CEF Digital
Connecting Europe

# Working group meeting #4

## Future of eDelivery (REST API profile and Blockchain under ISA² action)

16 December 2020
14.00 – 16.45 CET

# Agenda

| Time | Items | Speakers |
|---|---|---|
| 14.00 - 14.10 | Welcome and introduction | Maya Madrid (CEF eDelivery Business Owner, DG CNECT H4) |
| 14.10 - 15.30 | Update on REST API profile:<br>• Presentation of early work on REST API profile<br>• Presentation of pilot architecture<br>• Timeline and next steps | Bogdan Dumitriu, Jerry Dimitriou, Vlad Veduta (CEF eDelivery Technical team, DIGIT D3) |
| 15.30 - 15.50 | Update on JRC's work on API guidelines for government<br>• Presentation of deliverable on API management & discoverability<br>• Summary of public sector stakeholder engagement activities | |
| 15.50 - 16.00 | *Break (10 mins)* | |
| 16.00 - 16.20 | Q&A on REST API profile | Monica Posada, Lorenzino Vaccari (JRC B6) |
| 16.20 – 16.40 | Update on integration with CEF EBSI (blockchain):<br>• Update on functional specifications<br>• Timeline and next steps | Bogdan Dumitriu, Vlad Veduta (CEF eDelivery Technical team, DIGIT D3) |

# Future of CEF eDelivery and API guidelines for government

Overview of actions and next steps

2020 ISA² Innovative Public Services (IPS) action

**Action implementers** | **DIGIT + JRC (oversight by DG CONNECT)**

| | Objective | What was done so far? | What we will be doing next? |
|---|---|---|---|
| **API guidelines for government** | • Identification of technical, legal and organizational essentials that ensure usability, stability and continuity of API-enabled digital solutions in governments. | • **Delivery of interim reports** describing API discoverability solutions and guidelines to manage API life-cycles in governments | • **Delivery of interim reports** describing API solutions to manage security and solutions to manage privacy aspects: security (eIDAS) + traceability (GDPR) |
| **REST API profile** | • Definition of a **REST-based profile** as a candidate profile for future inclusion in CEF eDelivery, in order to support new patterns of data access and data sharing | • Collection of **use cases** from WG members<br>• **Scoping Document** and **consultations** with WG members<br>• Ongoing work to **define the profile** for eDelivery REST API and **PoC** | • **Completion of the profile** for eDelivery REST API and **consultations** with WG members<br>• **Completion of the PoC** piloting the profile |
| **Integration with CEF EBSI (blockchain)** | • Piloting the use by CEF eDelivery of **blockchain services** offered by the CEF EBSI building block | • Collection of **suggestions for functionalities** from WG members<br>• Presentation of **proposed functionalities** and **draft version of functional specifications** | • **Complete the** functional specifications for the eDelivery blockchain pilot and **consultations** with WG members<br>• **Implementation** of features in Domibus (as pilot) |

# Update on REST API profile:

- Presentation of early work on REST API profile
- Presentation of pilot architecture
- Timeline and next steps

Bogdan Dumitriu, Jerry Dimitriou, Vlad Veduta - CEF eDelivery Technical team, DIGIT D3

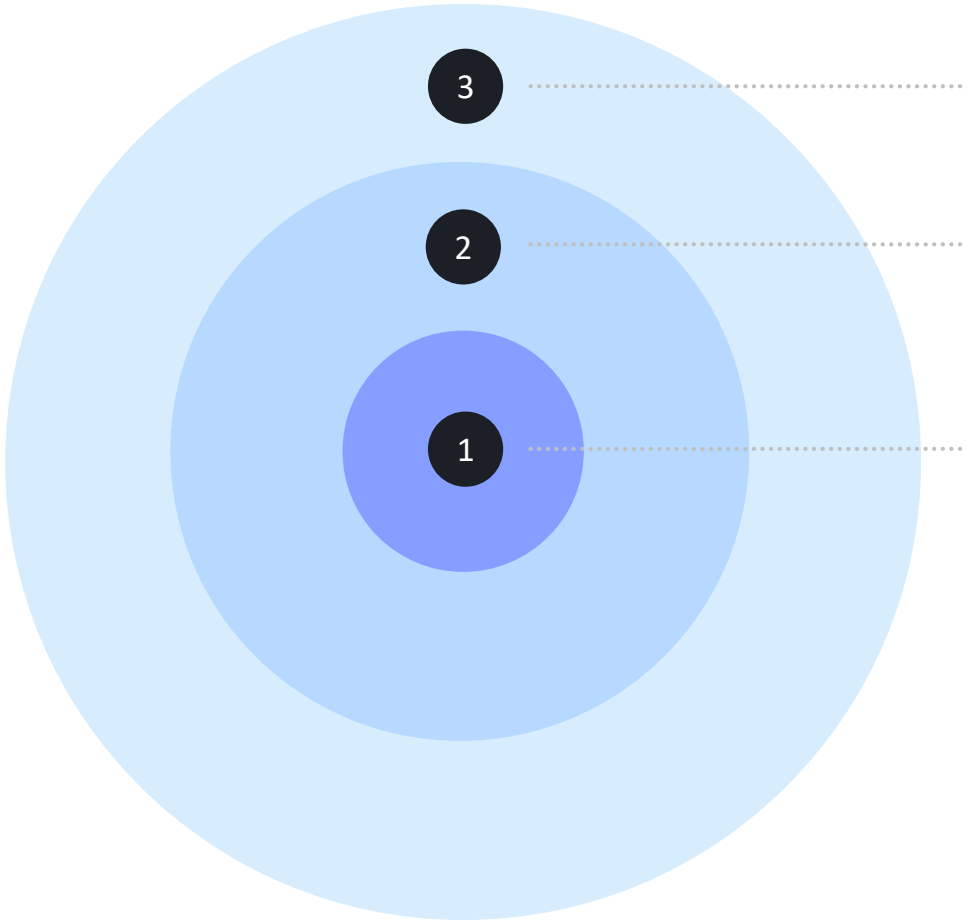
# Comments/feedback from stakeholders on Scoping document (3)

