Blinded pseudonymisation REST service
Cookbook
Version 1.8

eHealth platform

Willebroekkaai 38 – 1000 Brussel

38, Quai de Willebroek – 1000 Bruxelles

Table of contents

To the attention of: “IT expert” willing to integrate this web service.

1. Document management

1.1 Document history

Version	Date	Author	Description of changes / remarks
1.0	04/04/2023	eHealth platform	Initial version
1.1	20/07/2023	eHealth platform	- Section 5 : Global process Add a step to convert an ECPoint to an identifier - Add a new section 6 : Process description - Add a new section 8: Annex for transitInfo - Minor updates in the document
1.2	22/08/2023	eHealth platform	Minor updates in the document
1.3	28/11/2023	eHealth platform	- Refine the document to make it language agnostic - Section 5 : add test cases and test data for ECPoint calculation - Add a new section 6.4: One input argument - Add a new annex 9: Processing input exceeding 32 bytes - Add a new annex **Error! Reference source not found.**: Shared secret key rotation recommendation - Improve the API description
1.4	23/01/2024	eHealth platform	- Updated Process in Section 5 : Convert an identifier to an Elliptic Curve Point (ECPoint) - Title rename for Annex 10: Key recommendations - New recommendations added in Annex 10
1.5	30/01/2024	eHealth platform	- Section 5 : - Update the standard parameters values for the elliptic curve **P-521** - **Convert an identifier to an Elliptic Curve Point (ECPoint):** Revise the calculation steps Include an example for better understanding - **Blind an ECPoint:** - Revise the calculation steps - Add examples for ECPoint blinding

1.6	07/02/2024	eHealth platform	- Remove the annex: *Key Recommendation* It has been replaced by external document *Pseudo rest Guidelines* – See above eHealth platform document references - Add a new annex: Pseudonym in transit: Conversion to one-line Argument - Update the annex: transitInfo Add new fields *iat* and exp in header (section 8.1.1 and 8.3.2)
1.7	21/01/2025	eHealth platform	- Minor updates to the document on naming conventions. - Annex: Object accessRules. Updated the description of the properties element.
1.8	01/04/2025	eHealth platform	In Section 5, under 'Convert an identifier to an Elliptic Curve Point (ECPoint)', update the table representation of the X-coordinate in Step g.

2. Introduction

2.1 Goal of the service

In the context of a project handling medical data, it is crucial to maintain the confidentiality of information.

This is where the Pseudonymisation service plays a role in preventing the link between personal and medical data, whether at rest in the database, in transit during network transport, or in use during processing.

The main goal of this service is to generate a unique pseudonym for the patient identity, ensuring that identity remains private and secure. Each pseudonym is associated with a business domain to further enhance data security. The business domain will be the responsibility of the domain owner.

To ensure maximum security, a different pseudonym is generated for each business domain. This means that pseudonym associated with one business domain cannot be linked to pseudonym associated with another business domain.

Access to the patient identity or resulting pseudonyms depending on the actor:

- The domain owner will only have access to the patient's pseudonym for the domain under his responsibility, and not the patient's identity
- The Pseudonymisation service will not have access to the patient's identity and the patient's pseudonym
- A client, such as a professional using a web application will only have access to the patient's identity and not the patient's pseudonym

The Pseudonymisation service offers several benefits. Firstly, it allows applications to meet their business needs without ever linking personal data with medical data. Moreover, applications using pseudonyms can interact with each other through a method of pseudonym conversion

2.2 Goal of the document

This document is not a development or programming guide for internal applications. Instead, it provides functional and technical information and allows an organization to integrate the Pseudonymisation REST service in its own custom application.

This document will provide all the necessary elements to get you started by explaining:

- the main concepts and principles
- technical information about calling the service

This information should allow (the IT department of) an organization to integrate and use the WS call. However, in order to interact in a smooth, homogeneous and risk controlled way with a maximum of partners, these partners must commit to comply with all the requirements described in this document.

In addition, our partners in the healthcare sector must also comply with the business rules of validation and integration of data within their own applications in order to minimize errors and incidents. In other words, technical and business requirements must be met in order to allow the integration and validation of the eHealth platform service in the client application.

2.3 eHealth platform document references

On the portal of the eHealth platform, you can find all the referenced documents.[1]. These versions, or any following ones, can be used for the eHealth platform service.

[1] ***www.ehealth.fgov.be/ehealthplatform***

ID	Title	Version	Date	Author
1	eHealth Services – Web Access	2.0	12/07/2018	eHealth platform
2	Identity & Authorization Management (I.AM) Mobile integration	1.7	03/04/2023	eHealth platform
3	eHealth Pseudo v1 REST - Guidelines and Recommendations	1.0	24/04/2024	eHealth platform

2.4 External document references

All documents can be found through the internet. They are available to the public, but not supported by the eHealth platform.

ID	Title	Source	Date	Author
1	RFC 7516	***https://www.rfc-editor.org/rfc/rfc7516***	17/03/2020	Michael Jones
2	RFC 7515	***https://www.rfc-editor.org/rfc/rfc7515***	29/07/2020	Michael Jones
3	Standard curve P-521 parameter	***https://neuromancer.sk/std/nist/P-521***	N/A	Jan Jancar
4	Standards for Efficient Cryptography Group (SECG), "SEC1: Elliptic Curve Cryptography", Version 2.0, May 2009.	https://www.secg.org/sec1-v2.pdf see ECPoint encoded as defined in section 2.3.3 of SEC1	21/05/2009	Daniel R. L. Brown

2.5 Service history

This chapter contains the list of changes made to the service with respect to the previous version.

Current version	Current release date	Changes
1.0.0	24/03/2023	First version
1.0.4	16/11/2023	Minor fixes
1.1.1	18/06/2024	Minor fixes

3. Support

3.1 Helpdesk eHealth platform

3.1.1 Certificates

In order to access the secured eHealth platform environment you have to obtain an eHealth platform certificate, used to identify the initiator of the request. In case you do not have one, please consult the chapter about the eHealth Certificates on the portal of the eHealth platform

- ***https://www.ehealth.fgov.be/ehealthplatform/nl/ehealth-certificaten***
- ***https://www.ehealth.fgov.be/ehealthplatform/fr/certificats-ehealth***

For technical issues regarding eHealth platform certificates

- Acceptance: ***acceptance-certificates@ehealth.fgov.be***
- Production: ***support@ehealth.fgov.be***

3.1.2 For issues in production

eHealth platform contact centre:

- Phone: 02 788 51 55 (on working days from 7 am till 8 pm)
- Mail: ***support@ehealth.fgov.be***
- *Contact Form :*
  https://www.ehealth.fgov.be/ehealthplatform/nl/contact (Dutch)
  https://www.ehealth.fgov.be/ehealthplatform/fr/contact (French)

3.1.3 For issues in acceptance

Integration-support@ehealth.fgov.be

3.1.4 For business issues

- regarding an existing project: the project manager in charge of the application or service
- regarding a new project or other business issues: ***info@ehealth.fgov.be***

3.2 Status

The website ***https://status.ehealth.fgov.be*** is the monitoring and information tool for the ICT functioning of the eHealth services that are partners of the Belgian eHealth system.

4. Requirements

To use the Pseudonymisation REST service, the following conditions must be met:

- The eHealth security team is responsible for guaranteeing the existence of a pseudonym domain and managing its configuration, including the setup of access matrix rules that defines the conditions for accessing pseudonymisation resources for a user.

- An IAM connect client must be configured across the environment to obtain an access token (JWT) required to call the REST service. In addition to an access token, you can obtain an ID token and a user info token from the access token. Unlike a standard token, the access token will not contain the citizen's identifier, the ID token will include the citizen's first name and last name and the user info token will only contain the pseudonymized citizen identifier. For more information about the tokens and how to obtained them, please refer to the eHealth platform document references (section 2.3 – ID 2).

Both conditions require that the onboarding documents are completed and the configurations (IAM Connect client and pseudo domain) have been performed. For more information on the onboarding process, please refer to the following eHealth portal page:

- **FR**: ***https://www.ehealth.fgov.be/ehealthplatform/fr/pseudonymisation-anonymisation***
- **NL**: ***https://www.ehealth.fgov.be/ehealthplatform/nl/pseudonimisering-anonimisering***

After configuring an IAM connect client and obtaining an access token (JWT), you will have access to the following resources depending on the role(s) obtained in the access token:

Role	Resource
pseudonymise	• GET /domains • GET /domains/{domainKey} • POST /domains/{domainKey}/pseudonymize • POST /domains/{domainKey}/pseudonymizeMultiple
identify	• GET /domains • GET /domains/{domainKey} • POST /domains/{domainKey}/identify • POST /domains/{domainKey}/identifyMultiple
convert	• GET /domains • GET /domains/{domainKey} • POST /domains/{fromDomainKey}/convertTo/identify/{toDomainKey} • POST /domains/{fromDomainKey}/convertMultipleTo/identify/{toDomainKey}

4.1 REST and JSON

SOAP and REST are two API styles that approach data transmission from a different point of view. SOAP is a standardized protocol sending messages using other protocols such as HTTP and SMTP.
As opposed to SOAP, REST is not a protocol but an architectural style. The REST architecture lays down a set of guidelines you need to follow if you want to provide a RESTful web service, for example, stateless existence and the use of HTTP status codes. As SOAP is an official protocol, it comes with higher complexity, it requires more bandwidth and resources that can lead to slower page load times. REST was created to address the problems of SOAP.

Therefore, it has a more flexible architecture. It allows different messaging formats, such as HTML, JSON, XML, and plain text, while SOAP only allows XML. REST is also a more lightweight architecture, so **RESTful web services have a better performance**.

The REST architecture allows API providers to deliver data in multiple formats such as plain text, HTML, XML, YAML, and JSON, which is one of its most loved features. Thanks to the increasing popularity of REST, the lightweight and human-readable JSON format has also quickly gained traction, as it is an easy-to-parse and lightweight data-interchange format. In spite of its name, JSON is completely language-agnostic, so it can be used with any programming language, not just JavaScript. JSON files consist of collections of name/value pairs and ordered lists of values that are universal data structures used by most programming languages. Therefore, JSON can be easily integrated with any language.

