREl6TVRVek1qQXdXakF4TVJNd0VRWURWUVFLRXdwUWJHRjVaM0p2ZFc1a01Sb3dHQVlEVlFRRE
V4Rk1kWFI2SUZSbGMzUWdVbTl2ZENCRFFUQ0NBU0l3RFFZSktvWklodmNOQVFFQkJRQURnZ0VQQURDQ0FRb0NnZ0VCQU1SK1Bwdjg2MGthWUJRSXFpQUMzK243QzA
0bEZ6SVlzSlY2N2xPeS9LVW9STmJBWSsxeFdsK281emNCL3hZcnBNc0RVNUtBb1BrSXhvWmN0cWJ0OStYblRyekN3WDExSG1LWjA2WjZka2hMYTRhS3RWak85dFFX
U1lOQzhIbXVmYW5yZEhxNVk5U21JWjRFQnhJYWJFaklHRGhCVGM4QkFiWkVnTVpXYnJRZ2k1cVc0bDErZHgrN0pNWXJ0ZUJSbG4rSUhLOGRsSitnakx6eVpSWkFXN
GhLSUFyRTh0dnJHZDdqd2tXV1FlbTVCaVFBcHlpV3JVQXltNTVGd3RUclZid2VsVDVOMDJQZ3YzREtjRDBJR3Rqd24ydGMzc2NPNmpmZDE4cW4vT2d3YnBQbnlxaU
w0NGpFbnhOS2xYTy9telpldkxPejk5V1kvOGFQa0Rua3RSVUNBd0VBQWFOeU1IQXdEd1lEVlIwVEFRSC9CQVV3QXdFQi96QWRCZ05WSFE0RUZnUVVBN0hXWWdnUkg
2dGZUOVBCeXVUdmJUbWNEWll3Q3dZRFZSMFBCQVFEQWdFR01CRUdDV0NHU0FHRytFSUJBUVFFQXdJQUJ6QWVCZ2xnaGtnQmh2aENBUTBFRVJZUGVHTmhJR05sY25S
cFptbGpZWFJsTUEwR0NTcUdTSWIzRFFFQkN3VUFBNElCQVFDR0cySjBSOGI2ZU1GRWlMeCtTbGFIbGlQclU5M3FRMGxTcmV5RlNGMDdBbG5zczYxZVZ1T2o2dzQ5N
EcrYXgrbGR6N2loc2JmWmV6eVloTUJONWlZSjFvbW15ZGU4VHNQSkhaWE1LVU54ZEdOSVM3S3BJSWJjNWtNNGQ2OFhWc0dFb2E1YjRUa2V2ZGI3TEZRRzVhVktYem
Zac2ZweEU1QzNlclAvVzdYTkJsYVhPK0VVemZTUmZTTjNOTWQ1K1VCeUVNVU9MdVZGeWFzTnRFaFRYa0hDOEdWcDllcHJ2Rk03dTJlNlBxd1pwbkxTSDhnVkYyOGh
yZEVGR0VmYTIwcjRQWmd3KzRvbkpZZzBqb3NJbDZyTWVpaXAxM3J5Z1lsaWs0czg0WUpPa2tPcGxCNkcwRHVBbzE0N1JUbExyN3B4Z2FnamlaLytvM2NBdnozaUl1
THoiXX0.eyJqdGkiOiIyM2M1NTRmNS02MGU3LTQ3YzItOTQ3Yy1kZmQ2ODNjMzU1ZGYiLCJzdWIiOiJjbGllbnQuand0IiwiaWF0IjoxNjAxMDM1MDMzLCJlbWFpb
CI6ImxqYWVuaWNrZUBwaG9lbml4Y29udGFjdC5jb20iLCJuYmYiOjE2MDEwMzUwMzMsImV4cCI6MTYwMTAzNTA5MywiaXNzIjoiY2xpZW50Lmp3dCIsImF1ZCI6Im
h0dHBzOi8vYWRtaW4tc2hlbGwtaW8uY29tOjUwMDAxL2Nvbm5lY3QvdG9rZW4ifQ.NDk2nfIiv4-
Jk1ZL7yL6rvNgPhU6J36YZsfr2COf1LIOf7wTwPJeZ9dUAXQ77NF4GV49PYS-Nx5NBfMoy5Qy4dg3eJqzllCw1VGasjbbVYpX-hx3o6fQA6P2R-
Xpq0vHNcbL0dlubh8DZvwSB6OtkRnEiOy3i_fTwR99ZLVBclFnQ-
uh2FdzpFl8MmSDzBaR5FcEhMbuyXIJyZYAKPn8noHzWsftNCQeAz477P2aLWW4zN0MDuPcjM8DQuQ1IBs0U8W4v4bW0gnr-
OGVrneHJ94nBgIBsjkUtLqV4WCitkWWHX3CERRvvUKPvG-ITvoOLwdUYHR54orCBqjf5nZBQQ

EDITORIAL TEAM

André Braunmandl, Federal Office for Information Security | Vanessa Bellinghausen, Federal Office for Information Security | 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, Federal Office for Information Security | 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 (Chair), PHOENIX CONTACT GmbH & Co. KG | Michael Jochem, Robert Bosch GmbH | Thomas Lantermann, Mitsubishi Electric B.V. | Jens Mehrfeld, Federal Office for Information Security | 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, ifak – Institute for Automation and Communication

This publication is a joint outcome of the work by the Working Groups on Security of Networked Systems and Reference Architectures, Standards and Standardisation (Plattform Industrie 4.0).

www.plattform-i40.de