| Author | Key Comments | Reaction (by project team) |
|---|---|---|
| Nordic Institute for Interoperability Solutions (NIIS) | • Ideally, the profile should **not require adaptation** when used in different message/data exchange networks.<br>• The profile should define a framework for enabling the use of **different security levels**.<br>• How to carry the **client's identity**, how to facilitate its verification, how to facilitate the transformation of identities between different schemes.<br>• Avoid prescribing a specific format (JSON, XML) for the payload and – in general – remain **payload agnostic**. Use HTTP headers where possible.<br>• Recommendation to analyse 'Signing HTTP Messages' draft for **message signing**.<br>• Recommendation to support **all of 2-, 3- and 4-corner models**. | • Most suggestions adopted either as 'will implement' or 'will analyse'.<br>• Clarifications in the section 'Identity' for intended authentication scope (end-user, not client).<br>• 'Signing HTTP Messages' included in the list of standards to analyse.<br>• Full support for 4-corner model is not in scope. |
| Italian Digital Transformation Department | • Certain **protocols should not be specified** as part of the profile (e.g., IP, TCP).<br>• Suggestion to reconsider definition of "**light context**" as "relation between the overall resources, the size of the workload and business goals".<br>• The profile should define a framework for enabling the use of **different security levels**.<br>• Avoid prescribing a specific format (JSON, XML) for the payload and – in general – remain **payload agnostic**. Use HTTP features where possible.<br>• Recommendation to analyse 'Signing HTTP Messages' and 'Signed HTTP Exchanges' drafts for **message signing**. | • Removed protocols that should not be part of the REST API profile.<br>• "Light context" meant to capture specific scenarios that should be supported, not be limiting.<br>• Now stating that REST API profile should be designed to enable the use of different security levels instead of imposing a single, standard one.<br>• Now stating that proposed solutions will avoid constraining the payload to the largest extent possible.<br>• RFC 7807 included in the list of standards to analyse.<br>• Included suggestion to analyse message-signing approaches in addition to TLS.<br>• 'Signing HTTP Messages' and 'Signed HTTP Exchanges' included in the list of standards to analyse. |
| ETSI Electronic Signatures and Infrastructures Technical Committee (ETSI ESI TC) | • Is eDelivery a fully-fledged trust service as eIDAS **ERDS** or something broader?<br>• No coverage of **Evidence** generated by ERDS providers (ETSI EN 319 522-2)<br>• Recommendation to analyse XAdES and JAdES for **message signing**. | • eDelivery aims for alignment with ERDS, but meant to cover broader scope. Relationship between REST API profile and ERDS to be defined.<br>• Unclear if the REST API profile can/should be tightly linked with ERDS: new section on "Use in (Q)ERDS context"<br>• Clarifications in the section 'Identity' for intended authentication scope (end-user, not client).<br>• ETSI EN 319 522-2, XAdES and JAdES included in the list of standards to analyse. |

Please consult https://ec.europa.eu/cefdigital/wiki/x/hwZFEw for detailed comment resolutions and https://ec.europa.eu/cefdigital/wiki/x/fgZFEw for final version of Scoping Document.

## REST API profile

# Structure of the profile




**Messaging API Specification**

**Level 2: API Documentation**

**Level 1: API Core Profile**

### Profile enhancement

- ✓ High security enhancement
- ✓ Discoverability enhancement

## REST API profile

# Core profile

### Authentication and Authorization

**Key aspect**

- ✓ Choice on how authorization must be implemented (i.e. using OAuth 2.0)
- ✓ Specifications profiled according to security and interoperability standards/requirements

**OAuth 2.0 profile**

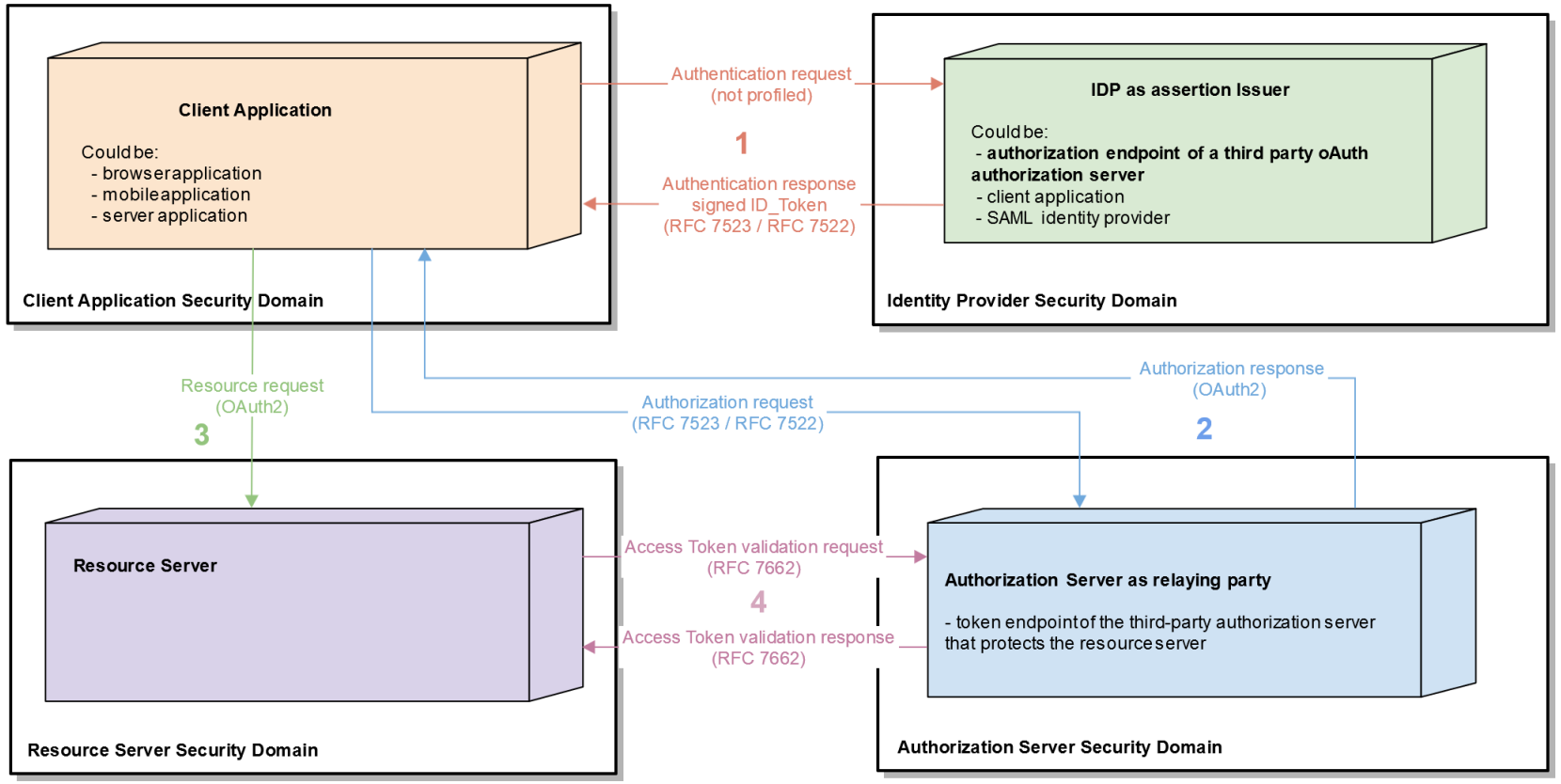
1. Overview of **actors**:
   - The Resource Owner (RO)
   - The Resource Server (RS)
   - The Client
   - The Authorisation Server (AS)
   - The Identity Provider (IdP)
2. OAuth 2.0 for **delegated authorisation** process/model
3. OpenID Connect **for authentication** process/model
4. Profiled the following aspects: **token format** (for enhanced identity assurance) and **authorisation grants** (i.e. flows)
5. **Different topologies** of authorisation and related restrictions

## REST API profile

# Core profile

## Authentication and Authorization – Service Topologies

External Authorization Server with external Identity Provider



## REST API profile

# Core profile

### Security

#### Scope

- ✓ Security as part of communication between the client and the resource server
- ✓ 3 levels of security:

1. **Transport**

2. **Message**

3. **Payload** (algorithms following recent security recommendations)

#### Design goals

- ✓ Confidentiality
- ✓ Integrity
- ✓ Audit trail
- ✓ Non-repudiation of the server-provided information
- ✓ Support lightweight context (e.g. easy configuration on the client side)

