D4.6 Homomorphic encryption embedded engine v1

WP4 - SPHINX Toolkits

Version: 1.00

SPHINX

A Universal Cyber Security Toolkit for Health-Care Industry

Disclaimer

Copyright message

Document information

Grant Agreement Number	826183	Acronym	SPHINX
Full Title	A Universal Cyber Security Toolkit for Health-Care Industry		
Topic	SU-TDS-02-2018 Toolkit for assessing and reducing cyber risks in hospitals and care centres to protect privacy/data/infrastructures		
Funding scheme	RIA - Research and Innovation action		
Start Date	1st January 2019	**Duration**	36 months
Project URL	http://sphinx-project.eu/		
EU Project Officer	Reza RAZAVI (CNECT/H/03)		
Project Coordinator	National Technical University of Athens - NTUA		
Deliverable	D4.6 Homomorphic encryption embedded engine v1		
Work Package	WP4 – SPHINX Toolkits		
Date of Delivery	**Contractual** M20	**Actual**	M20
Nature	R - Report	**Dissemination Level**	P - Public
Lead Beneficiary	TechInspire		
Responsible Author	Waqar Asif	**Email**	w.asif@techinspire.co.uk
		Phone	
Reviewer(s):	Alberto López Martín (INCM), Ilias Trochidis (VILABS)		
Keywords	Homomorphic encryption , privacy , anonymisation		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Document History

Version	Issue Date	Stage	Changes	Contributor
0.1	15/06/2020	Draft	ToC	Waqar Asif (TEC)
0.2	24/08/2020	Draft	Content	Waqar Asif (TEC)
0.3	26/08/2020	Draft	Internal Review 1	Alberto López Martín (INCM)
0.4	26/08/2020	Draft	Internal Review 2	Ilias Trochidis (VILABS)
0.5	27/08/2020	Pre-final	Comments Addressed	Waqar Asif (TEC)
0.6	31/08/2020	Pre - final	Quality Control	George Doukas (NTUA), Michael Kontoulis (NTUA)
1.00	31/08/2020	Final	Final Version	Christos Ntanos (NTUA)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Executive Summary

This deliverable is a report on the development progress of the Homomorphic Encryption Embedded Engine (HE). The HE tool provides both security and privacy to the SPHINX solution. The tool makes use of partial homomorphic encryption techniques to allow user to search in the encrypted domain. This eliminates the need for downloading and decrypting all stored data and thus ensures security of the data. With the help of this tool, healthcare professionals can store data in central repositories with increased security. The tool also provides the feature for searching in each other's database, given that prior permission is already granted. Alongside this, the tool provides network traffic anonymization capability, which adds privacy to the network traffic data. A pseudo-anonymized version of the network traffic is returned to the user, with added capability of de-anonymizing the data for latter use. This report elaborates on these functionalities and helps explain with the help of screenshots the different interfaces that it provides. This report is a versioned document, and this is the first version (v1).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Table of Figures

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Table of Tables

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Table of Abbreviations

HE - Homomorphic Encryption

RSA - Rivest, Shamir, Adleman

SE - Searchable Encryption

GDPR - General Data Protection Regulation

AES - Advanced Encryption Standard

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

1 Introduction

1.1 Purpose & Scope

This deliverable is being submitted as a partial requirement for D4.6 Homomorphic Encryption Embedded Engine. This is part of Work Package 4 - SPHINX Toolkits. This embedded engine deals with providing a data anonymization service to all incoming network traffic and also provides an interface to different SPHINX end users to search in each other's databases using a double-sided blinded process. This deliverable highlights the current state of the embedded engine at month 20. At the moment, the core functionalities are complete, with testing in progress. The document contains detailed descriptions of all modules in place and highlights the different functionalities with the help of screenshots.

1.2 Structure of the deliverable

The rest of the deliverable is structured as follows. Section 2 provides a brief overview of Homomorphic Encryption, details about various partial homomorphic encryption techniques and elaborates on the background of searchable encryption. Section 3 explains in detail the three different components of the Homomorphic Encryption Embedded Engine and it elaborates on the architecture and design that were adopted. Section 4 highlights the different interfaces that the tool presents, it presents the API definitions and elaborates with the help of screenshots on the used test cases. Section 5 concludes the document.