4.2 Endpoints

The last version of REST interface described with a JSON / Swagger API is available on the eHealth API Portal :

Environment	Endpoint
Acceptance	***https://portal-acpt.api.ehealth.fgov.be***
Production	***https://portal.api.ehealth.fgov.be***

4.3 Technical requirements

4.3.1 Tracing

To use this service, the request SHOULD contain the following two http header values
(see RFC **https://datatracker.ietf.org/doc/html/*rfc7231#section-5.5.3***):

1. **User-Agent**: information identifying the software product and underlying technical stack/platform.
   - Pattern: {company}/{package-name}/{version} {platform-company}/{platform-package-name}/{platform-package-version}
   - Regular expression for each subset (separated by a space) of the pattern: [[a-zA-Z0-9-\/]*\/[0-9a-zA-Z-_.]*
   - Examples:
     User-Agent: MyCompany/myProduct/62.310.4 eHealth/Technical/3.19.0
     User-Agent: Topaz-XXXX/123.23.X Taktik/freeconnector/XXXXX.XXX

2. **From:** email-address that can be used for emergency contact in case of an operational problem
   Examples:
   From: *info@mycompany.be*

5. Global process

The Pseudonymisation REST service is designed to generate a unique pseudonym for a given clear identifier, typically a citizen's identifier. This pseudonym belongs to a domain defined and configured specifically for a domain owner, ensuring that different partners generate different pseudonyms for the same citizen's identifier.

Every identifier or pseudonym will be represented as a coordinate point (x, y) on an Elliptic Curve Cryptography (ECC) over a finite field of elements. ECC is a cryptographic method based on elliptic curves. The elliptic curve is defined by the equation $y^2 = x^3 + ax + b \bmod p$, where a and b are the defined constants, and p is a prime number that defines the finite field.

We will use the elliptic curve **P-521** with following standards parameters :

Element	Value
p	0x01ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
a	0x01fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffc
b	0x0051953eb9618e1c9a1f929a21a0b68540eea2da725b99b315f3b8b489918ef109e156193951ec7e937b1652c0bd3bb1bf073573df883d2c34f1ef451fd46b503f00
n	0x01fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffa51868783bf2f966b7fcc0148f709a5d03bb5c9b8899c47aebb6fb71e91386409

More information see section 2.4, "Standard curve P-521 parameter"

In order to convert an input to a point on an elliptic curve prior to sending a request to the Pseudonymisation service, multiple steps are required:

1. **Convert an identifier to an Elliptic Curve Point (ECPoint):**
   a. Convert the identifier into a byte array representation *idBytes*

      <u>**Important:**</u> The maximum allowable input size value is restricted to 32 bytes**.** If you need to process input data exceeding this 32 bytes limit, please refer to the annex 9 for guidance.

   b. Create a new byte array *xBytes* that has a length of : *1 + idBytes.length + 1 + xBufferSize.*
      The first byte is set to zero to avoid negative integer and the remaining bytes are filled with zeros.

      The value for xBufferSize to be used can be obtained from the *bufferSize* property found in the response for the api resource *GET /domains/{domainKey}* (see section 7.1.2).

   c. Copy the idBytes into the xBytes array, starting from the second byte.

   d. The byte following the copied idBytes in the xBytes array should contain the length of idBytes.

   e. Compute the x-coordinate by converting the xBytes to a integer.

      <u>**Important**</u>**:** The byte array should be interpreted in big endian format when converting it to an integer.

   f. Compute the y-coordinate using the elliptic curve equation:
      $y^2 = x^3 + ax + b \bmod p$ or the optimize equation $y^2 = (x^2 + a)\, x + b \bmod p$

      If there is no valid square root for y, we increment the x-coordinate by one and start over until we have a valid square root for y is found.

   Finally, the method returns an ECPoint on the curve and the x and y coordinates of the computed point.

For example, let's suppose that we wish to convert the identifier *27589314370* into a ECPoint, using a buffer size of 8 bytes :

a. Convert the identifier into a byte array representation *idBytes*.

In the byte array representation, the ASCII values of each digit in the identifier are located at positions 1 to 11.

Position	1	2	3	4	5	6	7	8	9	10	11
ASCII	2	7	5	8	9	3	1	4	3	7	0
BYTE	50	55	53	56	57	51	49	52	51	55	48

b. Create a new byte array *xBytes* that has a length of : *1 + idBytes.length + 1 + xBufferSize.*
The first byte is set to zero to avoid negative integer and the remaining bytes are filled with zeros.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

c. Copy the idBytes into the xBytes array, starting from the second byte.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	50	55	53	56	57	51	49	52	51	55	48	0	0	0	0	0	0	0	0	0

d. The byte following the copied idBytes in the xBytes array should contain the length of idBytes.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	50	55	53	56	57	51	49	52	51	55	48	11	0	0	0	0	0	0	0	0

e. Compute the x-coordinate by converting the xBytes to a integer.

X=28668071561010989222337884734618748926120742092 8

f. Compute the y-coordinate using the elliptic curve equation.

Y=188178764951682755389851931589844778797575008423303079003697335670303135562247441576140589895265645722294840245775339215094670449846003744468637289234539838 5

g. Convert the x-coordinate and y-coordinate value in big endian two-complement byte array.

X=

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
50	55	53	56	57	51	49	52	51	55	48	11	0	0	0	0	0	0	0	0

Y=

1	2	3	4	5	6	7	8	9		63	64	65	66
0	-116	89	-94	110	35	-124	102	102	.....	106	68	116	113

h. Convert the x-coordinate and y-coordinate byte array to a base64 string.

X= Mjc1ODkzMTQzNzALAAAAAAAAAAA=

Y=
AIxZom4jhGZmdZxOmVydi5Whp5btbktt5k3T95AkVigxP82+i6NMXbENqPnvyOegn9B9RZ9dZgIVRw+Qxa5qRHRx

You can use the provided input from the table below to calculate the ECPoint with both x and y coordinates and verify that these coordinates match the values x and y for the input:

input as base64	xBufferSize	X	Y
MQ==	8	MQEAAAAAAAAAAA==	AIaELoSge8ZH9YhlIeahODkJDe23j1NUsQqwm32j1o+CdG6lIKhi1KqdONLRsXh+ciZjddZtm2dShPpp3K5aDxD8
MTI=	8	MTICAAAAAAAAAAA=	AURV0WACmddkLIQK2IF76J0XGOygU+mSZ5gnbQFQ7WSyEUq9H/c0738e2pwVTiSxcI1xz1dGqTcYfotir3K0LQrz
MTIz	8	MTIzAwAAAAAAAAAC	QRJBFXWQueauTw1qeNzTM+3qZtLLbykEN8LCmrn29Nv3IXHlLFa9Hhd/iuRBHDsgv4usav38XCWgxroQ8zWruQ4=
MTIzNA==	8	MTIzNAQAAAAAAAAAAg==	CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfMTB/rM+/H3jeAgFkR+lvIQ=
MTI0NQ==	8	MTI0NQQAAAAAAAAAAA==	AULi2K88MtySUCftPD3k2HAtq2myTVkgT33sMxm4Np2RJSdBJjPiamRyS3vIbKEaf0aNRan6B1lp4xJAr/ZAsdit
MTIzNDU2	8	MTIzNDU2BgAAAAAAAAA	AdnB30YG45HqxJyenfTwyomOV5NSCs9AbKKQFRBpEzDwuJ4FMak5UpxYElx7yGAGYjBcbEqV8vdjEBoGSewjYJx4
MTIzNDU2Nw==	8	MTIzNDU2NwcAAAAAAAAAA==	D5OUrNIZPGXYb+soXNh8VMiPYWudaCNAEZX/4B1EftfRMQOxStZBr23pCdCpaxpyZNc0HicDeBo+/TflO9kF10c=
MTIzNDU2Nzg=	8	MTIzNDU2NzgIAAAAAAAAAE=	AfSlL6dUUnkvIowaMspc6avl4TvCqC4WE/NmEb1q3edqhmjBi8d3ku4GahorYpTkKDDGf1mV36ynC/o2/Zh8PqC7
MTIzNDU2Nzg5	8	MTIzNDU2Nzg5CQAAAAAAAAB	AWxzHDVoA6b08d7mpWsCn/AatpfglyUpsa4soLUzaS+HfRiwFT3EfvEd/1/CLVnVS0U/CUqT9tGNlhry0eWiyDMJ
MTIzNDU2Nzg5MA==	8	MTIzNDU2Nzg5MAoAAAAAAAAAA==	ALoZzeaX7m5Yz1YEUKfbyNYSOblCaUyK+CagReh+9BArAcI+d77cdV3iZQ52hI06Xfbd67J7jXbTKuVoRsDbCIdY
MTIzNDU2Nzg5MA==	10	MTIzNDU2Nzg5MAoAAAAAAAAAAC	ANSl+G1Ihrn/y4KZmB2iO8ck7FXMvW6ExHVptUrKGfzInHfPukbx+xYNX9OhO/hRFQic3i7+Q6O6rlwhH4da4rrd
MTIzNDU2Nzg5MA==	12	MTIzNDU2Nzg5MAoAAAAAAAAAAAAE=	ASvP5vN+onuAc4jzCxECvkdBbUTfu3XpV6tUNGh3x68aUmMkz/6kFzrd+DEw/9g+oP1cLsJCuyYlzjq2PRVxwmrl
RzDziSOxzz1fT6lMEPYT8C5xenPFTFwOhZe4CACeLbc=	8	RzDziSOxzz1fT6lMEPYT8C5xenPFTFwOhZe4CACeLbcgAAAAAAAAAA=	ALARpzdxggw1mTjxYZKwdGOP0oyYKYjmqye1MewE9SP1zCp5wtSOpedAZNeyN1THUV0+WoXLUDCB1NZWT25xz5N6
cmFuZG9tdGV4dA==	8	cmFuZG9tdGV4dAoAAAAAAAABA==	AJXsoDpTMzMepP5g7Q0/aY11CrEp1cfgrw5+0S2XO5nNxzOu2jtpkgV9GOutVporRsKnLBs0VSglcRs+qINRK/4D
MTIzNDU2Nzg5MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTIzNDU2Nzg5MA==	input too large		

2. Blind an ECPoint:

To protect identifiers and pseudonyms from being exposed to the Pseudonymisation service, it is necessary to blind the ECPoint obtained in step 1 by following these steps:

a. For each request, generate a new true random integer to be used as a scalar, ensuring that its value is in the following range:]1 ,p]

b. Compute a new ECPoint on the curve using the following formula:
ECPoint × (scalar mod n)

c. Normalize the ECPoint to ensure that resulting point's coordinate is in the canonical form

This new ECPoint represents the x, y coordinates that should be included in your request when invoking the pseudonymisation service.