#### Non goals

- ✓ Message/Payload level encryption
- ✓ Client trust model (i.e. PKI)

## REST API profile

# Core profile

### Security analysis

- ✓ TLS 1.2 mandatory / TLS 1.3 recommended
  - TLS 1.3 mandatory in high-security profile enhancement
- ✓ Payload security using ETSI JAdES, based on JWS (IETF standard)
- ✓ No established standard for message integrity
  - Use of ETSI JAdES HttpHeaders Mechanism

Analysed and evaluated:

- Signed HTTP Messages Internet-Draft (IETF draft)
- Signed HTTP Exchanges (Google draft)
- JAdES (ETSI draft, extending the JWS standard)
- JSON Web Signature Profile for Open Banking (led by OBE and ETSI, domain standard)

REST API profile

# Core profile

## Lifecycle management

### Versioning

- ✓ Mandatory use of **semantic versioning** including
  - Definition of backwards compatible/incompatible changes
  - Specific rules for significant changes
  - Wildcard rule to allow escalation of the impact of a change from minor to major
- ✓ Choice of using **URL versioning** instead of MediaType versioning (considered safer)
- ✓ Include only **major version number** in URL

## REST API profile

# Core profile

### Lifecycle management

#### Deprecation and sunset of APIs

- ✓ Specific HTTP headers (Deprecation Response Header Internet-Draft and Sunset HTTP Response Header)
- ✓ OpenAPI specification property extension for capturing lifecycle events
- ✓ Property extension

```
OpenAPI:
  ...
  info:
    ...
    x-edel-lifecycle:
        maturity: "deprecated"
        deprecated_at: 2020-12-01
        sunset_at: 2021-01-01
```

## REST API profile

# Core profile

### Common Semantics

1. **Common Payload Representations**
   - Promote semantic interoperability with semantics used in common repositories, such as ISA² Core Vocabularies and Schema.org
   - OpenAPI mandates the use of JSON schemas
   - Selected payload representations are provided as JSON-LD format
   - Open question on how to move from JSON-LD to JSON

## REST API profile

# Core profile

### Common Semantics

2. **Common Semantics on Verbs and Status Codes**
   - Profile use of HTTP verbs
     - When to use POST vs When to use PUT
   - Profile use of HTTP Status Codes (i.e. constrain via additional specifications)
     - When 204 (No content) should be returned instead of 200 (OK)
3. **Error Messages**
   - Mandate of specific structure (Problem+JSON)
   - Use of Problem+JSON 'type' property to add additional error codes
   - Allows future creation of a registry with additional problem types


```
{
  "type": "https://rest.edelivery.ec.europa.eu/problems/resourceNotFound"
  "instance": "d9e35127-e9b1-4201-a211-2b52e52508df",
  "status": 404,
  "title": "Citizen not found",
  "detail": "No citizen with ID number 0206731645",
}
```

## REST API profile

# Core profile

### Documentation

#### Principles

- ✓ The API following the specification must be documented using the OpenAPI v3 standard, in the form of an OpenAPI Document
- ✓ The OpenAPI Document must be available under:
  - https://{domain}/{baseURL}/openapi.json
  - https://{domain}/{baseURL}/openapi.yaml
  - https://{domain}/{baseURL}/v{version.major}/openapi.json
  - https://{domain}/{baseURL}/v{version.major}/openapi.yaml
- ✓ The API Documentation section will provide an OpenAPI document containing a **reference template** (covering all the entities defined as part of Section A) to be used by the implementer
- ✓ The API Documentation section will also define the **possible OpenAPI specification extensions** (e.g. x-edel-lifecycle)

## REST API profile

# Core profile

### Discoverability

- Provide the proper mechanisms to become discoverable both in terms of its structure and operations
- To facilitate Discoverability the API MUST Have:
  - A complete OpenAPI v3 document accessible under its base URL
  - The OpenAPI Document MUST contain information on the **`servers`** property, pointing to all the known deployed instances
  - Use the additional OpenAPI Info Attribute extensions as defined in the Specification that can be used as metadata by repositories (eg. **`info.x-edel-publisher`**)

# Objectives of the pilot

- Test and validate the draft REST API profile
- Create a **feedback loop** during the development of the profile, ensuring its feasibility and completeness
- Demonstrate an **API specification** conformant to the REST API profiled by the project (the REST API for DSD)
- Demonstrate a client for the **above API** (the Broker)
- Demonstrate an **implementation of the messaging API** provided by the REST API profile by the project (the Broker)
- Demonstrate a client for the **messaging API instantiation** (the JavaScript client)

