END-TO-END ENCRYPTION
Known recipient (EtkDepot) - REST
Cookbook
Version 1.1

This document is provided to you free of charge by the

eHealth platform

Willebroekkaai 38 – 1000 Brussel

38, Quai de Willebroeck – 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	28/11/2019	eHealth platform	Initial version
1.1	09/04/2020	eHealth platform	I-WS Compliance

2. Introduction

2.1 Goal of the service

The End-To-End Encryption (ETEE) basic REST services only offer building blocks that allow integrating secure communications in applications.

It does not offer a pre-packaged 'End-To-End' business solution. This means you have to create your own client application with an implementation of a:

- ETK Client
- software that integrates a cryptographic solution
- way to pass on a message reference to a message receiver
- way to pass on a key reference to a message receiver (optional if a key reference is used in the Message Storage Server (MSS))
- a Message Storage Center (you could store the message reference in the MSS).

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 ETEE services in their own custom application.

This document will provide all the necessary elements to get you started developing. It explains in that context:

- the main concepts and principles
- the usage of ETK Depot WS

This cookbook only deals with the first phase of the ETEE project: encryption of messages for a 'known recipient' (also known as 'addressed messages').

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 the requirements of specifications, data format and release processes of the eHealth platform as described in this document.

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 versions can be used for the eHealth platform service.

ID	Title	Version	Date	Author
1	Glossary.pdf	1.0	Pm	eHealth platform
2	eHealth Services – Web Access	2.0	12/07/2019	eHealth platform

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

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	Basic Profile Version 1.1	http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html	24/08/2004	Web Services Interoperability Organization

3. Support

3.1 For issues in production

eHealth platform contact center:

- Phone: 02/788 51 55
- 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)

3.2 For issues in acceptance

Integration-support@ehealth.fgov.be

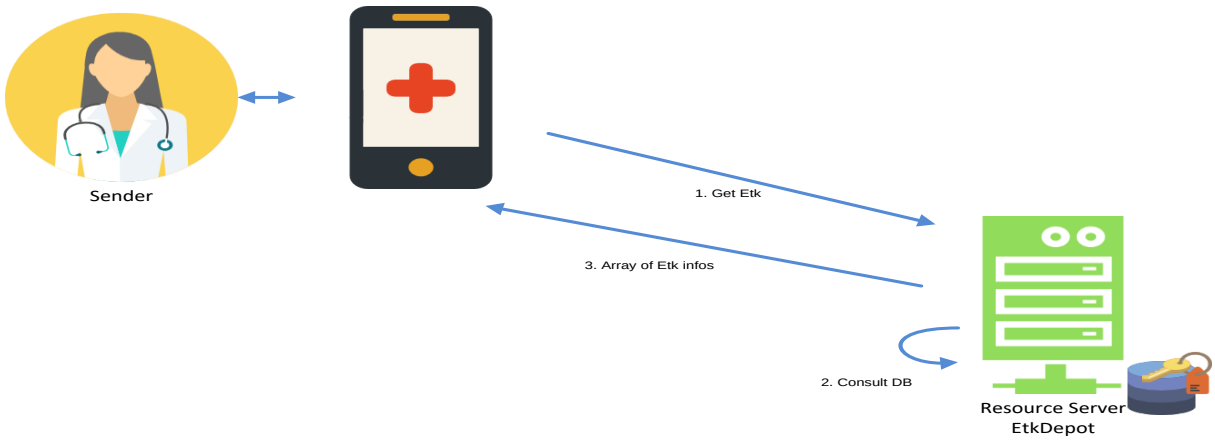
3.3 For business issues

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

4. Global overview

4.1 High-level schema of the ETEE Know Recipients functionality

The example below describes how to use the ETEE Etk Rest service in a business scenario. This example show the method GET /etks, the process is applicable for the others methods.



1. The client can request an Etk (with eventually application id).
2. The service will consult and search the (these) Etk(s) in ETEE database.
3. The eHealth platform will return the Etk(s) asked.

5. Step-by-step

5.1 The Etk Rest Services

The REST interface is described with a JSon / Swagger API.

5.1.1 GET /Etks

5.1.1.1 Request

Requests public key of known recipient.

The input is composed by:

- The identifier = The identifier of the Etk owner (ssin for exemple).
- The type of the identifier = Type of previous identifier (ssin/nihii/cbe/ehp).
- The application identifier = The application for which we the requester wants the public key for the identifier specified. This parameter is OPTIONAL

Example:

GET https://testURL/etee/v1/etks?identifier=0123456789&type=CBE&applicationIdentifier=appID

5.1.1.2 Response

This GET operation returns an array of etkInfo Json object, containing all the keys found in EtkDepot for parameters specified in input.

→ Each etkInfo object contains the identifier, the identifier type, the application id, and the etk itself.

Example:

[{
"key": {
"applicationIdentifier": " appID ",
"cbe": "0123456789"
},
"value":
"MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgEFADCABgkqhkiG9w0BBwGggCSABIID6DCCA/MwggLboAMCAQICEQCgvrRngX….."
}]

5.1.2 WS-I Basic Profile 1.1

Your request must be WS-I compliant (Cfr External Ref). If not you will receive one of the errors SOA-03001 – SOA-03003.

