

● iSHARE: an introduction

Getting an access token:
Certificates, JWTs and client assertions



iSHARE

There are four steps towards getting an access token



iSHARE

1

Certificates

- What is a certificate?
- What is .p12 and x5c?
- How does iSHARE use certificates?

JWTs

2

- What is a JWT?
- How do I create a JWT?
- How do I sign it with a certificate?

3

Client assertion

- What is a client_assertion?
- How does it use JWTs?
- How do I create a valid client_assertion?

Access token

4

- What is an access token?
- How does it use a client_assertion
- How do I use access tokens?



A certificate is a digital file used for *integrity* and *authenticity*



iSHARE

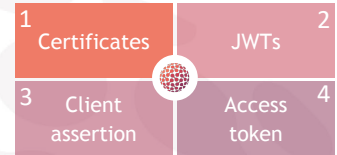
certificate

(public key infrastructure)



- Digital file that proves ownership of public key
- Certificate provides information on the owner
- Certificate is issued by a Certificate Authority
- Keys used to sign or encrypt other files

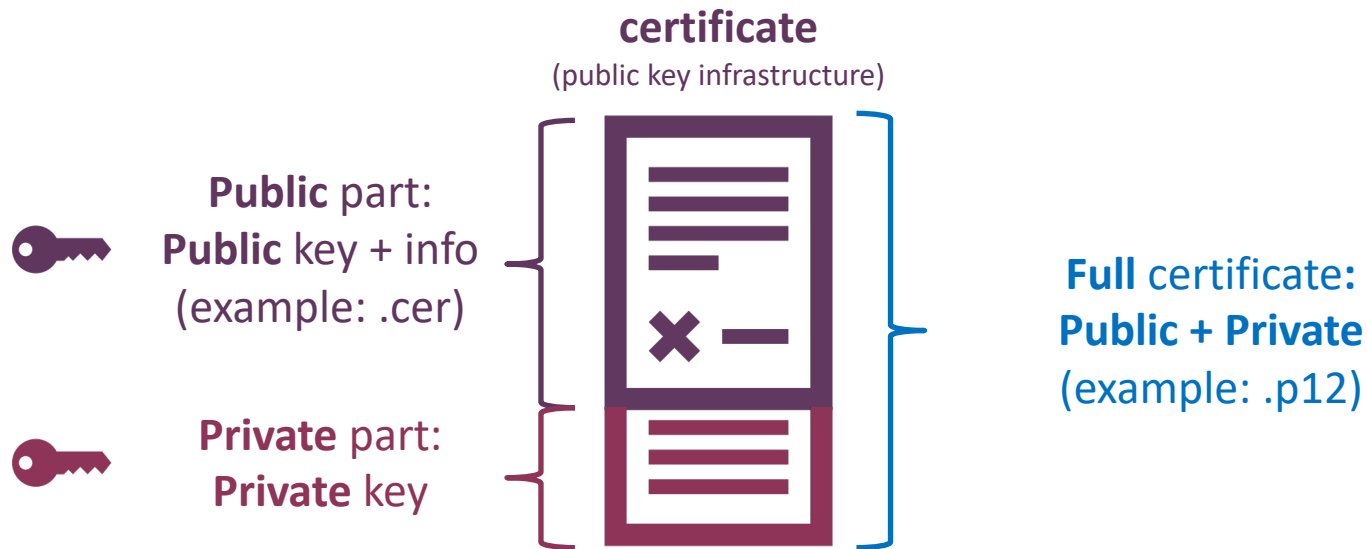
More info:
[Cheat Sheet](#)
or [Dev. Portal](#)



A certificate consists of a public and a private part



iSHARE



More info:
[Cheat Sheet](#)
or [Dev. Portal](#)

| | | | |
|---|------------------|--------------|---|
| 1 | Certificates | JWTs | 2 |
| 3 | Client assertion | Access token | 4 |

iSHARE uses PKI-certificates to verify identities

certificate

(public key infrastructure)



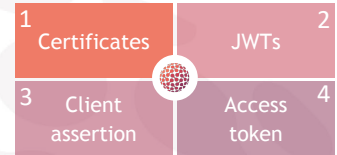
In *general*

- digital file used to sign or encrypt other files
- contains both a **private key** and **public key**
- part of **public-key infrastructure** (e.g. PKIoverheid or eIDAS)

In *iSHARE*

- **x5c** - value
 - String of letters representing public part of certificate
 - Used to *verify* signing and identity
- **.p12** file containing both **private and public part**
 - Used to sign (*integrity and authenticity*)
 - Password protected

More info:
[Cheat Sheet](#)
or [Dev. Portal](#)