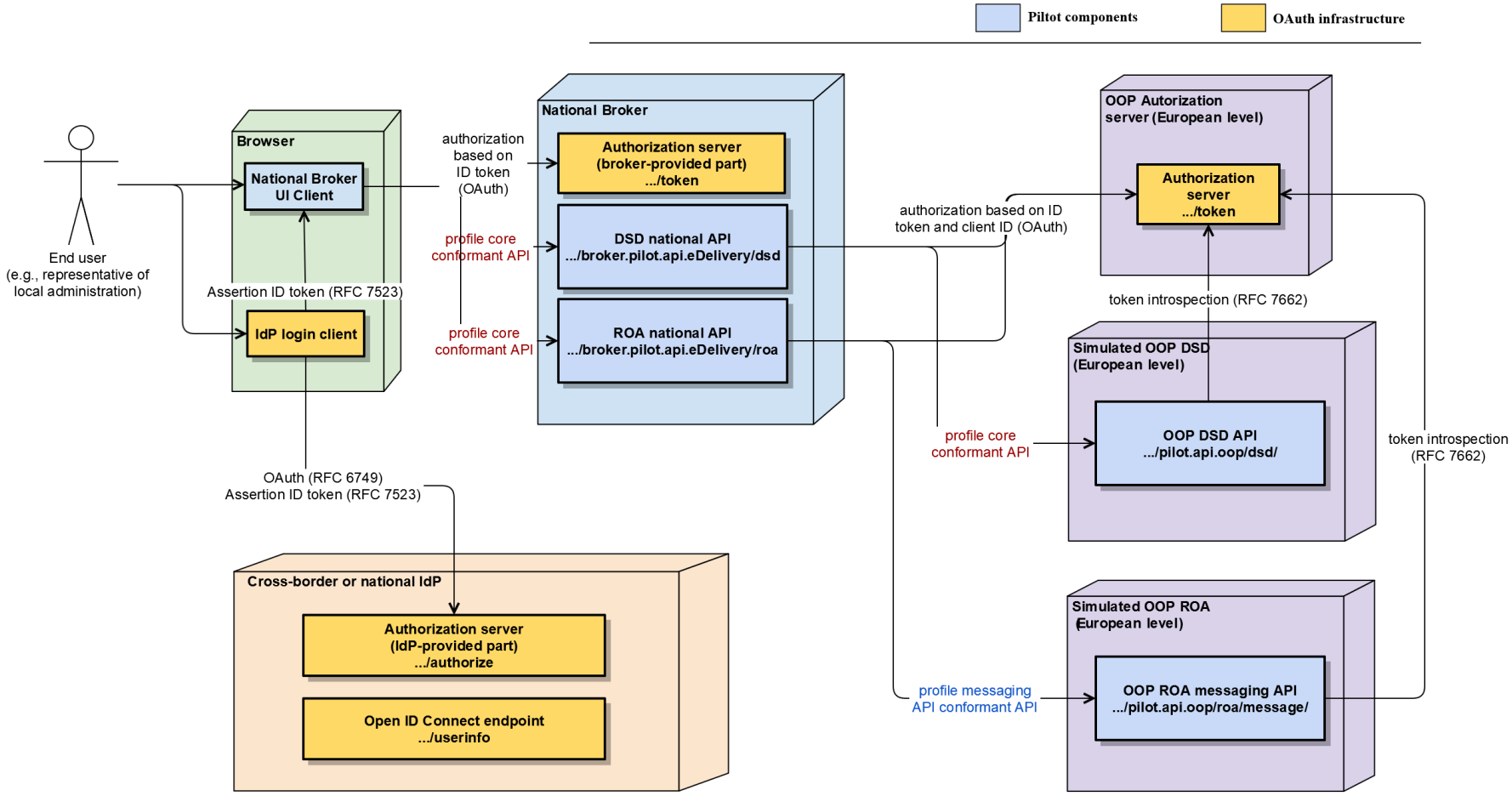
# Pilot architecture overview

## OAuth topologies:

- National Broker - Internal AS with external IdP
- DSD and ROA - External AS with external IdP

## REST API:

- Broker API conformant with the API Core Profile
- ROA API conformant with Messaging API specifications
- DSD API conformant with the API Core Profile



# Functionality of the pilot

**Story 01 – End user authenticated by a central IdP (EU Login) before being authorized to update any of the (simulated) OOP registries**

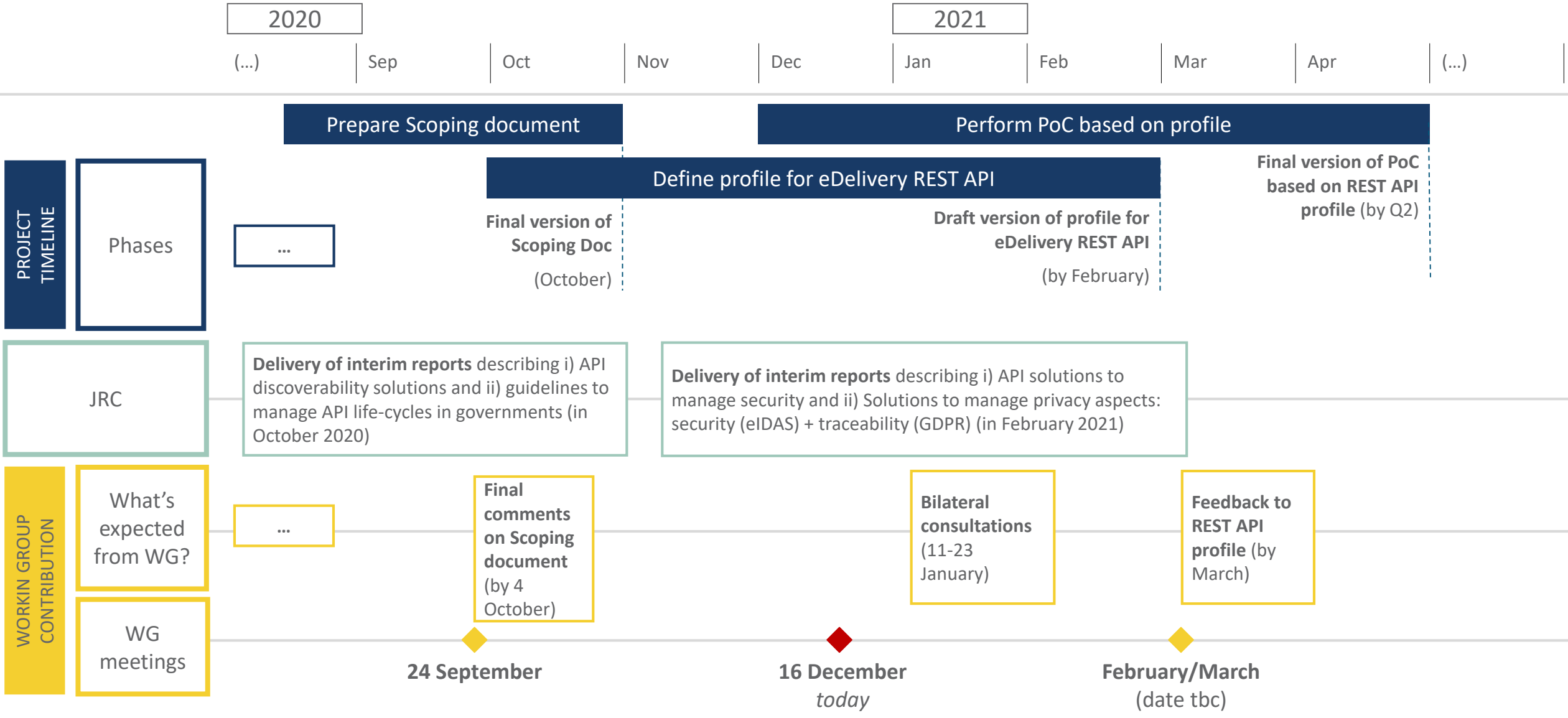
- End user accesses the login page
- End user is redirected to log in when accessing the UI application
- End user is not asked to log in if there is already a valid browser session

**Story 02 – End user uses the UI to update the (simulated) OOP registries**

- End user selects the organization (s)he represents
- End user creates a request to update a record in ROA/DSD
- End user creates a request to create a record in ROA/DSD
- End user sees the status of its requests
- The Broker application forwards the request to the targeted registry

# REST API profile for eDelivery

Indicative timeline and next steps



# Update on JRC’s work on API guidelines for government

- Presentation of deliverable on API management & discoverability
- Summary of public sector stakeholder engagement activities

Monica Posada - JRC B6

# Q&A – REST API profile

# Update on Integration with CEF EBSI (blockchain):

- Update on functional specifications
- Timeline and next steps

Bogdan Dumitriu, Vlad Veduta - CEF eDelivery Technical team, DIGIT D3

# Scenarios analysed for the eDelivery – EBSI pilot

1. Timestamp the WS-Security signature of the AS4 message (*EBSI on-chain*)
   Validate the timestamp on the receiving side

~~2.~~ Collect transaction data from the APs in an eDelivery network (*EBSI off-chain*)
   Query & aggregate transaction data to generate statistics

2. Use the EBSI Distributed Identity Registry as *on-chain (white)list* for authenticating/authorising participants in an eDelivery network