You can use the provided input from the table below to calculate the blinded ECPoint with both x and y coordinates and verify that these coordinates match the values x and y for the input:

X	Y	Scalar	Scalar mod inverse	Expected X	Expected Y
MQEAAAAAAAAAAA==	AIaELoSge8ZH9YhlIeahODkJDe23j1NUsQqwm32j1o+CdG6lIKhi1KqdONLRsXh+ciZjddZtm2dShPpp3K5aDxD8	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AV+VXF9H5LdTe4b1SSC7bHjp6b2enJmfplC6a3/jCR5fUHxXRSaRniYR8h7ugNqalGvP49cZnv6lf9B72RUG0rA/	eSmII52CEtsZzSseUDY3YKLtSgqhq1wLPm9ncHBzGiv1wMIxmc1jSmpW36GhTt/s1P5shZGhG8ncoWKSGkJDyfw=
MTICAAAAAAAAAAA=	AURV0WACmddkLIQK2IF76J0XGOygU+mSZ5gnbQFQ7WSyEUq9H/c0738e2pwVTiSxcI1xz1dGqTcYfotir3K0LQrz	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	ANuVBi6VwQfa2F8Y+OM4NOQ3EtEOaLhWA92tYheMdK5DN9iubeC2pyy7gWEQKHkr5wx9VLv5ahX+3cqXdi+KvSn1	ASbUF7UOw7WtsUKPk7yVCL9gPZSoP9DD2Mm0cxSUJd1qrO5A+mbQfQhKnQdkXp00T0U5kHNIhVKRNQtN7tIqW1FK
MTIzAwAAAAAAAAAC	QRJBFXWQueauTw1qeNzTM+3qZtLLbykEN8LCmrn29Nv3IXHlLFa9Hhd/iuRBHDsgv4usav38XCWgxroQ8zWruQ4=	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AJDTs17IHvcNMOdK7qZjfbnze1QFoTm8HBbvTpfD3khGbM01goo+fhkJ77VsCAZGSxny/JnlG6Vy0QE8hA/fX3JL	APgdC1G8Whnf/F84mrlrw5FsJxy7GTkjjuVZmXFQTGvb9A2ppoyckftqX6Da8H2jib1wJUbspxPjfXXwrB6yMRgh
MTIzNAQAAAAAAAAAAg==	CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfMTB/rM+/H3jeAgFkR+lvIQ=	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AXRwRG7lqfJT8IHCDFE0Zi6SNM0mciAf/q5SpS2gdAzZdAu+L+lFnahEtHKSLt86l+T45kl1CVP8OM7LdcFJYuJn	AYUMbBVYgqgshWpf487AKX3ESaW2ftxzP90tW5hNOxPgETMzcsODv7oEU2Zdip3WZUa+ZhSL1avLJh68I3DjYGac

MTI0NQQAAAAAAAAAA==	AULi2K88MtySUCftPD3k2HAtq2myTVkgT33sMxm4Np2RJSdBJjPiamRyS3vIbKEaf0aNRan6B1lp4xJAr/ZAsdit	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AKJ4Hl8nxryGEvH++kUsg908A1W5od0ZFFsx6M1ayvhHsVHEWaPah5zblwn5KhLVwcRmpYAH1CjLMexa2YEV3FCB	AINtXU2NW833s/ZZng2LFykdRmKnEoJ/U+QUvtltalUaH/YNPK6fdIll7maOXwTY21Ch4BFwgdy+qbKGy/oQLGUc
MTIzNDU2BgAAAAAAAAAA	AdnB30YG45HqxJyenfTwyomOV5NSCs9AbKKQFRBpEzDwuJ4FMak5UpxYElx7yGAGYjBcbEqV8vdjEBoGSewjYJx4	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AVNW4mBCsSdzEs+JAi1vME8T4o5bB2chKvOiwr8ZxSu/53Epo8avyesca7XABzLqAWeFGHLQZxR9NZuE9O7Ztthf	AUWBi3+hVu2oceV6Et67l1GieDF0WOehUD1ulk78TI0qdcE6IeJWx/XWaY052oOqHy+0WjBJXJ7Unvgf4oXiPVfd
MTIzNDU2NwcAAAAAAAAAAA==	D5OUrNIZPGXYb+soXNh8VMiPYWudaCNAEZX/4B1EftfRMQOxStZBr23pCdCpaxpyZNc0HicDeBo+/Tfl O9kF10c=	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AJjABYkAK6Oo5WkvSfVfubJSvH4px0T9Y5//P5gFv///BvAqF3AiGKwdUDLrbWRkEVjZOYB6icxImRWA5j1cJH0j	AQiKLKsMCcpLZKXHMvgSjYWXKLIZA0+lyvbO+BzZiaxiiJUBlUdZG4PhN/3MwEHP85nMMxBVMg9DDSqQM2+KLqU1
MTIzNDU2NzgIAAAAAAAAAAE=	AfSlL6dUUnkvIowaMspc6avl4TvCqC4WE/NmEb1q3edqhmjBi8d3ku4GahorYpTkKDDGf1mV36ynC/o2/Zh8PqC7	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	AM95qyrZGpjcEpa/mXDyPdU/XkFQukQctmXRDUjS2Y1//x5BORsCxoRhEkOKYnt8u11g3vmm+dnc+t5Q37RevJzt	AYZdFHNds2xdHG1ysp0YahcatO2mzyIjnN75NHo718Gxv2goRdvluyuJrCcH33Nnn/ei5NXrQNh9P8Q18uNReVgx
MTIzNDU2Nzg5CQAAAAAAAAAB	AWxzHDVoA6b08d7mpWsCn/AatpfglyUpsa4soLUzaS+HfRiwFT3EfvEd/1/CLVnVS0U/CUqT9tGNlhry0eWiyDMJ	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	Ad8a+82BH4JSpxQ/hT3jMEXLZ1AQ5zKmzyRviSrjfXZsCCpy2LXRbaBjpeAg1Yjl9xO5yazIfoqWxB/swoiKWYvc	AJr1+M1tQeV+UH5Em1jNWCfmmOX+JVQNMt3myKJrY5o8b87gCnF/XqRkS/so6pCAo+s1ltrcDC6oGjpVxMp/GKR0
MTIzNDU2Nzg5MAoAAAAAAAAAAA==	ALoZzeaX7m5Yz1YEUKfbyNYSOblCaUyK+CagReh+9BArAcI+d77cdV3iZQ52hI06Xfbd67J7jXbTKuVoRsDbCIdY	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	ATXxK6NZFjvkwVVdDaZhxdloOo5FhvuZFse61cipoEk84ZLOk1w0qBqUsVWOm/upGAwMD8l1FmvuZ1gH+2u7ECd/	AKGmtoy//h6dsi2nvNfywPA4g0G+vP94xkg+XU2Cd5vROjCbu109N/EtAf+3x+jcaUH00s6gO+ENinJ/I+hnJXs9
MTIzNDU2Nzg5MAoAAAAAAAAAAAAC	ANSl+G1Ihrn/y4KZmB2iO8ck7FXMvW6ExHVptUrKGfzInHfPukbx+xYNX9Oh	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGg	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	dQ1JgDTuuYJtHI0pQOiFJGDGBLoQYqyrENshTQKzHmDv5SSfsmbvN2PA553VleAM3xhxcB6mGmBOADrXsjhoRU8=	AN++bpBMbAaib6MNKAUlfm/cZbKYWmG7xNyp5CqiIpR6MHATDMo6YGKvohBWncq/DUzet1DUKq1mpo1CCiGW6Mj6

	O/hRFQic3i7+Q6O6rlwhH4da4rrd	v5l0Gk68qczZjt8yfO8FRcF			
MTIzNDU2Nzg5MAoAAAAAAAAAAAAAAE=	ASvP5vN+onuAc4jzCxECvkdBbUTfu3XpV6tUNGh3x68aUmMkz/6kFzrd+DEw/9g+oP1cLsJCuyYlzjq2PRVxwmrl	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	fUzCiOY+2TINafVJV8Fw/b+FKmvupngaZa6P/GG25KRybVHIO8p2xHhW1xBsb1FvKskgAhIXUAoekeCRpPDTXuc=	AdHBseAXBpudGLk+wmPOKAGv2af5y1KYqGsZfiNgGCZ6k+L91coa5HeBDOP9MIdT5dgdolvcbZXbUbYZUd3g/GmR
RzDziSOxzz1fT6lMEPYT8C5xenPFTFwOhZe4CACeLbcgAAAAAAAAAAA=	ALARpzdxggw1mTjxYZKwdGOP0oyYKYjmqye1MewE9SP1zCp5wtSOpedAZNeyN1THUV0+WoXLUDCB1NZWT25xz5N6	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	ARKh2805XnUxXwd7qAZ946L5HypbSD0J+A8W4Z+cmenFgeZFwbPkJpZ2tpso9EwOJkboGtVvzASFL/D/Lf94/vA=	APNowE+iovhITW/UipIZQSFJGuTmYlacQDL+1bJgpbFtPrB3hv0jWrApaDOcYzgDFjTy1v892o18DnbO/z21SN1a
cmFuZG9tdGV4dAoAAAAAAAAABA==	AJXsoDpTMzMepP5g7Q0/aY11CrEp1cfgrw5+0S2XO5nNxzOu2jtpkgV9GOutVporRsKnLBs0VSglcRs+qINRK/4D	AZ2TzlDmMWQW/kgJng4nThRUW149uzE+SNBUp4T0/0kRO2/zg2os/8dOEzZrvQPGgv5l0Gk68qczZjt8yfO8FRcF	ANjr/s4E0Mnbt2fy7CroYLQcVbH3YoktqrvzfYgVwB8sg4HsxgBLGNRqeDrj8OLR7v068HROzzwWc6P3gdgDmD7U	P/au8R8yR8aYFNOfY5EPu4DVlqXC+6Oh/Z5d6T/S4huWQgxsadxbcFRniWvnsjAsvitvRODmM8NK+nh1QwW1vIU=	AeHg+C0RKMlsWzZem0sYd8+2+/i0XYHDCNiXTFz5fo6eHzOYFwUe3LLcPizLq/OhHWlJ5/zRlk5stjXpAiD/qpqT

3. Unblind an ECPoint:

When you obtain a blinded ECPoint and you need to unblind it, proceed with these steps:

a. Compute modular multiplicative inverse of the given scalar (from step 2) with respect to the order of the elliptic curve by using the following formula:
$(scalar)^{-1}$ mod n

b. Compute the ECPoint using the following formula:
ECPoint × $(scalar \bmod n)^{-1}$

You can use the provided input from the table below to calculate the unblinded ECPoint with both x and y coordinates and verify that these coordinates match the values x and y for the input:

4. To Convert an ECPoint into identifier. Follow these detailed steps:

Note: this operation is relevant only with identify operation

a. Compute the x-coordinate into a byte array

b. Obtain the identifier by removing the length of bufferSize from the byte array