5.1.3 GET /jwks

5.1.3.1 Request

Requests authentication or encryption certificate of know recipient.

The input is composed by:

- The identifier = The identifier of the Etk owner (ssin for exemple).

- The type of the identifier = Type of previous identifier (ssin/nihii/cbe/ehp).
- The application identifier = The application for which we the requester wants the public key for the identifier specified. This parameter is OPTIONAL.
- The use of the jwks (“sig” or “enc”). See RFC7517 §4.2 for more information
- The time for the lookup (lookup moment default is now). This parameter is OPTIONAL.

Example:

GET https://
testURL/etee/v1/pubKeys/cacerts/jwks?identifier=0123456789&type=CBE&applicationIdentifier=appID&use=sig

5.1.3.2 Response

This GET operation returns an array of etkInfo Json object, containing all the keys found in EtkDepot for parameters specified in input.

→ Each etkInfo object contains the use, the kid, and the certificate itself.

Example:

{"keys": [{
"kty": "RSA",
"use": "sig",
"kid": "S5436060553504243135",
"x5t#S256": "rV5UMJFfcn_SYdTwxA73US6SyNpnB-ghnqt1UG5TlN8",
"e": "AQAB",
"x5c":
["MIIGnTCCBIWgAwIBAgIIS3DERZ0iPb8wDQYJKoZIhvcNAQELBQAwczELMAkGA1UEBhMCQkUxETAPBgNVBAoMCFpFVEVTIFNBMQwwCgYDVQQFEwMwMDExQzBBBgNVBAMMOlRFU1QtWmV0ZXNDb25maWRlbnMtZUhlYWx0aCBhY2NlcHRhbmNlIHRlc3QtaXNzdWluZyBDQSAwMDEwHhcNMTkwNTIwMDk1ODMzWhcNMjAwNTIwMDk1ODMzWjCBqjELMAkGA1UEBhMCQkUxGzAZBgNVBAoMEkZlZGVyYWwgR292ZXJubWVudDEQMA4GA1UECwwHTkVXUk9PVDEXMBUGA1UECwwOQ0JFPTA0MDY3OTgwMDYxDjAMBgNVBAsMBVNNQUxTMSEwHwYDVQQLDBhlSGVhbHRoLXBsYXRm…."
}]}

5.1.4 GET /jwks/{kid}

5.1.4.1 Request

Requests authentication or encryption certificate of the kid given in input.

5.1.4.2 Response

This GET operation returns only one etkInfo Json object, containing the key found in EtkDepot for kid specified in input.

→ The etkInfo object contains the use, the kid, and the certificate itself.

5.1.5 GET /keyholders/{kid}

5.1.5.1 Request

Requests key holder of an ETK with the kid given in input.

5.1.5.2 Response

This GET operation returns only one etkInfo Json object, containing the key holder found in Etkdepot for kid specified in input.

→ The etkInfo object contains the identifier, the identifier type, the name of the key holder.

Example:


```
{
  "aud": "public",
  "keyholder": {"enterprise":   {
    "cbe": "0123456789",
    "name": "JohnDoeCompany"
  }},
  "iss": "urn:be:fgov:ehealth:etee:etkdepot",
  "exp": 1568814875,
  "iat": 1568811275,
  "jti": "2085bd23-7da1-4bcc-98c7-60ffea7c6146"
}
```

6. Risks and security

6.1 Risks & safety

6.2 Security

6.2.1 Business security

In case the development adds an additional use case based on an existing integration, the eHealth platform must be informed at least one month in advance with a detailed estimate of the expected load. This will ensure an effective capacity management.

In case of technical issues on the WS, the partner may obtain support from the contact center (see Chap 3)

In case the eHealth platform finds a bug or vulnerability in its software, we advise the partner to update his application with the newest version of the software within 10 business days.

In case the partner finds a bug or vulnerability in the software or web service that the eHealth platform delivered, he is obliged to contact and inform us immediately. He is not allowed to publish this bug or vulnerability in any case.

6.2.2 The use of username, password and token

The username, password and token are strictly personal. Partners and clients are not allowed to transfer them. Every user takes care of his username, password and token and he is forced to confidentiality of it. Moreover, every user is responsible for every use, which includes the use by a third party, until the inactivation.

7. Implementation aspects

7.1 Procedure

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

7.1.1 Initiation

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

7.1.2 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.

Upon request, the eHealth platform provides you in some cases, with a mock-up service or test cases in order for you to test your client before releasing it in the acceptance environment.

7.1.3 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***.

7.1.4 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.

8. Error and failure messages

8.1 Http codes

Here are the error status codes that can be returned by the EtkDepot Rest service (non-exhaustive list):

- 400 The Etk REST message is incorrect.
- 500 The Etk request could not be completed due to an internal server error.

8.2 Error codes originating from the eHealth platform:

These error codes first indicate a problem in the arguments sent, or a technical error.

Error code	Component	Description	Solution
SOA-03001	*Consumer*	*This is the default error for content related errors in case no more details are known.*	***Malformed message***
SOA-03002	*Consumer*	*Message does not respect the SOAP standard.*	***Message must be SOAP***
SOA-03003	*Consumer*	*Message respects the SOAP standard, but body is missing.*	***Message must contain SOAP body***