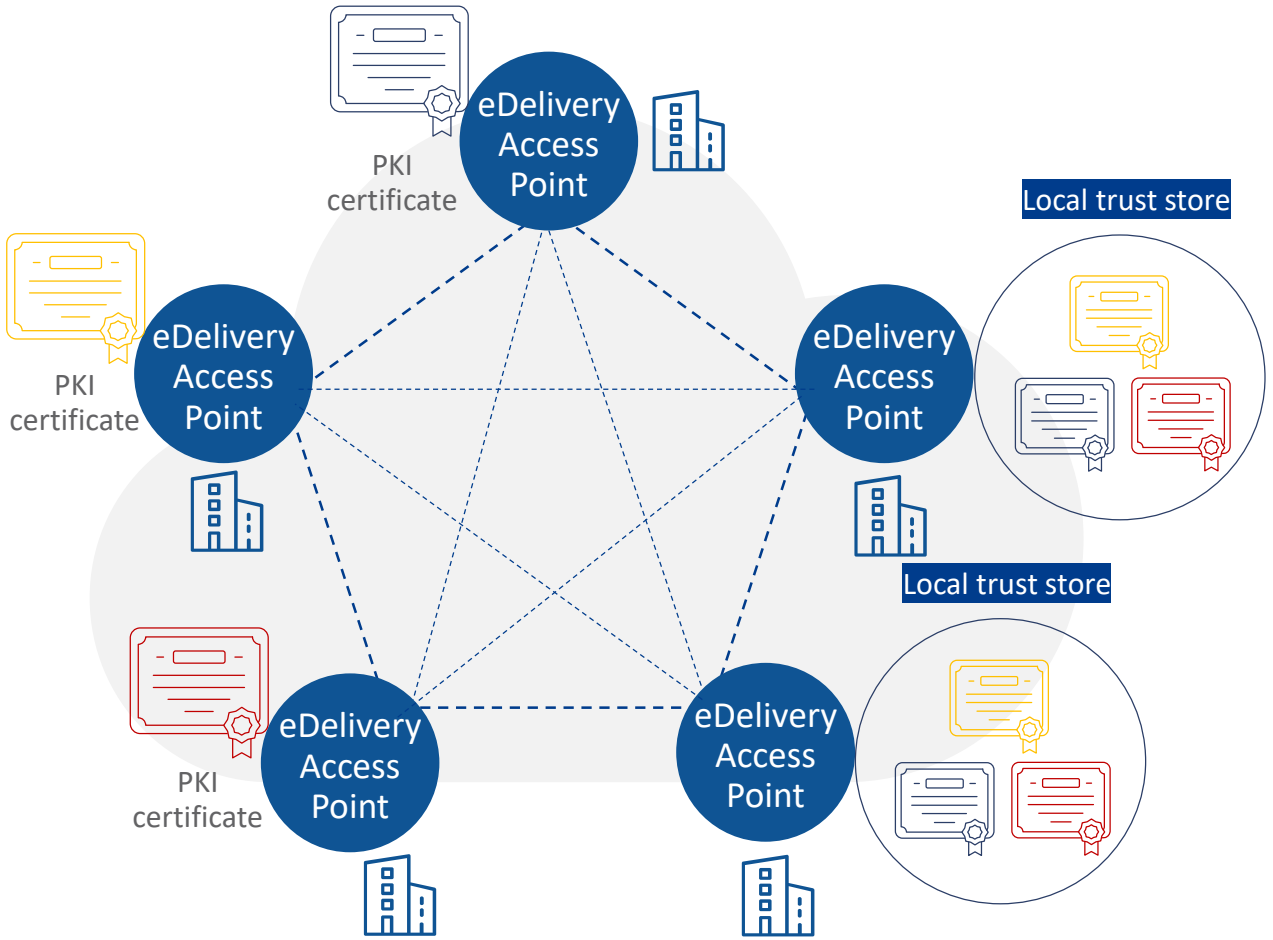
# Scenario 2 (new)

Use the EBSI Distributed Identity Registry as on-chain (white)list for authenticating/authorising participants in an eDelivery network

# eDelivery authentication/authorisation – standard scenario

**Authentication** of sending parties based on local trust store

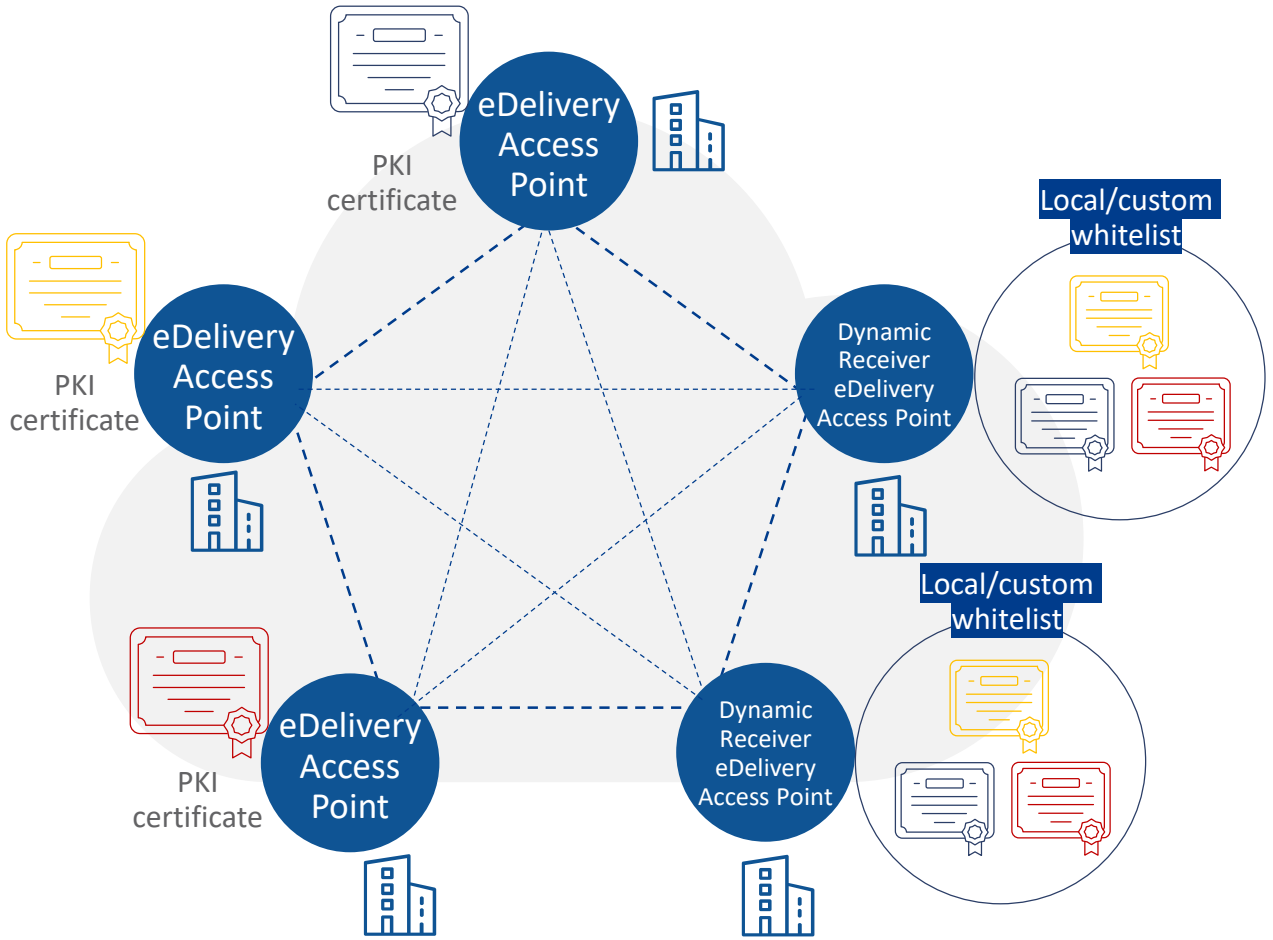
**Authorisation** of sending parties based on (local) PMode configuration