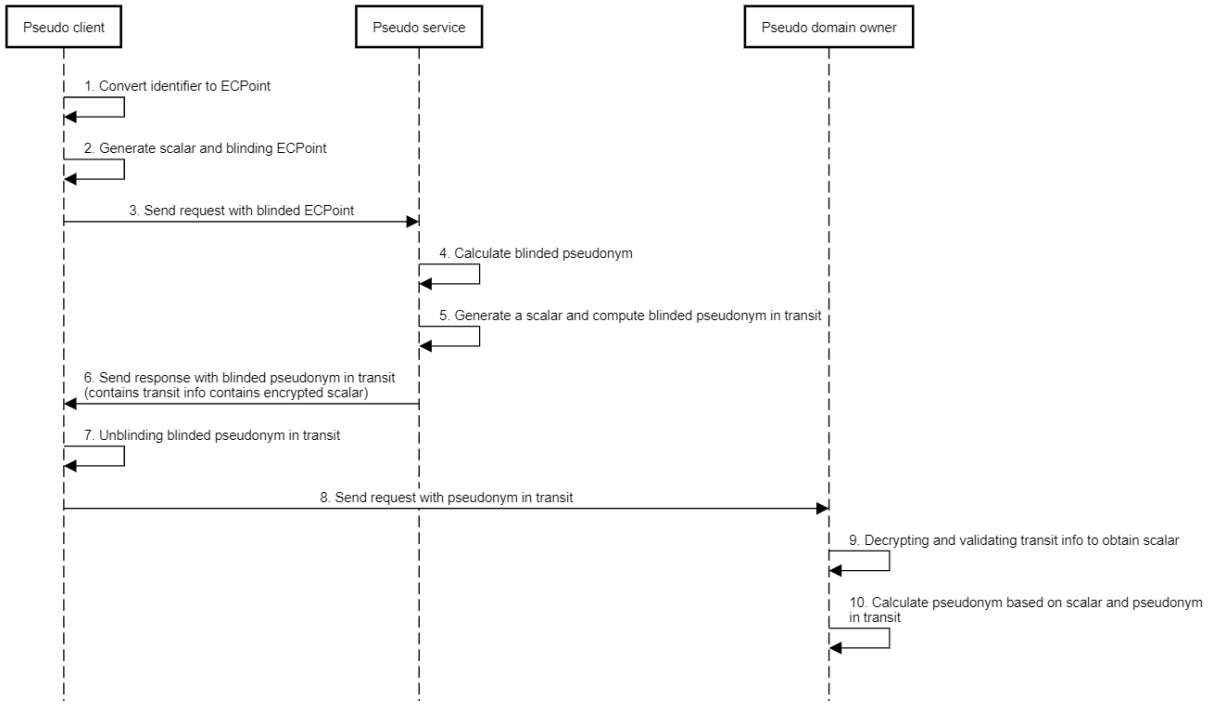
The value for BufferSize to be used can be obtained from the *bufferSize* property found in the response for the api resource *GET /domains/{domainKey}* (see section 7.1.2).

6. Process description

In this section, we will describe the step-by-step process of carrying out pseudonymisation, identification, conversion and one-input argument operations involving various actors. Within the forthcoming diagrams, we will distinguish three actors: *Pseudo client (e.g. a healthcare professional through a web application)*, the *Pseudo service*, the *Pseudo domain owner (e.g. a back-end system) and the Pseudo Consumer (Attribute Authority, RNConsult services).*

6.1 Pseudonymisation

This process describes the steps required within the same domain for a *Pseudo client* to compute a pseudonym from an identifier and send it to the *Pseudo domain owner*.

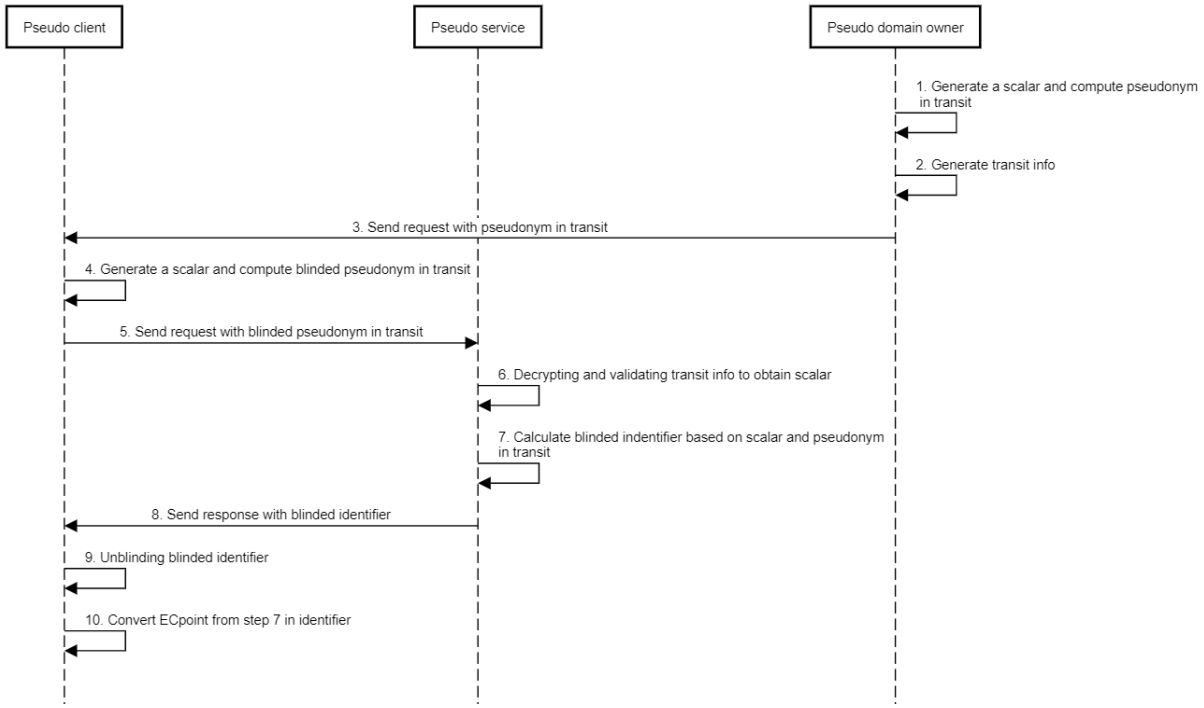


Step	Description
1	The pseudo client converts an identifier into an elliptic curve point (ECPoint). For more details on how to convert an identifier into an ECPoint, please refer to point 1 in section 5.
2	The pseudo client generates a scalar value (true random between 1 and order of the curve p) and derives a new blinded ECPoint based on the scalar and the ECPoint from step 1. For more details on blinding, please refer to point 2 in section 5.
3	The pseudo client sends the blinded ECPoint to pseudo service. For more details on the request, please refer to section 7.2.1.1.
4	The pseudo service receives this blinded ECPoint and calculates the blinded pseudonym.
5	The pseudo service generates its own scalar value (true random between 1 and order of the curve p) and derives a new blinded ECPoint for the blinded pseudonym in transit.

	The same logic is applied as in step 2, which involves calculating the new ECPoint using the formula (ECPoint multiplied by the scalar).
6	The pseudo service sends the blinded pseudonym in transit including the encrypted transitInfo to the pseudo client. For more details on the response, please refer to section 7.2.1.2. For more details on the transitInfo, please refer to section 8.
7	The pseudo client proceeds to unblinding the blinded pseudonym in transit. For more details on unblinding, please refer to point 3 in section 5.
8	The pseudo client sends a request to the pseudo domain owner, including the pseudonym in transit obtained from the previous step and the transitInfo from step 6.
9	The pseudo domain owner receives the request and decrypts the transit information to obtain the scalar value. The decryption of transitInfo can be achieved by retrieving the secret key through a get domain operation. It is essential that the retrieved secret key matches the corresponding secret key version indicated by the "kid" value from the header of the transitInfo JWE. For more details on the get domain operation, please refer to section 7.1.2.
10	The pseudo domain owner calculates a pseudonym based on the scalar obtained from the transitInfo and the pseudonym in transit received from the client.

6.2 Identification

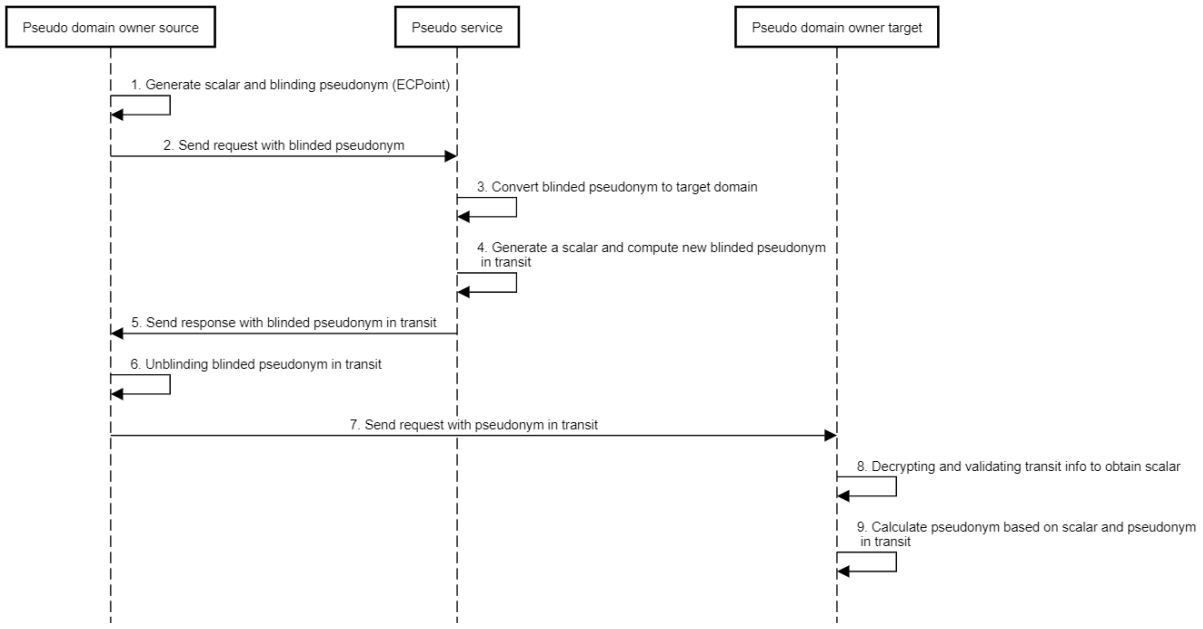
This process describes the steps required within the same domain for a *Pseudo domain owner* who sends the pseudonym to the *Pseudo client* in order to get the identifier.