1.3 Relation to other WPs & Tasks

This report is closely related to WP2 and more specifically to D2.3 Use Case Definition and requirement document, D2.4 SPHINX requirements and guidelines and D2.5 SPHINX architecture. The work presented in this document has been tailored to fulfil the integration requirements of T6.1 Definition and specification of the system integration.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

2 Homomorphic Encryption

Homomorphisms are maps between algebraic structures that allow the development of cryptographic techniques that, in turn, permit computations to be performed on encrypted data. This as a result helps maintain data confidentiality while it is being processed, thus enabling tasks to be performed when data is residing in untrusted environments. In the current age of heterogeneous networking, this is a highly valuable capability [1].

A homomorphic encryption system emanates from conventional public key cryptographic systems, which means that it uses a public-private key pair to execute the cryptographic functions. The public key is thus used to encrypt data, whereas a private key is used to decrypt the ciphertext. What makes it different from most cryptographic solutions is that it allows one to perform arithmetic operations on the encrypted data. Mathematically, a homomorphism refers to the transformation of one dataset into another while preserving relationships between elements in both sets [1]. As homomorphic encryption maintains the same structure, identical mathematical operations produce similar results irrespective of what they are exercised on. An arithmetic operation performed on ciphertext would give similar results to an arithmetic operation performed on plain text.

2.1 Literature review

The search for a solution that allows computing on the encrypted data has been a long set goal since 1978, when it was initially proposed by Rivest, Shamir, Adleman (RSA). The key reason for interest in this topic is the large set of applications that it can help execute. The development of a fully homomorphic encryption model is a revolutionary advance and thus relies on huge computational resources. A fully homomorphic encryption model assumes the capability of performing all arithmetic operations in the encrypted domain. As resources can be a big limitation, researchers have resorted to slight variations of this model. There are three different types of homomorphic encryption models depending on the frequency of mathematical operations that can be performed on the cipher [1][2]:

- Fully homomorphic encryption,
- Partial homomorphic encryption,
- Somewhat homomorphic encryption.

While fully homomorphic encryption gives one the freedom to perform all arithmetic operations without any limitations in operation of frequency, its computational complexity limits the usability of such approaches. To circumvent this issue, researchers typically resort to partial homomorphic encryption approaches, which only allow selected arithmetic operations to be performed. These include either multiplication or addition. The development of most partial homomorphic encryption models supports multiple iterations of such arithmetic operations. In environments where resources are hugely constrained, the use of somewhat homomorphic encryption is preferred, allowing only for single arithmetic operations to be executed only a limited number of times.

In this work, we focus on the use of partial homomorphic encryption due to the nature of the task. Partial homomorphic encryption can be exercised based on a number of asymmetric encryption schemes. These include RSA, ElGamal and Paillier:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

- RSA

RSA is known to exhibit multiplicative homomorphism [3]. This means that any content that is encrypted using the well-known RSA algorithm, if multiplied in the encrypted domain, would produce the same outcome as if it was multiplied in plain text. For instance, consider an RSA key pair (d, e) and modulus n. Then the encryption procedure for message m is:

$$c \equiv m^e mod\ n$$

And decryption procedure is:

$$m \equiv c^d mod\ n$$

If we encrypt two plain text messages p_1 and p_2 using RSA, then the corresponding ciphertext would be p_1^e and p_2^e. Multiplying these cipher texts results in $(p_1 p_2)^e$. When decrypted, this would result in $((p_1 p_2)^e)^d \equiv p_1 p_2 mod\ n$.

- ElGamal

The ElGamal encryption scheme also exhibits multiplicative homomorphism [4]. This means that the multiplication of two or more cipher texts that was created using the ElGamal scheme would result in a decrypted plain text that is same as multiplication in plain text. For instance, consider an example with an ElGamal public key (α, β, ρ) and private key a. Then the encryption of a plaintext x with nonce k would be $\varepsilon(x, k) = (y_1, y_2)$. Here:

$$y_1 \equiv \alpha^k mod\ p$$

$$y_2 \equiv x\beta^k mod\ p$$

In case, there are two plaintext messages x_1 and x_2 with nonces k_1 and k_2, then the ciphertexts are:

$$\varepsilon(x_1, k_1) = (y_1, y_2) = (\alpha^{k_1} mod\ p, x\beta^{k_1} mod\ p)$$

$$\varepsilon(x_2, k_2) = (y_3, y_4) = (\alpha^{k_2} mod\ p, x\beta^{k_2} mod\ p)$$