# eDelivery authentication/authorisation – dynamic receiver

**Authentication** of sending parties based domain rules (PKI, policies) + match From/PartyId – certificate fields

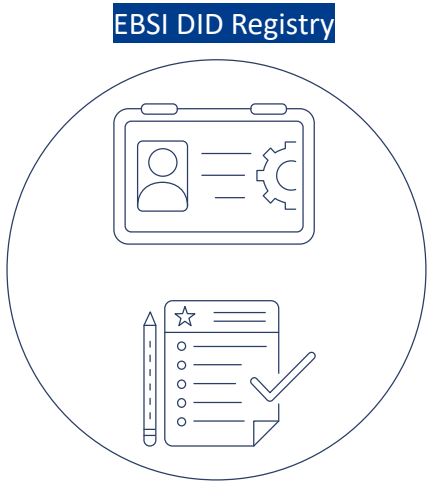
**Authorisation** of sending parties may use a whitelist



# EBSI DID Registry as an eDelivery (white)list

The **EBSI DID Registry** is suited to be used as a **(white)list:**

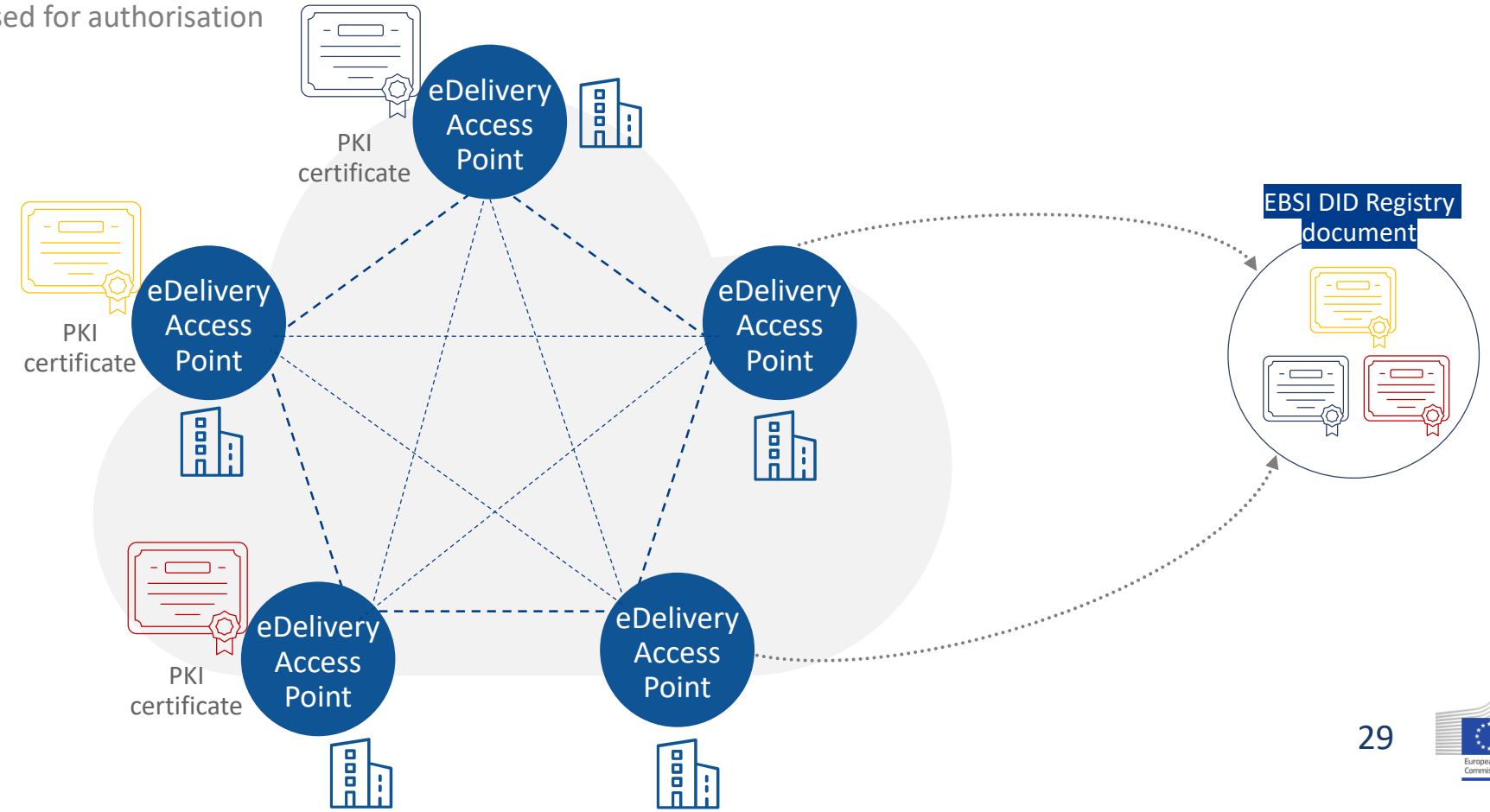
- It can be interrogated via a **public API**
- It is designed to **store and manage lists** of public key hashes
- It is secure since it is **stored on-chain**, and hence it is **immutable**
- It is editable in a **controlled** and **traceable way**