Step	Description
1	The pseudo domain owner generates his own scalar value (true random between 1 and order of the curve p) and derives a new ECPoint representing the pseudonym in transit. The calculation of the new ECPoint can be achieved by multiplying the existing ECPoint with the scalar.
2	The pseudo domain owner generates an encrypted transitInfo. For more details about transitInfo, please refer to section 8.
3	The pseudo domain owner sends the request with pseudonym in transit including the encrypted transitInfo from previous step. For more details about the request, please refer to section 7.3.1.1.
4	The pseudo client generates his own scalar value and derives a new blinded ECPoint representing the blinded pseudonym in transit. For more details about blinding, please refer to point 2 in section 5.
5	The pseudo client sends a request to the pseudo service, including the blinded pseudonym in transit and the encrypted transitInfo from the previous step.
6	The pseudo service receives the request and decrypts the transitInfo to obtain the scalar value.
7	The pseudo service calculates a blinded identifier based on the scalar obtained from the transitInfo and the pseudonym in transit received from the pseudo client.
8	The pseudo service sends the response with the blind identifier.
9	The pseudo client unblinds the identifier. For more details about unblinding, please refer to point 3 in section 5.
10	The pseudo client converts the ECPoint into a identifier. For more details about converting an ECPoint into a identifier, please refer to point 4 in section 5.

6.3 Conversion

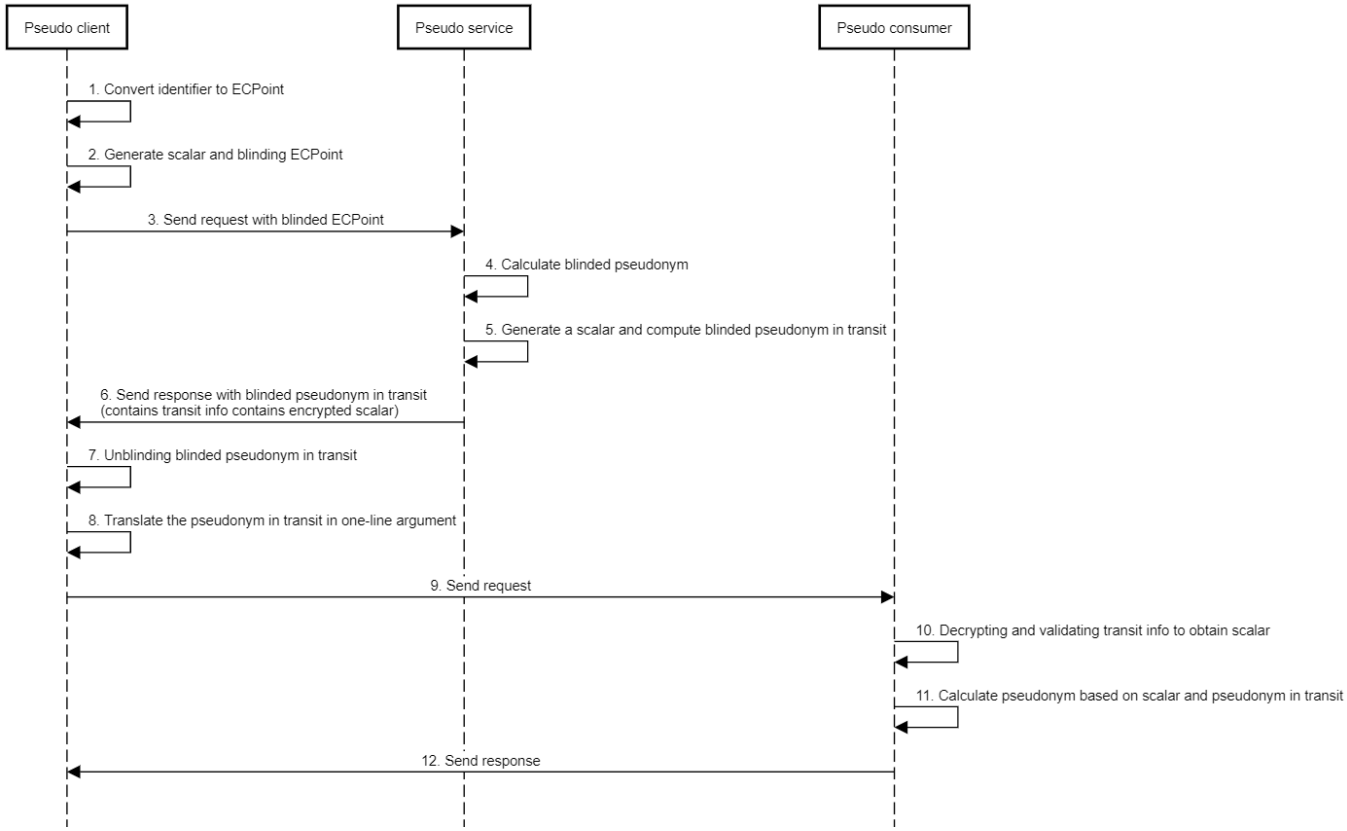
This process describes the steps required for a *Pseudo domain Owner* within the source domain to convert a pseudonym and send it to the *Pseudo Domain Owner* in the target domain.



Step	Description
1	The pseudo domain owner source generates his own scalar value (true random between 1 and order of the curve p) and derives a new blinded ECPoint for the pseudonym in transit. The calculation of the new ECPoint can be achieved by multiplying the existing ECPoint with the scalar using the formula.
2	The pseudo domain owner source sends the request with the blinded ECPoint to pseudo service. For more details about request, please refer to section 7.4.1.1.
3	The pseudo service converts the blinded pseudonym to target domain.
4	The pseudo service generates its own scalar value and derives a new blinded pseudonym in transit (ECPoint).
5	The pseudo service sends the response to pseudo domain owner source, including the blinded pseudonym in transit and the encrypted transitInfo. For more details about the response, please refer to section 7.2.1.2. For more details on the transitInfo, please refer to section 8.
6	The pseudo domain owner source unblinds the blinded pseudonym in transit. For more details about unblinding, please refer to point 3 in section 5.
7	The pseudo service sends the request with pseudonym in transit and encrypted transitInfo to pseudo domain owner target.
8	The pseudo domain owner target receives the request and decrypts the transit information to obtain the scalar value. The decryption of transitInfo can be achieved by retrieving the secret key through a get domain operation. It is essential that the retrieved secret key matches the corresponding secret key version indicated by the "kid" value from the header of the transitInfo JWE. For more details on the get domain operation, please refer to section 7.1.2. For more details on the transitInfo, please refer to section 8.
9	The pseudo domain owner target calculates a pseudonym based on the scalar obtained from the transitInfo and the pseudonym in transit received from the pseudo domain owner source.

6.4 One input argument

This process describes the necessary steps for a *Pseudo client* within the source domain to compute a pseudonym from an identifier and send it as one argument to the *Pseudo consumer*.



Step	Description
1	The pseudo client converts an identifier into an elliptic curve point (ECPoint). For more details on how to convert an identifier into an ECPoint, please refer to point 1 in section 5.
2	The pseudo client generates a scalar value (true random between 1 and order of the curve p) and derives a new blinded ECPoint based on the scalar and the ECPoint from step 1. For more details on blinding, please refer to point 2 in section 5.
3	The pseudo client sends the blinded ECPoint to pseudo service. For more details on the request, please refer to section 7.2.1.1.
4	The pseudo service receives this blinded ECPoint and calculates the blinded pseudonym.
5	The pseudo service generates its own scalar value (true random between 1 and order of the curve p) and derives a new blinded ECPoint for the blinded pseudonym in transit. The same logic is applied as in step 2, which involves calculating the new ECPoint using the formula (ECPoint multiplied by the scalar).
6	The pseudo service sends the blinded pseudonym in transit including the encrypted transitInfo to the pseudo client. For more details on the response, please refer to section 7.2.1.2.

	For more details on the transitInfo, please refer to section 8.
7	The pseudo client proceeds to unblinding the blinded pseudonym in transit. For more details on unblinding, please refer to point 3 in section 5.
8	The pseudo client translates the pseudonym In transit in one-line argument in base64. For more detail about the conversion process, please refer to annex 11.
9	The pseudo client sends a request to the pseudo consumer with the pseudonym obtained in previous step.
10	The pseudo consumer receives the request and decrypts the transit information to obtain the scalar value.
11	The pseudo consumer calculates a pseudonym based on the scalar obtained from the transitInfo and the pseudonym in transit received from the client.
12	The pseudo consumer send the response with the pseudonym in transit to the pseudo client.

7. API description

7.1 Domain information

7.1.1 GET /domains

This resource enables to retrieve the list of domains configured.

7.1.1.1 Request

Example:

```
GET https://api-acpt.ehealth.fgov.be/pseudo/v1/domains
```

7.1.1.2 Response

The response will return if a list of domains configured with the following properties exists:

Element	Description
Domain	A string representing the name of the domain.
Desc	A string representing a description of the domain.
Crv	A string representing the name of the elliptic curve used for cryptography.

Example:

```
[
      {
      "domain": "domainA_v1",
      "desc": "Pseudo domain",
      "crv": "P-521"
   },
      {
      "domain": "domainB_v1",
      "desc": "Pseudo domain",
      "crv": "P-521"
   }
]
```

7.1.2 GET /domains/{domainKey}

This resource enables retrieving information about a defined domain.

7.1.2.1 Request

Path parameter	Description
domainKey	A string representing the name of the domain.

Example:

```
GET https://api-acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1
```

7.1.2.2 Response

The response will return if the domain configured with the following properties exists:

Element	Description
audience	A string representing the target audience.
bufferSize	An integer representing the buffer size for the data.
jku	An array of URLs representing the location of JSON Web Key Sets (JWKS) that contains the public for encrypting data
timeToLiveInTransit	A string representing the duration for which a pseudonym in transit remains valid. The representation format for this duration follows the [ISO 8601] extended format: PnYnMnDTnHnMnS where: nY represents the number of years. nM represents the number of months. nD represents the number of days. 'T' serves as the separator between date and time components. nH represents the number of hours. nM represents the number of minutes. nS represents the number of seconds. **Remark**: The number of seconds can include decimal digits to arbitrary precision.
domain	A string representing the name of the domain.
desc	A string representing a description of the domain.
crv	A string representing the name of the elliptic curve used for cryptography.
secretKeys	A string representing the secret key(s).
secretKeys.kid	A string representing the identifier of the secret key. **Remark**: The identifier is unique and will be associated with only one jku.
secretKeys.active	A BOOLEAN value indicating whether the key is currently active.

secretKeys.encoded	An object representing the JSON Web Encryption secret key.
	See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.2***
	
	The encrypted secret key's content should be represented as a JSON Web Key (JWK) for the symmetric key, rather than as a byte array of the symmetric key.
	