Multiplying the two ciphers would result in:

$$(y_1 . y_2).(y_3 . y_4) = (y_1 . y_2 . y_3 . y_4) = (\alpha^{k_1} . \alpha^{k_2} . x_1\beta^{k_1} . x_2\beta^{k_2})$$

$$= (\alpha^{k_1 + k_2} . x_1 x_2 \beta^{k_1 + k_2})$$

Decrypting the cipher text would result in:

$$d\left(\alpha^{k_1 + k_2} . x_1 x_2 \beta^{k_1 + k_2}\right) = x_1 x_2$$

- Paillier

The Paillier encryption scheme is different from both the RSA and the ElGamal approach as it exhibits additive homomorphism [4]. Under the Paillier encryption scheme, multiplication performed in the encrypted domain produces a similar result as if the relevant plain text components were added together. For instance, consider two plain text messages x_1 and x_2, the resultant cipher text under the Paillier encryption scheme would then be:

$$\varepsilon(x_1, r_1) = g^{x_1} r_1^n\ mod\ n^2$$

$$\varepsilon(x_2, r_2) = g^{x_2} r_2^n\ mod\ n^2$$

Multiplication of these two cipher texts would result in:

$$\varepsilon(x_1, r_1).\varepsilon(x_2, r_2) = g^{x_1} r_1^n . g^{x_2} r_2^n\ mod\ n^2 = g^{x_1 + x_2} (r_1 r_2)^n\ mod\ n^2$$

Decryption of the cipher text would result in:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

$$d\left(\varepsilon(g^{x_1+x_2}(r_1r_2)^n)\right) = (x_1 + x_2) mod\ n$$

2.2 Searchable Encryption

Searchable Encryption (SE) is not a new topic. It has been around for quite some time now. The key purpose of having SE was to provide users the flexibility of searching in the encrypted domain. In a conventional approach, security-conscious users encrypt data before storing it on a third-party cloud resource [5]. This enables them to ensure the right level of security. When in need of a specific content, researchers download all the content, decrypt it and then search in the decrypted text. Once the file is found, the rest of the files are then encrypted and stored back onto the cloud. This is a tedious approach and that makes it infeasible for large datasets [6].

In order to facilitate security conscious users, SE is used to provide the feature of being able to search in the encrypted domain. This minimizes the overhead of downloading all encrypted files. Conventional SE approaches rely on generation of an index table [7]. The index table keeps a track of keywords and their existence in the dataset. This index table then facilitates in performing the search in the encrypted domain. These approaches work well in reducing the overhead of downloading and decrypting all encrypted content, but they bring along their own complexities. The index table limits the keywords that can be searched. In case a new keyword needs to be added, the index table generation needs to be repeated, which means re-encrypting all content that needs to be sent to the cloud. This procedure increases computational costs whenever a new keyword needs to be added to the list. In this project, we overcome these limitations by using a Partial Homomorphic Encryption-based searchable encryption scheme. This eliminates the need of an index table and increases the usability of the solution. We make use of the multiplicative homomorphism nature of the RSA algorithm to produce ciphertext that facilitates the possibility of search in the encrypted domain.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

3 Homomorphic Encryption Embedded Engine

The Homomorphic Encryption (HE) based searchable encryption developed for the SPHINX toolkit is based on a partial homomorphic encryption scheme. The RSA-based partial homomorphic encryption scheme allows one to create a searchable cipher thus eliminating the need for any index generation. The tool built using this scheme provides a multitude of features for the SPHINX solution. These include searching in the encrypted domain, allowing a double-sided blinded search capability and providing data anonymity.

3.1.1 Search in the Encrypted Domain

The search in the encrypted domain capability allows one to search in the database that they have encrypted themselves. The encryption process makes use of two schemes, which include an AES encryption and an HE-based searchable cipher creation. This duo helps reduce computational complexity and increases performance efficiency. The HE-based searchable cipher is a one-way cipher, which means that it cannot be decrypted. On the other hand, the AES encryption-based cipher is created to ensure that, when needed, decryption can be exercised. The use of this duo is seamless. The user selects the set of files that need to be stored in the healthcare database, these files are encrypted by the tool and forwarded for storage. When a search needs to be performed, the tool uses the private key of the user and creates a trapdoor from the search query. This trapdoor is sent to the healthcare database, where it is executed. The trapdoor looks for the searched keyword and responds with the name of the file that contains that content, thus allowing users to search in the encrypted domain among multiple encrypted files. Throughout this process, no plain text files are shared between the client module and the healthcare repository, thus making it security compliant.