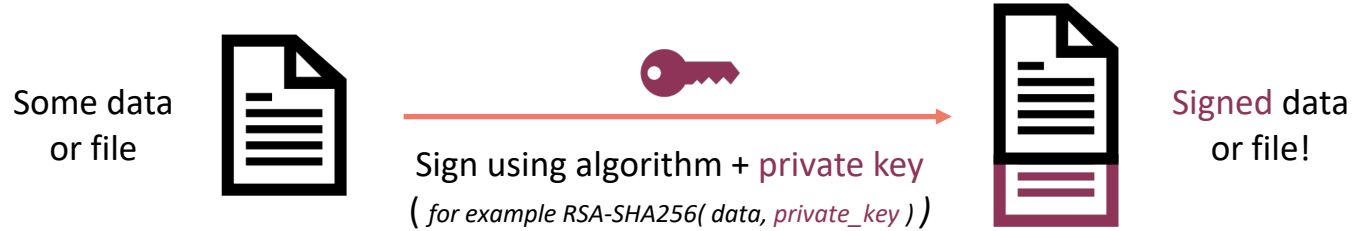


A private keys signs data

A public keys verifies this signature



iSHARE



More info:
[Cheat Sheet](#)
or [Dev. Portal](#)

| | | | |
|---|------------------|--------------|---|
| 1 | Certificates | JWTs | 2 |
| 3 | Client assertion | Access token | 4 |

JSON Web Tokens (JWTs) are base64-encoded JSON objects

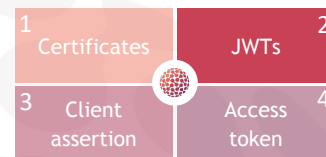


iSHARE

A basic JWT consists of three base64-encoded strings joined together by a dot ‘.’

| | INPUT | OUTPUT |
|-----------------------------|--|--------|
| 1. Header (JSON) | <pre>{ "alg": "RS256", "typ": "JWT" }</pre> | |
| 2. Payload (JSON) | <pre>{ "name": "iSHARE", "iat": 1516239022 }</pre> | |
| 3. Signature (RSASHA256) | | |

More info:
<https://jwt.io>
or [Dev. Portal](#)




JSON Web Tokens (JWTs) are base64-encoded JSON objects

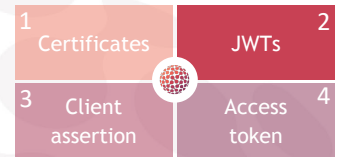


iSHARE

A basic JWT consists of three base64-encoded strings joined together by a dot “

| | INPUT | OUTPUT |
|-----------------------------|--|--|
| 1. Header (JSON) | <pre>{ "alg": "RS256", "typ": "JWT" }</pre> | <pre>eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9</pre> |
| 2. Payload (JSON) | <pre>{ "name": "iSHARE", "iat": 1516239022 }</pre> | <pre>eyJuwYw1lIjoiaVNIQVJFIiwiaWF0IjoxNTE2MjM5MDIyfQ</pre> |
| 3. Signature (RSASHA256) | <pre>RSASHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), private key )</pre> | <pre>NgQ-hpSnPKZ7ZuobA3rkqi4b0qpL2X9UXGGe_bapAum29V_wqhFBWBqE-BkjRE0c_xiiXC30AA1EC6pJICmXqa2GL6NuyT4x7-daPzTgCDXoFMD9vAEL5aSC_vhsP1UnpvNzrepT2YNqSv1RbNBSJNGUTmd-rMDCwvGdmALRON60uZSTPuHZY0Z2-yMZLzOgwjdN-9DhB_FnJALKNHLyB9meJ0d3GhaasBVQjZyiqL0rKLlv3ItJg3dXnca-U5JycVzQz02xTMUNUaxAs9UNCuEYFIE01NrXIbNpU5gN-ZBw3M7Vxp9ZJhS02dor0bQ1KpiWRaCB50xu8p1qWtGcg</pre> |

More info:
<https://jwt.io>
or [Dev. Portal](#)



In iSHARE, client assertions are JWTs with iSHARE specific Header and Payload



A client assertion is a JWT used to prove your identity to another party. To do so, iSHARE prescribes a specific format for the Header and Payload of the JWT.