	The JWK content will include the following claims:
	`{`
	`"kid": "generated kid"`
	`"kty": "oct",`
	`"alg": "A256GCM|A192GCM|A128GCM linked to the security strength",`
	`"k": "<value of the symmetric key>"`
	`}`

accessRules	An object representing the active access rules configured by the security team for the *domainkey*. For more details about the object and his content, please refer to annex 10.

Example:


```
{
    "audience": "https://api-
acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1",
    "bufferSize": 8,
    "jku": ["https://api.ehealth.fgov.be/etee/v1/pubKeys/cacerts/jwks?identifi
er=XXXXXXXXX&type=CBE&use=enc"],
    "timeToLiveInTransit": "PT10M",
    "secretKeys":    [
            {
         "kid": "a3d64de8-c516-4a5a-8e5a-c9227b5d52c2 ",
         "active": true,
         "encoded":          {
            "protected": "eyJlbmMiOiJBMTkyR0NNIn0",
            "iv": "jnpnTYhuk2X0omtS",
            "ciphertext": "wPYvxS1FtPMZ92pyKKY58AHI8tkSmV85b-
FrQ8bcSdDq73EhLmgwnH6XYGY",
            "tag": "qX1YzJbGpO1JEd1Ni8ndjw",
            "recipients": [           {
               "header":               {
                  "kid": "E139831296625319143514103319122908200342",
                  "alg": "RSA-OAEP-256",
                  "jku": "
https://api.ehealth.fgov.be/etee/v1/pubKeys/cacerts/jwks?identifier=XXXXXXXXXX
&type=CBE&use=enc"

               },
               "encrypted_key": "p5TfiABjZXfIgWCLMVYy9vDi1kifMi0LwZzpRfgHD4JS
5kMTu2lnmbYDVrRWQlyPmU5SLot2od0dYkptY3hnxh_RdmP4txeZOET-
f2N2X7xMbuacd6WQ8jO3qcwNMcMAYmbY95RmtqeJncSYAUo611TEXVYPthdq2pa7tvI6P02qKLrLEC
c3PPhdnpJeUqgJ07-9feFGcSCSqvorI5_motHLoNkCIaleJJ1lauP6-
jLk2mvhiklitnprf_s_gof9TkwP3O6JuGs2G0uVBm26KQoDAZTz2OoWxrwExMcAc9zrml2gogDBh19
mFXTgYFIsPcz1-i-a2FeTufrCGA-ZwA"
            }]
         }
      }
    ],
 "accessRule":    {
      "domain": "domain_v1",
      "type": "custom",
      "details":       [
```

```
                {
            "operation": "identify",
            "userGroups": [            {
               "claims": [               {
                  "path": "$.resource_access.api.roles[*]",
                  "value": "identify"
               }],
               "name": "identify",
               "description": "identify role"
            }]
         },
                {
            "operation": "pseudonymize",
            "userGroups": [            {
               "claims": [               {
                "path": "$.resource_access.api.roles[*]",
                  "value": "pseudonymize"
               }],
               "name": "pseudonymize",
               "description": "Pseudonymize role"
            }]
         },
                {
            "operation": "convert/ehealth_v1",
            "userGroups": [            {
               "claims": [               {
                   "path": "$.resource_access.api.roles[*]",
                  "value": "convert"
               }],
               "name": "convert",
               "description": "Convert role"
            }]
         },
      ],
      "signature": "..."
   },
    "domain": "domainA_v1",
    "desc": "Refferal prescriptions",
    "crv": "P-521"
}
```


7.2 Pseudonymize

7.2.1 POST /domains/{domainKey}/pseudonymize

This resource enables the pseudonymisation of the citizen identifier (i.e., INSS), in order to anonymize the identity.

Important: please refer to Annex Error! Reference source not found. **(section** Error! Reference source not found.**) for recommendation regarding the caching of pseudonyms in transit.**

7.2.1.1 Request

Path parameter	Description
domainKey	A string representing the name of the domain.

Body	Description
Id	A string representing the request identifier in UUID format.
Crv	A string representing the name of the elliptic curve used for cryptography.
X	A string representing in base64 the X-coordinate of a point on an elliptic curve.
Y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** **Important** : For each response that includes a transitInfo property, you should ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-
acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1/pseudonymize
{
    "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
    "crv": "P-521",
    "x": "MTIzNAQAAAAAAAAAAg==",
    "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfMTB
/rM+/H3jeAgFkR+lvIQ="
  }
```

7.2.1.2 Response

The response will return a pseudonym, that is, the pseudonymised citizen's identity along with the following properties:

Element	Description
Id	A string representing a unique identifier for the request.
domain	A string representing the domain name.
Crv	A string representing the name of the elliptic curve used for cryptography.
iat	The timestamp of when the response was created (in seconds since the Unix epoch).
Exp	The timestamp of when the response will expire (in seconds since the Unix epoch).
X	A string representing in base64 the X-coordinate of a point on an elliptic curve.
Y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.

transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1.*** For more details on the transitInfo property, please refer to section 8.
inResponseTo	A string representing a unique identifier for the response.

Example:


```
{
    "id": "70a1eedc-d2a4-4b5d-b264-8c4be3e4c06d",
    "domain": "domainA_v1",
    "crv": "P-521",
    "iat": 1677231615,
    "exp": 1677232215,
    "x": "IYzzUb0l5mmewxHJ+6i3qijKuNFzo6fAHiqg5sn3C3t2ApoIAxX6hOCpzUjjZYJBZECy
ShWVFqxIzTFqbUV9Lm4=",
    "y": "ATSq992VI1ojO0fEbUNiO08d69ZRaNMTAt6tLPDD8JC9ti14DLr0g4fs6S2SGWquX2KS
p6+1exdhQYiMdlX5DreA",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..U49qk7KS4rIz0T4s.FNVDAkUteWZNG_kP2kC_IcgQqnq3hsXw3yqQeAQkuVqla
dk7wGaSvY1GfeMsHJ6MOsdqeRhoffUJ0IOUbsyTuhIEL-_oiXIHn3uu7Bj_TK0AvfIeowqUHj87t_-
V5bnCtEQzTu2ZOFejMF7G5x_xJGnF6Ne7lKKrZ4rf64JlVCO2zHd2UFYck2TnW9sUP7IQ8jULBnR_2
AZd2rFsjRtCi4arbxUxUetfZTvG9ybCf4H1pN2hYrtWL5yRvTL24JU7KYFqGd6S8XcfMDgJ.VHO1kp
roGsB-YeP-VRwzqQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
 }
```


7.2.2 POST /domains/{domainKey}/pseudonymizeMultiple

This resource enables the pseudonymisation of multiple citizen identifiers (i.e., INSS), in order to anonymize the identity of multiple citizens.

7.2.2.1 Request

Path parameter	Description
domainKey	A string representing the name of the domain.

Body	Description
inputs	An array representing the properties required to pseudonymise citizen identifiers. The number of citizen identifier to pseudonymise is limited to a minimum of 2 and a maximum of 10.
id	A string representing the request identifier in UUID format.
crv	A string representing the name of the elliptic curve used for cryptography.
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.

y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** Important : For each response that includes a transitInfo property, you must ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-
acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1/pseudonymizeMultipe
{"inputs":[
    {
      "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
      "crv": "P-521",
      "x": "MTIzNAQAAAAAAAAAAg==",
      "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfM
TB/rM+/H3jeAgFkR+lvIQ="
    }
    ]}
```

7.2.2.2 Response

The response will return multiple pseudonyms, that is, the pseudonymised citizens identity along with the following properties:

Element	Description
outputs	An array representing the properties resulting from the pseudonymisation of the citizen identifier. Remark: the order of the elements in this array follows the same order as in the request *inputs* array.
id	A string representing a unique identifier for the response.
domain	A string representing the domain name.
crv	A string representing the name of the elliptic curve used for cryptography.
iat	The timestamp of when the response was created (in seconds since the Unix epoch).
exp	The timestamp of when the response will expire (in seconds since the Unix epoch).
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** For more details on the transitInfo property, please refer to section 8.
inResponseTo	A string representing a unique identifier for the response.

Example:


```
"outputs": [   {
    "id": "a82c91d1-b8c5-48da-9450-18b2b26258e5",
    "domain": "domainA_v1",
    "crv": "P-521",
    "iat": 1677231615,
    "exp": 1677232215,
    "x": "Aa8QD8qZnADuWxzUFlKwXgtuwhoUKS8m+NtbXqKhBLuenB+sZj1aJ6t9qN0+YzZrQTIE
vnFbMmM2h8+diHYXyNl4",
    "y": "OVzMxZg1LVy9Sl+JgK+1Nw5/L0UkuIx3SyIbOhFtbxbuxo+ko5OoWf4fPmVi68O+/8NR
9b9YP9OnfK4F536OilY=",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..G3o3ie1J3Wux5HzW.UWiS_6bjhtaSj4ewTwY4mGzoHPIbN4dwSdkIceYRy1hL3
cRFEvr4Sf-
NIiYPiMCP87XRNvDXtyRYPrA2l35rTgHLB9KfWr1dR8nrZu8yeZdDIZ50qH5Z8SoE_zvZN-
9rraEkZ587ufmCaUxFnQ4dZWXJ4jhfpObA8TFutTXIMgvjMEfbm1-z5xNXRx8t-CqDdtMwmiP3-
4n3vZDro1lb2--Bg1iqZDBNxApc5sSw1E7GLdcLxHlNGDH1kyIvXu-ZEu28o3ZkZHmEw-
Zo.X9Vau4zsDaeVuWiH3K2GBQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
 }]
}
```


7.3 Identify

7.3.1 POST /domains/{domainKey}/identify

This resource is the opposite of resource *pseudonymize*. It enables to identify the pseudonym of a citizen.