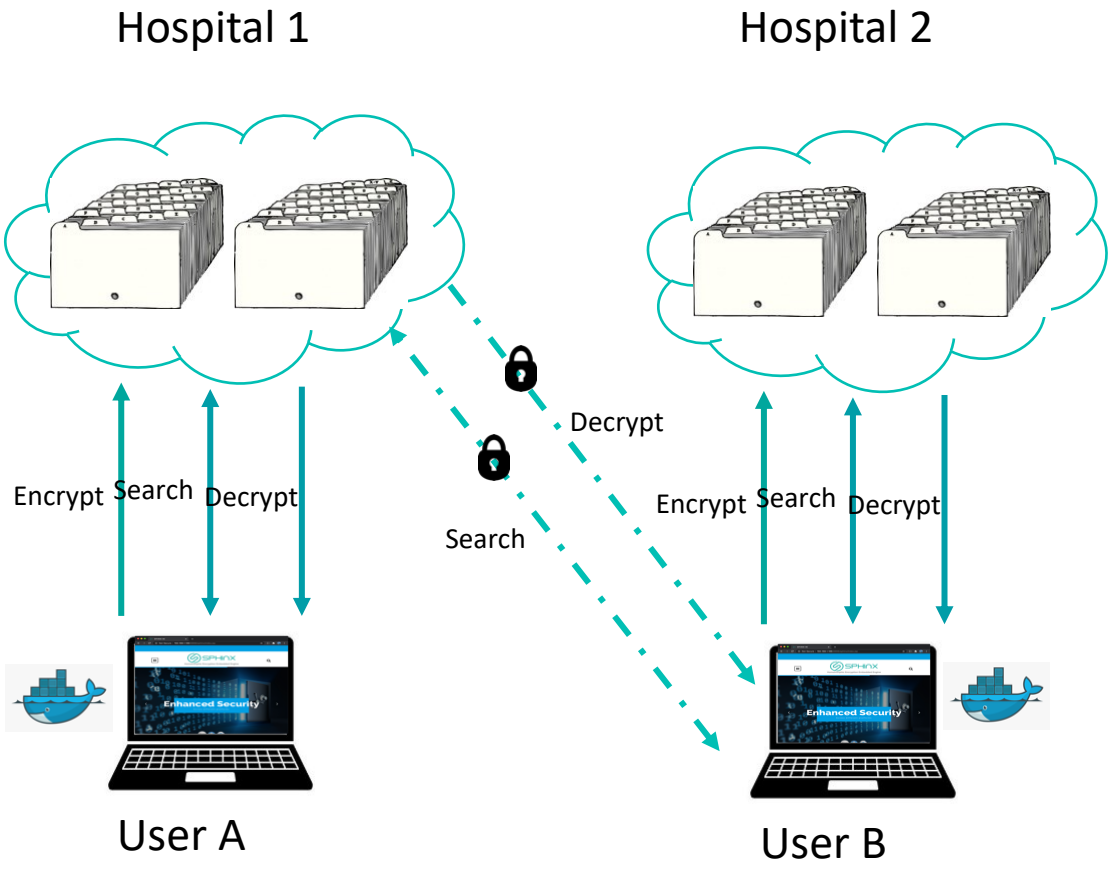

Figure 1 : HE-based searchable encryption

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

3.1.2 Double-Sided blinded process

In current times, it is eminent to have a secure way of sharing healthcare information among geographically distant entities. The double-sided blinded process of the HE solution provides entities with this feature. When using this feature, each entity encrypts their own datasets and then allows different entities to perform search operations on their data. For instance, in Fig 1, *User A* encrypts data and stores it on its repository, *User B* also stores its encrypted data on the repository. Both users allow each other to perform a search. *User B* creates a trapdoor using the public key of *User A* and the search query. This trapdoor is then sent to be processed at the data repository of *User A*. Upon execution of the search request, *User B* gets a binary response, which is true if the search query exists in the database and false when the search query does not exist. Once a true response is received, *User B* can then request the content from *User A*. Throughout this process, *User A* does not know who requested the search and, due to the nature of the trapdoor, is not able to identify what the search query was. In the meantime, *User B* only gets a binary response, which means that it is not able to identify what else is present in the database. This tool provides a controlled environment which, as a result, ensures the security of the data.

3.1.3 Data Anonymization

The data anonymization module of the HE tool provides the SPHINX toolkit with the desired level of data anonymization. Data anonymization is key for ensuring data privacy. This is necessary to ensure compliance with the EU General Data Protection Regulation (GDPR) [8].