# EBSI DID Registry as an eDelivery authentication list

**EBSI DID registry** as a **certificate list** in a standard eDelivery scenario:

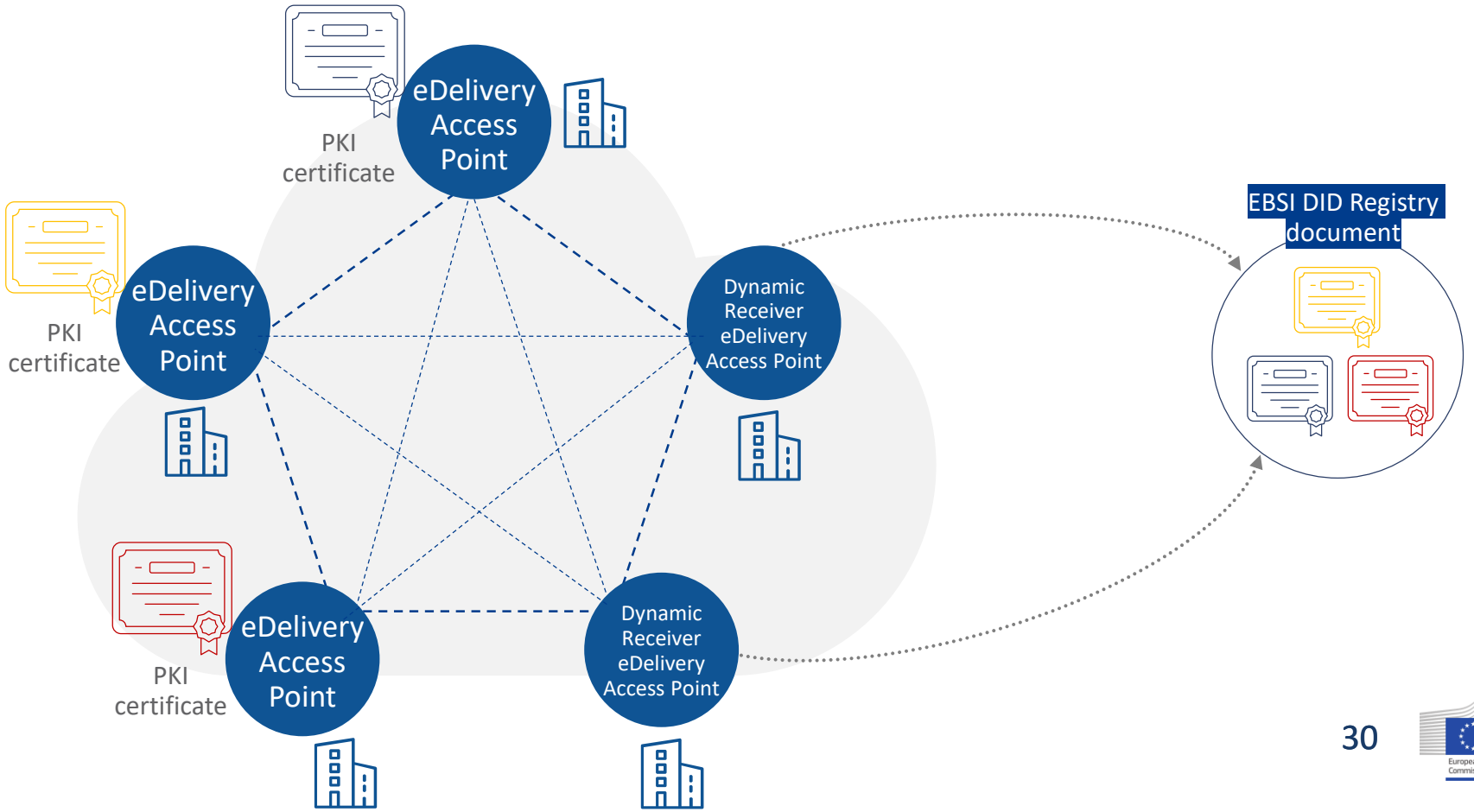
- The registry will store one DID document for **each domain**
- The document will contain the **list of the hashes** of the certificate + From/PartyId of authorised parties
  - Ensuring authentication only
- (Local) PMode configuration used for authorisation



# EBSI DID Registry as an eDelivery whitelist

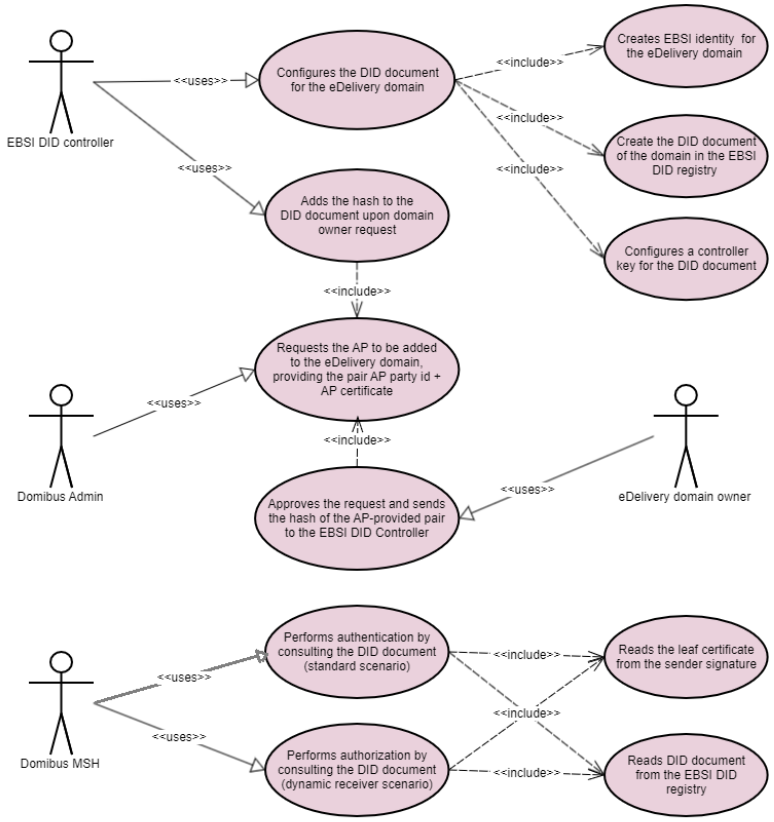
**EBSI DID registry** as a **whitelist** in a dynamic receiver eDelivery scenario:

- The registry will store one DID document for **each domain**
- The document will contain the **list of the hashes** of the certificate + From/PartyId of authorised parties
  - Ensuring authentication and authorisation



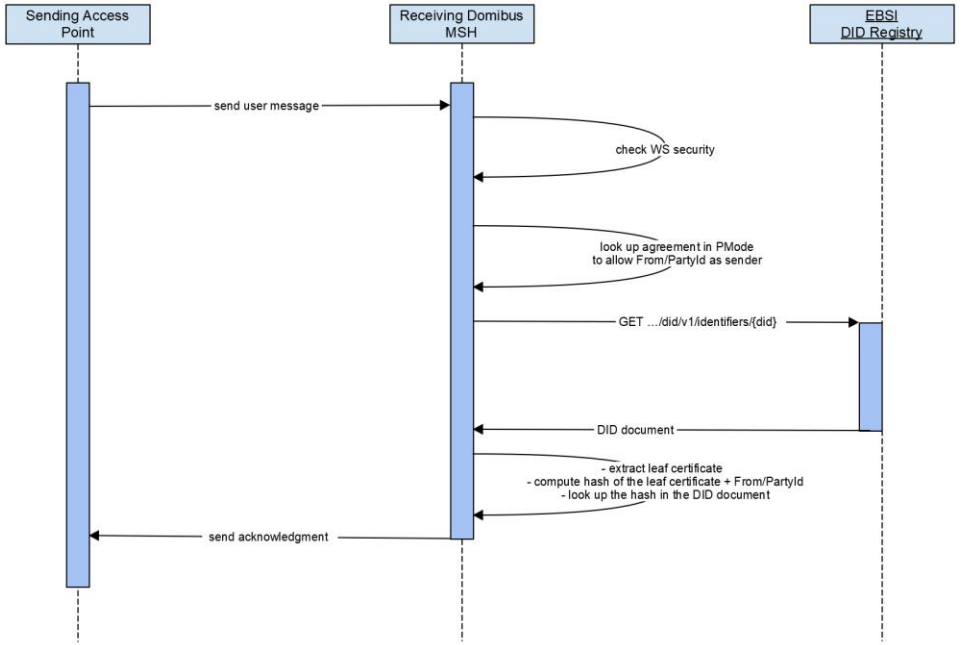
# Story 4: Using the EBSI DID registry for authentication/authorisation

1. EBSI DID Controller **configures** the DID document for the eDelivery domain
2. Domibus Admin **requests** the AP to be added to the eDelivery domain, providing the pair AP party id + AP certificate
3. eDelivery Domain Owner **approves** the request and **sends the hash** of the AP-provided pair to the EBSI DID Controller
4. EBSI DID Controller **adds the hash** to the DID document upon domain owner request
5. MSH **performs authentication** by consulting the DID document (standard scenario)
6. MSH **performs authorization** by consulting the DID document (dynamic receiver scenario).