7.3.1.1 Request

Path parameter	Description
domainKey	A string representing the name of the domain.
Body	Description.
id	A string representing the request identifier in UUID format.
crv	A string representing the name of the elliptic curve used for cryptography.
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** **Important** : For each response that includes a transitInfo property, you must ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1/identify
{
    "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
    "crv": "P-521",
    "x": "MTIzNAQAAAAAAAAAg==",
    "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfMTB
/rM+/H3jeAgFkR+lvIQ="
  }
```

7.3.1.2 Response

The response will return the citizen's identifier along with the following properties:

Element	Description
id	A string representing a unique identifier for the request.
domain	A string representing the domain name, in this case.
crv	A string representing the name of the elliptic curve used for cryptography.
iat	The timestamp of when the response was created (in seconds since the Unix epoch).
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** For more details on the transitInfo property, please refer to section 8.
inResponseTo	A string representing a unique identifier for the response.

Example:

```
{
    "id": "70a1eedc-d2a4-4b5d-b264-8c4be3e4c06d",
    "domain": "domainA_v1",
    "crv": "P-521",
    "iat": 1677231615,
    "exp": 1677232215,
    "x": "IYzzUb0l5mmewxHJ+6i3qijKuNFzo6fAHiqg5sn3C3t2ApoIAxX6hOCpzUjjZYJBZECy
ShWVFqxIzTFqbUV9Lm4=",
    "y": "ATSq992VI1ojO0fEbUNiO08d69ZRaNMTAt6tLPDD8JC9ti14DLr0g4fs6S2SGWquX2KS
p6+1exdhQYiMdlX5DreA",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..U49qk7KS4rIz0T4s.FNVDAkUteWZNG_kP2kC_IcgQqnq3hsXw3yqQeAQkuVqla
dk7wGaSvY1GfeMsHJ6MOsdqeRhoffUJ0IOUbsyTuhIEL-_oiXIHn3uu7Bj_TK0AvfIeowqUHj87t_-
V5bnCtEQzTu2ZOFejMF7G5x_xJGnF6Ne7lKKrZ4rf64JlVCO2zHd2UFYck2TnW9sUP7IQ8jULBnR_2
AZd2rFsjRtCi4arbxUxUetfZTvG9ybCf4H1pN2hYrtWL5yRvTL24JU7KYFqGd6S8XcfMDgJ.VHO1kp
roGsB-YeP-VRwzqQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
```

```
}
```

7.3.2 POST /domains/{domainKey}/identifyMultiple

This resource is the opposite of resource *pseudonymizeMultiple*. It enables to get the identification of multiple pseudonym.

7.3.2.1 Request

Path parameter	Description
domainKey	A string representing the name of the domain.

Body	Description
Inputs	An array representing the properties required to identify pseudonymised citizen identifiers. The number of citizen identifier to identify is limited to a minimum of 2 and a maximum of 10.
Id	A string representing the request identifier in UUID format.
crv	A string representing the name of the elliptic curve used for cryptography.
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** **Important** : For each response that includes a transitInfo property, you must ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-
acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1/identifyMultiple
{"inputs":[
    {
      "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
      "crv": "P-521",
      "x": "MTIzNAQAAAAAAAAAAg==",
      "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfM
TB/rM+/H3jeAgFkR+lvIQ="
    }
    ]}
```

7.3.2.2 Response

The response will return multiple identifications, that is, the non-pseudonymised citizens identities along with the following properties:

Element	Description
outputs	An array representing the properties resulting from the identification of the citizen identifiers. Remark: the orders of the element in this array follows the same order as in the request *inputs* array.
id	A string representing a unique identifier for the request.
domain	A string representing the domain name.
crv	A string representing the name of the elliptic curve used for cryptography, in this case, "P-521".
iat	The timestamp of when the pseudonym was created (in seconds since the Unix epoch).
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1aud:*** This claims specifies the audience. For more details about the transitInfo property, please refer to section 8.
inResponseTo	A string representing a unique identifier for the response.

Example:


```
"outputs": [   {
    "id": "a82c91d1-b8c5-48da-9450-18b2b26258e5",
    "domain": "domainA_v1",
    "crv": "P-521",
    "iat": 1677231615,
    "exp": 1677232215,
    "x": "Aa8QD8qZnADuWxzUFlKwXgtuwhoUKS8m+NtbXqKhBLuenB+sZj1aJ6t9qN0+YzZrQTIE
vnFbMmM2h8+diHYXyNl4",
    "y": "OVzMxZg1LVy9Sl+JgK+1Nw5/L0UkuIx3SyIbOhFtbxbuxo+ko5OoWf4fPmVi68O+/8NR
9b9YP9OnfK4F536OilY=",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..G3o3ie1J3Wux5HzW.UWiS_6bjhtaSj4ewTwY4mGzoHPIbN4dwSdkIceYRy1hL3
cRFEvr4Sf-
NIiYPiMCP87XRNvDXtyRYPrA2l35rTgHLB9KfWr1dR8nrZu8yeZdDIZ50qH5Z8SoE_zvZN-
9rraEkZ587ufmCaUxFnQ4dZWXJ4jhfpObA8TFutTXIMgvjMEfbm1-z5xNXRx8t-CqDdtMwmiP3-
4n3vZDro1lb2--Bg1iqZDBNxApc5sSw1E7GLdcLxHlNGDH1kyIvXu-ZEu28o3ZkZHmEw-
Zo.X9Vau4zsDaeVuWiH3K2GBQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
 }]
}
```

7.4 Convert

7.4.1 POST /domains/{fromDomainKey}/convertTo/{toDomainKey}

This resource enables to convert a pseudonym from a source domain to another pseudonym for a target domain.

7.4.1.1 Request

Path parameter	Description
FromDomainKey	A string representing the name of the source domain.
toDomainKey	A string representing the name of the target domain.

Body	Description
id	A string representing the request identifier in UUID format.
crv	A string representing the name of the elliptic curve used for cryptography.
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** **Important** : For each response that includes a transitInfo property, you must ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-acpt.ehealth.fgov.be/pseudo/v1/domains
/domainA_v1/convertTo/domainB_v1


{
    "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
    "crv": "P-521",
    "x": "MTIzNAQAAAAAAAAAAg==",
    "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfMTB
/rM+/H3jeAgFkR+lvIQ="
  }
```

7.4.1.2 Response

The response will return the pseudonym along with the following properties:

Element	Description
id	A string representing a unique identifier for the request.
domain	A string representing the domain name.

crv	A string representing the name of the elliptic curve used for cryptography, in this case, "P-521".
Iat	The timestamp of when the response was created (in seconds since the Unix epoch).
exp	The timestamp of when the response will expire (in seconds since the Unix epoch).
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** For more details about the transitInfo property, please refer to section 8.
InResponseTo	A string representing a unique identifier for the response.

Example:


```
{
    "id": "01ce55a0-c575-430c-9ef3-11a9b55cb2ed",
    "domain": "domainB_v1",
    "crv": "P-521",
    "iat": 1677231615,
    "exp": 1677232215,
    "x": "OZADJVppdeQzwgvAUjQNaLvuf94ulY6iDgeip7iSHAW7TNrDBa0XMGeS6G3s/HWLSQ4e
irpcox28GghzbtaiUzg=",
    "y": "UPOBi75XsreuYfQwyVvIaHgpzrrdS6joS8JaPlkMPxeU8FmFHRtteJp/FAq91pEllcbH
4V4PRSC+QEm0C9thkO4=",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..h4IU5TFgQVqIR7VM.iSQbF1_yavnS_wocHZRViX3Xtfo25dq8bQto_C-
ZHzfJugVyRpQ3mlCDJRqHaBlAq10nd5z0L_dM2ut1NvRKglJvF0vk2cv6HY0RJgs1Wzbt-
EWxH7R2ttsKNo1TPdI8FEQfoNGJafQLktWHaIMUl5w2FeMFSYBVZtL4cn1sSCH_ndinbReAzJi-
ZX1aa3TD9abQVNy5xaF5-Df9Xys5G_SdjHxwUfH-BKudQmLk-
fcG5rFkSX4EhrbkptPOcSIO8im44d4rAsG8D5Bd.kBfA7FUaLyB0SMqhIyzOzQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
 }
```


7.4.2 POST /domains/{fromDomainKey}/convertMultipleTo/{toDomainKey}

This resource enables to convert multiple pseudonyms from a source domain to other pseudonyms for a target domain.

7.4.2.1 Request

Path parameter	Description
FromDomainKey	A string representing the name of the source domain.
toDomainKey	A string representing the name of the target domain.

Body	Description
Inputs	An array representing the properties required to identify pseudonymised citizen identifiers. The number of citizen identifier to identify is limited to a minimum of 2 and a maximum of 10.
Id	A string representing the name of the domain.
crv	A string representing the name of the elliptic curve used for cryptography.
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.
transitInfo (optional)	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** **Important** : For each response that includes a transitInfo property, ensure that this property is present in the next request. For more details about the transitInfo property, please refer to section 8.

Example:

```
POST https://api-acpt.ehealth.fgov.be/pseudo/v1/domains
/domainA_v1/convertMultipleTo/domainB_v1
{"inputs":[
    {
      "id": "989580dd-c17e-4c57-bab4-0cec27b7a859",
      "crv": "P-521",
      "x": "MTIzNAQAAAAAAAAAg==",
      "y": "CEI1kmrzzVyfeDxtQS3Gyme8//vbrYDneMIEmTFtaX2GyLhlA2JGNu0Wjd+qbpKUfM
TB/rM+/H3jeAgFkR+lvIQ="
    }
    ]}
```

7.4.2.2 Response

The response will return multiple pseudonym along with the following properties:

Element	Description
outputs	An array representing the properties resulting from the identification of the citizen identifiers. Remark: the orders of the element in this array follows the same order as in the request *inputs* array.
id	A string representing a unique identifier for the request.
domain	A string representing the domain name.
crv	A string representing the name of the elliptic curve used for cryptography.
Iat	The timestamp of when the pseudonym was created (in seconds since the Unix epoch).
exp	The timestamp of when the pseudonym will expire (in seconds since the Unix epoch).
x	A string representing in base64 the X-coordinate of a point on an elliptic curve.
y	A string representing in base64 the Y-coordinate of a point on an elliptic curve.

transitInfo	A string representing a JSON Web Encryption compact serialization containing transit information. See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1*** For more details about the transitInfo property, please refer to section 8.
InResponseTo	A string representing a unique identifier for the response.

Example:


```
"outputs": [   {
    "id": "5c557c59-18bf-4a80-94c5-c2c3ad12bfde",
    "domain": "domainB_v1",
    "crv": "P-521",
    "iat": 1677231616,
    "exp": 1677232216,
    "x": "L/TU8WTRYcbRRN3OI/5Uc6fM+IrO217icvo/XkjhiaZnzLuwH8b2hnbbiNx4f4BbHBjw
syAZsgN8EvptRoOmNOE=",
    "y": "AMZK0NZmhV7MEUtubSnG6ZxTdVr+AT0yAOjP4NA/HZ4W8wfNBZ0lg6w1SSx4BAMvDfZF
+YKUmdWikX4bbdTdgueU",
    "transitInfo": "eyJhbGciOiJkaXIiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiMjAyMi0xMi
IsImF1ZCI6Imh0dHBzOi8vYXBpLWludC5laGVhbHRoLmZnb3YuYmUvcHNldWRvL3YxL2RvbWFpbnMv
dWhtZXBfdjEifQ..YPTJ2AOw7b4Z0U5V.ICzTI-Ru1AgM-
U67Z1mjyB2dyQkKbtAVMw2vi76WOPhDT2dLvuXsaT9y5bAvD_UEYg0pCYVFtUaCPJwpAUwIPyvfKQY
OrXS_byObV7LA5yKtpvOushtahgQDeFGeX0x9Pm3wiGYo0Ytof-5fZUCw5DlWfkp5kwOUm4-e-
WrCtDuOq2X2vjV5mfGLhlzXR4GF-lnpZ4I1rcQRRZqT-4pX1moEYj20WdbOkNUh2iyv3Rk5T-
w16wVkL5OjtC7Ov5dBX0nonFm_T2Vfwqin.K44kzP1RMUf6KiIzH1d6CQ",
    "inResponseTo": "989580dd-c17e-4c57-bab4-0cec27b7a859"
 }]
}
```

8. Annex: TransitInfo

8.1 Description

The transitInfo is a string representing a JSON Web Encryption compact serialization containing transit information (See ***https://www.rfc-editor.org/rfc/rfc7516#section-7.1***). It will contains the properties mentioned below.

8.1.1 Header

Element	Description
alg	A string representing the algorithm used for the cryptographic operation.
enc	A string representing the encryption algorithm used for protecting the data.
iat	The timestamp of when the response was created (in seconds since the Unix epoch).
exp	The timestamp of when the response will expire (in seconds since the Unix epoch).
kid	A string representing the key identifier.
aud	A string representing the target audience.

8.1.2 Payload

The decryption of payload can be achieved by retrieving the secret key through a get domain operation (please refer to section 7.1.2) . It is essential that the secret key being retrieved matches the corresponding secret key version indicated by the "kid" value from the header of the transit info JWE.

Once the payload has been decrypted, you will have access to the following properties:

Element	Description
iat	The timestamp of when the response was created (in seconds since the Unix epoch). Important: In the next version, this field will be deprecated and will be available only in the header.
exp	The timestamp of when the response will expire (in seconds since the Unix epoch). Important: In the next version, this field will be deprecated and will be available only in the header.
scalar	A byte representing the scalar value.

8.2 Validation

When receiving a pseudonym in transit, it is important to verify the validity of the TransitInfo by performing the validations as described below.

8.2.1 Header

- The value of the "aud" claim must be equivalent to the endpoint of the resource in order to retrieve information about a given domain (For more details, please refer to section 7.1.2).
- To decrypt the payload and retrieve the scalar, you must obtain the decryption key by calling the getDomain/{domainKey} resource (For more details, please refer to section 7.1.2).

Please ensure that the secret key version in the response matches to the value of the "kid" claim for the obtained decryption key.

8.2.2 Payload

To verify the timestamp of the payload, you need to ensure that:

- The value of the claim "iat" is a timestamp in the past, indicating that the transitInfo was created before the current time.
- The value of the claim "exp" is a timestamp in the future, indicating that the transitInfo will expire after the current time.

Remark: It is also important to consider possible clock skew for the timestamps.

8.3 Generation

To generate a transitInfo, you need to generate a JSON Web Encryption compact serialization containing the following properties:

8.3.1 Encrypted payload

The following properties are required :

Element	Value
iat	Generate a timestamp for the current time. Important: In the next version, this field will be deprecated and will be need to be generated only in the header.
exp	Generate a timestamp in the future. Important: In the next version, this field will be deprecated and will be need to be generated only in the header.
scalar	Generate a new true random byte array to be used as a scalar, ensuring that its value is in the following range:]1 ,p] where p is the order of the curve.

Once the claims and the value for the payload are defined, they must be encrypted with the active encryption key of the domain. To retrieve this key, you need to call the getDomain/{domainKey} resource and obtain the active encryption key (For more details, please refer to section 7.1.2).

8.3.2 Header

The following properties are required :

Element	Value
alg	The value for the direct encryption algorithm must be “dir”.
enc	The value for the encryption algorithm must be "A256GCM".
kid	The value must be the secret key version associated with the encrypted key used for payload encryption. For more details, please refer to section 7.1.2.
aud	The value must be the resource endpoint for get information about a given domain Example for acceptance/domainA_v1 : ***https://api-acpt.ehealth.fgov.be/pseudo/v1/domains/domainA_v1***
iat	Generate a timestamp for the current time.
exp	Generate a timestamp for the current time.

9. Annex: Processing of input data exceeding 32 bytes

When you need to process input data where size exceeding the 32 bytes limit, please follow these steps:

1. Generate a Secure Random sequence of 32 Bytes to ensures that the generated values are unpredictable and suitable for use in encryption.

2. The 32-byte secure random value generated in step 1 is used as the encryption key for the AES-GCM algorithm with a specific key length of 256 bits (A256GCM).

   A JSON Web Encryption (JWE) header is constructed, containing:
   "kid": A Universally Unique Identifier (UUID) that serves as a unique identifier for this encryption key.
   "alg": Specifies the algorithm used, which is "dir" (direct encryption), indicating that the content is directly encrypted with the given key.
   "enc": Specifies the encryption algorithm, which is "A256GCM."

3. Serialize the Output to a Compact JWE String:
   The JWE header constructed in step 2, along with the payload (data to be encrypted), is serialized into a compact JWE string.

4. The AES-GCM key generated in step 1 (the 32-byte secure random value) is used as the input value (X) for the pseudonymisation service.

5. The output of the pseudonymisation service is translated into a one-line argument
   Important: the “kid” must be used as reference.

10. Annex: Object accessRules

An object containing access rule information configured by eHealth security team. These accessRules determine permissions and actions for specific user groups under certain conditions.

This object contains the following properties:

Element	Value
domain	A string representing the domain to which the access rule applies.
type	A string specifying the type of access rule. It will always have the value “custom”.
details	An array of objects, each object defining a pseudonymisation operation and associated user groups for the access rule.
details.operation	A string representing the pseudonymisation operation. The values are: “identify”, “pseudonymise” or “convert/target_domain”. **Remark**: For the “convert” operation, it should include the target domain. For example, “convert/ehealth_v1” in domain is about a conversion from domain X to domain eHealth_v1.
details.usergroups	An array representing the list of user groups configured for a given operation. Each user group is identified by a technical name and a description.
details.usergroups.claims	An array representing the list of claims defined for a user group. For each claim, there is an associated value that must be verified in the access token obtained via eHealth IAM Connect. Note: If a claim and its associated value are prefixed with ">>", the claim is not verified in the access token. Instead, it is verified elsewhere, such as in the user info token or transit information.
signature	A string representing the digital signature associated with the access rule content for ensuring data integrity.

11. Annex: Pseudonym in Transit: Conversion to one-line Argument

To convert an pseudonym in transit to one-line argument, consider the following approaches:

1. **Encoding a pseudonym in transit as a Base64 JSON String**
   Suppose you have a pseudonym in transit with elliptic curve point (x, y) and transitInfo. You need to:
   - convert it into a single-line string.
   - Encode this string in base64 format.

2. **Encoding a pseudonym in transit as a String with the pattern ${SEC1}:${transitInfo}**
   Suppose you have a pseudonym in transit with elliptic curve point (x, y) and transitInfo. You need to :
   - Convert your elliptic curve point (x, y) as a String using SEC1 encoding.
   - Append the transitInfo to the String calculated to comply with the pattern ${SEC1}:${transitInfo}.

   For details on converting (x, y) elliptic curve point to SEC1 string, please refer to the eHealth platform document references (section 2.4 – ID 4)

Although both approaches can be considered, the second approach in less overhead than the first.

The eHealth base services that support the pseudonym as input, such as the Attribute Authority, are compatible with the two approaches.

12. Risks and security

12.1 Security

12.1.1 Business security

In case the development adds a use case based on an existing integration, the eHealth platform must be informed at least one month in advance. A detailed estimate of the expected load is necessary to be able to ensure an effective capacity management.

When technical issues occur on the service, the partner can obtain support from the contact centre (see Chap 3).

If the eHealth platform should find a bug or vulnerability in its software, the partner must update his application with the latest version of the software, within ten (10) business days.

If the partner finds a bug or vulnerability in the software or web service made available by the eHealth platform, he is obliged to contact and inform us immediately. He is not allowed, under any circumstances, to publish this bug or vulnerability.

12.1.2 The use of username, password and token

The username, password, and token are strictly personal.
Every user takes care of his username, password and token, and he is forced to confidentiality of it. It is prohibited to transfer them to partners and clients. Until inactivation, every user is responsible for every use, including the use by a third party.

13. Implementation aspects

13.1 Procedure

This chapter explains the procedures for testing and releasing an application in acceptation or production.

13.1.1 Initiation

If you intend to use the eHealth platform service, please contact ***pseudo@ehealth.fgov.be*** who will provide you with the necessary information and mandatory documents.

13.1.2 Development and test procedure

You have to develop a client in order to connect to our WS. Most of the required integration info to integrate is published on the portal of the eHealth platform.

In some cases, the eHealth platform provides you, upon request, with test cases in order for you to test your client before releasing in the acceptance environment.

13.1.3 Release procedure

When development tests are successful, you can request to access the acceptance environment of the eHealth platform. From this moment, you start the integration and acceptance tests. The eHealth platform suggests testing during minimum one month.

After successful acceptance tests, the partner sends his test results and performance results with a sample of "eHealth request" and "eHealth answer" by email to his point of contact at the eHealth platform.

Then the eHealth platform and the partner agree on a release date. The eHealth platform prepares the connection to the production environment and provides the partner with the necessary information. During the release day, the partner provides the eHealth platform with feedback on the test and performance tests.

For further information and instructions, please contact: ***integration-support@ehealth.fgov.be***.

13.1.4 Operational follow-up

Once in production, the partner using the eHealth platform service for one of his applications will always test first in the acceptance environment before releasing any adaptations of its application in production. In addition, he will inform the eHealth platform on the progress and test period.

13.2 Error and failure messages Http codes

These are the error status codes that can be returned by the Pseudonymisation Rest service:

Code	Message	Description
200	OK	The resource has been fetched and is transmitted in the message body or the resource describing the result of the action is transmitted in the message body.
400	Bad request	The structure of the query parameters or the JSON body is not correct.
401	Not authenticated	The user did not sent a valid JWT with the request.
403	Forbidden access	The accessKey does not match the profile of the user in the JWT.
404	Not Found	The resource or the endpoint does not exist.

500	Internal server error	An unexpected error occurred, but the service is not down (to remove).
503	Service temporarily unavailable	The service is down.