HEADER + "." + **PAYLOAD**

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5c": [
    "MIIEgTCCAmg.....",
    "(long string)",
    ".....reK18+JkAjAJU="
  ]
}
```

x5c-value of your certificate
(in this case of party ABC Trucking)

```
{
  "iss": "EU.EORI.NL000000001",
  "sub": "EU.EORI.NL000000001",
  "jti": "df7ffc54cf148d15b0",
  "iat": 1556210430,
  "nbf": 1556210430,
  "exp": 1556210460,
  "aud": "EU.EORI.NL000000003"
}
```

- Your EORI nr. (here: ABC Trucking's EORI)
- Should equal "iss"
- Identifier of the JWT
- Time of issuing in seconds (UNIX time)
- Should equal "iat"
- Expiry time ("iat" + 30 seconds)
- Other party's EORI nr. (here: Warehouse 13's EORI)

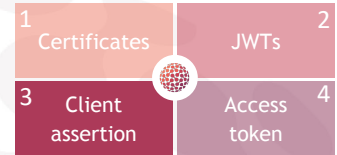
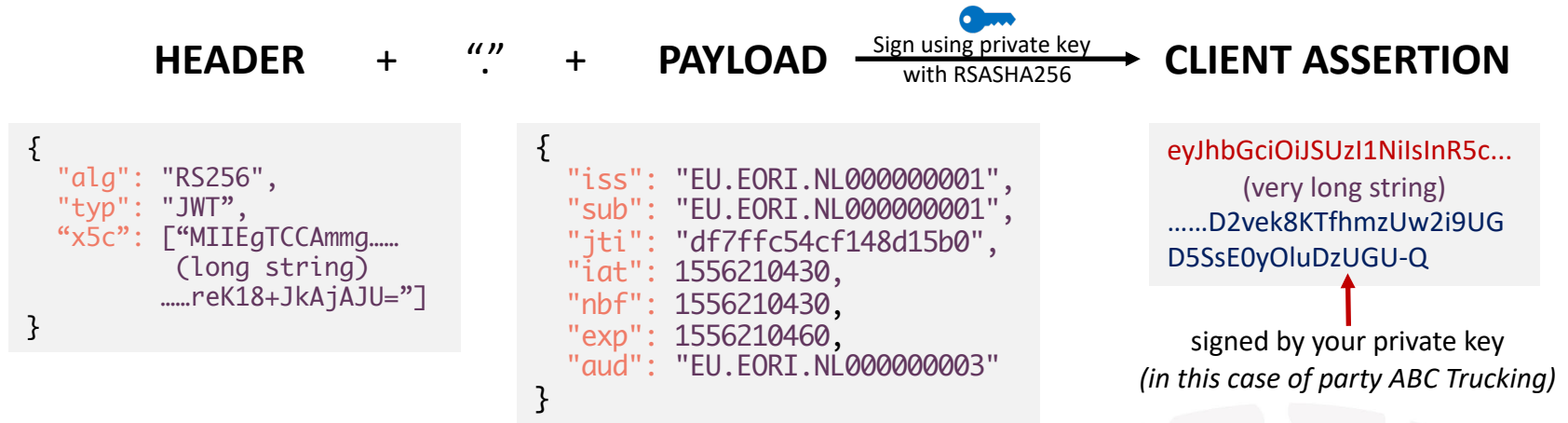


In iSHARE, client assertions are JWTs with iSHARE specific Header and Payload



iSHARE

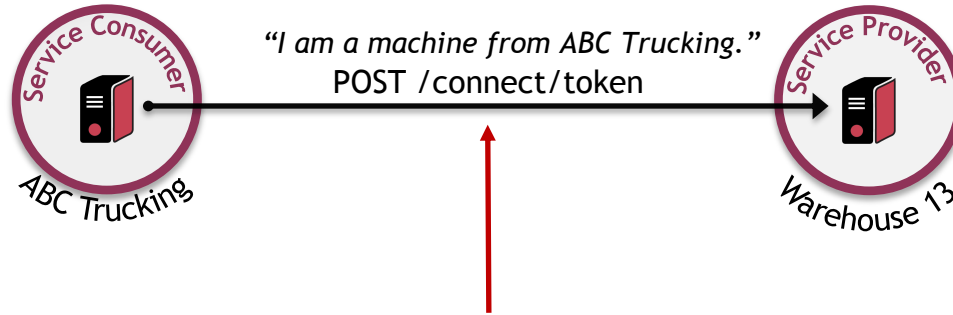
A client assertion is a JWT used to prove your identity to another party. To do so, iSHARE prescribes a specific format for the Header and Payload of the JWT.



iSHARE uses the client_assertion to get an access token from an iSHARE Party



Every iSHARE Party that exposes services has a /token endpoint, where you can get an access token for the services it exposes.



It is here that we use the client_assertion (JWT)

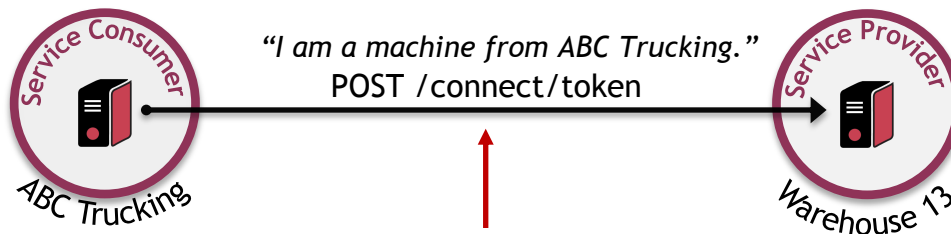


iSHARE uses the client_assertion to get an access token from an iSHARE Party



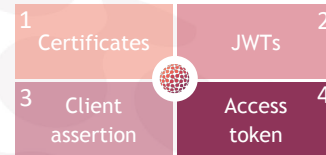
iSHARE

Every iSHARE Party that exposes services has a /token endpoint, where you can get an access token for the services it exposes.



Values in the body this request

| | | | |
|--------------------------|---|---|---|
| "grant_type": | "client_credentials", | ← | MUST be "client_credentials" |
| "scope": | "iSHARE", | ← | MUST be "iSHARE" |
| "client_id": | "EU.EORI.NL000000001", | ← | Your EORI nr. (here: ABC Trucking's EORI) |
| "client_assertion_type": | "urn:ietf:params:oauth:client-assertion-type:jwt-bearer", | ← | MUST be this string |
| "client_assertion": | "eyJhbGciOiJIUzI1NiIsInR5cGU6Ij....D2vek8KTF5SsE0y0LuDzUGU-Q" | ← | The JWT we created earlier |

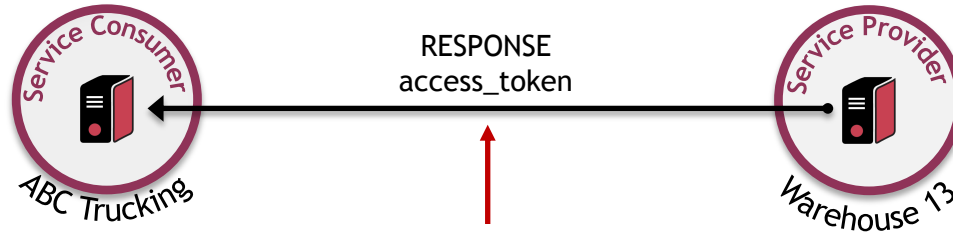


iSHARE uses the client_assertion to get an access token from an iSHARE Party



iSHARE

Every iSHARE Party that exposes services has a /token endpoint, where you can get an access token for the services it exposes.



Values in this response (if 200 OK)