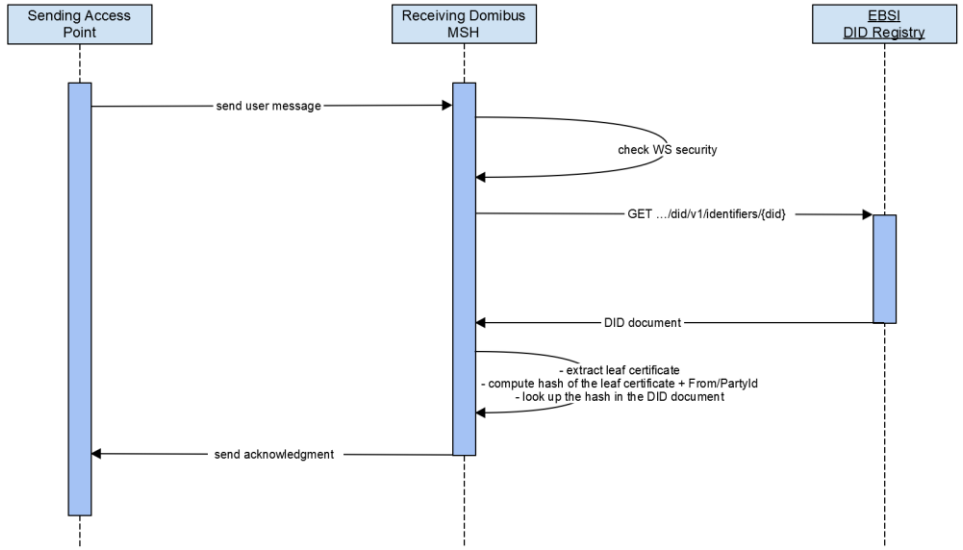


## MSH performs authorisation by consulting the EBSI DID Registry document

# Process for authorisation



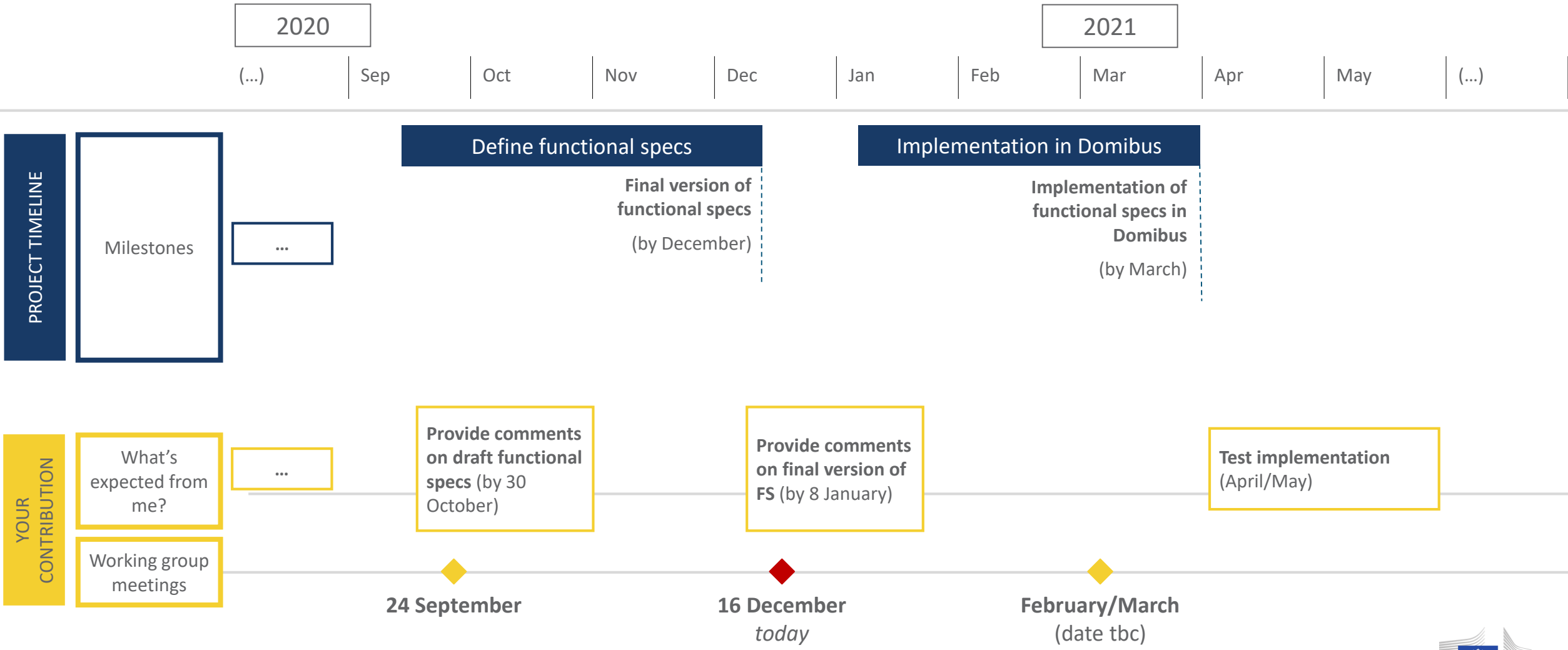

**Authentication via DID document, authorisation via (local) PMode configuration**



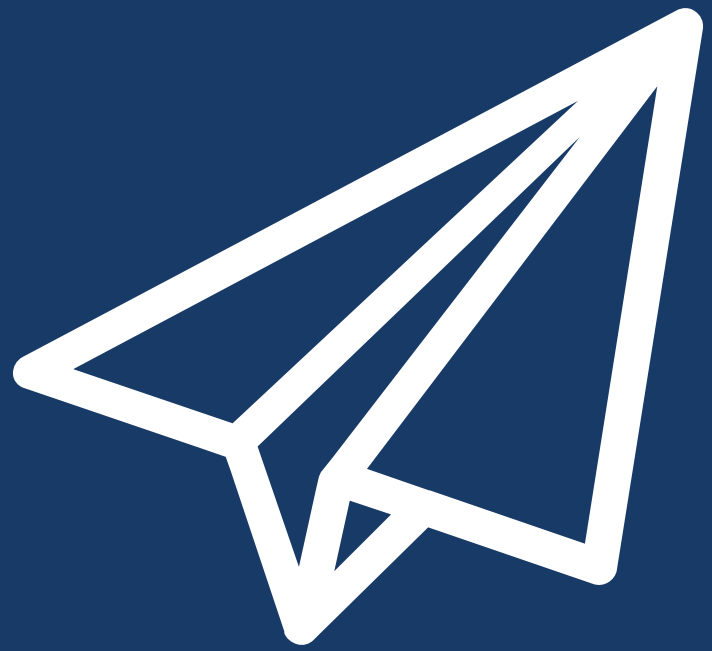

**Authentication & authorisation via DID document**

# Functional specifications for eDelivery / Blockchain integration

Indicative timeline



# Thank you!



ec.europa.eu/cefdigital

CEF-BUILDING-BLOCKS@ec.europa.eu