```
{  
  "access_token": "eyJhbGciOi...(long string)...ahvEQ5w",  
  "expires_in": 3600,  
  "token_type": "Bearer"  
}
```

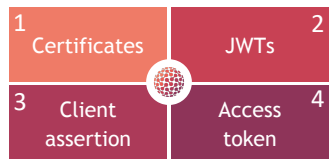
- ← The access token, for your further use
- ← The expiry time, in seconds
- ← The type of token (*should be "bearer"*)



**Uniform, simple
and controlled
way of sharing
logistical data**



iSHARE



● /authorize



iSHARE

Values in the body

```
"grant_type": "client_credentials",  
"scope": "openid",  
"client_id": "EU.EORI.NL000000001",  
"request": "eyJhbGciOiJSUzI1NiIsInR5c.....  
                (very long JWE)  
                .....D2vek8KTf5SsE0y0LuDzUGU-Q"
```

Objects in the JWE

```
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.  
0K0awDo13gRp2ojaHV7LFpZcgV7T6DVZKTyK0MTYUmKoTCVJRgckCL9kiMT  
03JGipsEdY3mx_etLbbWSrFr05kLzCsr4qkAq7YN7e9jwQRb23nfa6c9d-  
StnImGyFDSv04uVuxIp5Zms1gNxKKK2Da14B8S4rzVR1tdYwam_lDp5XnZA  
YpQdb76FdIKLamagfwX7XWRxv2322ivDxRfqNzo_tETKzpvLzfiwQyeyPGL  
BI056YJ7e0bdv0je1860ppamavo35UgoRdbYaBcoh9QcfylQr66oc6vFWXR  
cZ_ZT2LawVCWTIy3brGPi 6UklfCpIMfIj7iGdXKHgz.  
48V1_Al h6IIS04U3h  
5eym8TW_c8SuK01tJ3rpyIz0eDQz7TALvtu6UG9oMo4vpzs9tX_EFShS8iB  
7j6ji_SdiwkIr3ajwQzaBtQD_A.  
XFBOmYUZodetZdv1lFVSKQ
```

Signature verified by x5c from header

✔ Signature Verified

Decrypted ciphertext is JWS

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.  
SfLKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c
```