When performing data anonymization, one needs to take into account the different data features that exist in a data string. Any dataset can be divided into three broad categories: identifiers, quasi-identifiers and data [9]. Identifiers are all parameters in the data that can be directly linked to an individual. These are the key metrics and they need to be removed from the data to ensure complete data anonymization, this includes but is not limited to name, e-mail address, IP address, etc. Quasi-identifiers are parameters that do not hold much value when used independently but, when used with other quasi-identifiers, can be linked to an individual. Most data anonymization approaches such as k-anonymity, l-diversity and t-closeness work on quasi-identifier values [10]. The remaining data are left untouched and it can be described as everything else that is independent from individual users and could be generated from anyone. In the realm of network traffic, where the key idea is to identify any potential threats to the system, true data anonymization is not feasible for all scenarios and for this we resort to Pseudo-anonymization.

Pseudo-anonymization refers to a data anonymization approach where a key that can be used to de-anonymize the dataset exists [9]. This means that the data are anonymized for everyone else but there still exists a key that can de-anonymize the dataset. This is a viable solution when dealing with network traffic that is being analyzed for threat analysis and intruder detection.

The data anonymization tool developed for the SPHINX tool takes as input network traffic information. It processes all information and separates the identifiers from the rest of the dataset. These identifiers are then sent to the HE-based data anonymization module which encrypts all data and stores it in a repository. The tool creates some surrogate identifiers and replaces the original identifiers with these surrogate values. The data is then compiled and sent back from the tool. The sole reason for the use of the surrogate values is to ensure symmetry in the dataset. This keeps the data useable for the rest of the modules in the SPHINX solution. As shown in Fig 2, the data anonymization tool provides two extra interfaces, IP search and data de-anonymization. The IP search tool allows users to search in the database while knowing the original IP address that they want to search for. The data de-anonymization interface takes as input the surrogate value and returns the original value that was anonymized against that surrogate value.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

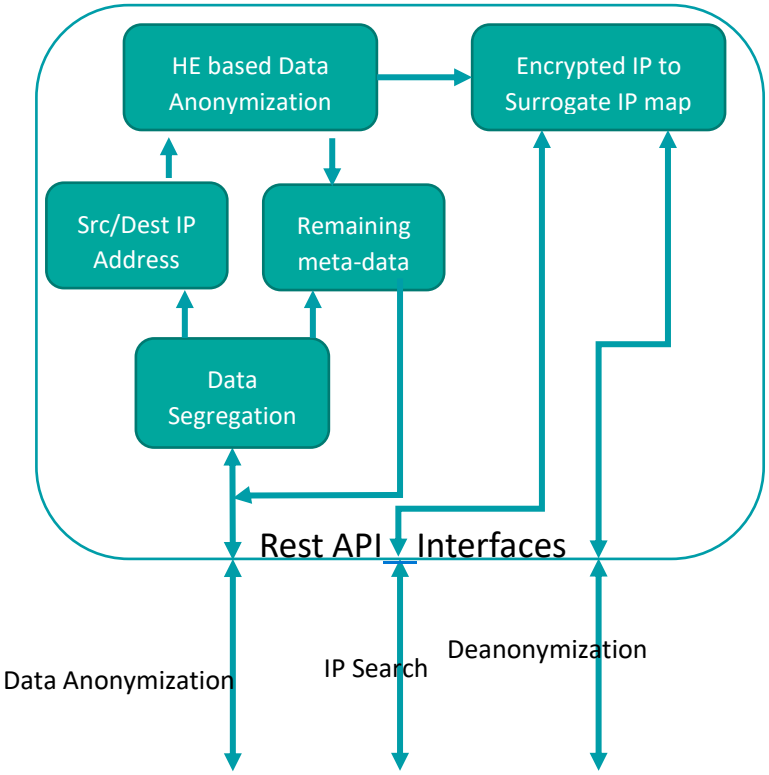

Figure 2: Data Anonymization

3.1.4 Architecture and Design

The Homomorphic Encryption embedded engine plays a vital role in the SPHINX solution for providing both security and privacy. It provides security in the form of encryption and allows entities to search in the encrypted domain. Moreover, it provides privacy with the aid of the data anonymization tool that anonymizes all network traffic information and ensures that user personal data is not revealed. Both these tasks are achieved in a seamless manner as data is shared with the HE tool from the Anonymization and Privacy tool as shown in the next section.

3.1.4.1 External Interfaces of the HE tool

This section is an excerpt from D2.3 where the interfaces are specified for the HE tool. These are added here to better illustrate how different components interact with the HE tool. This facilitates in explaining how the tool operates.

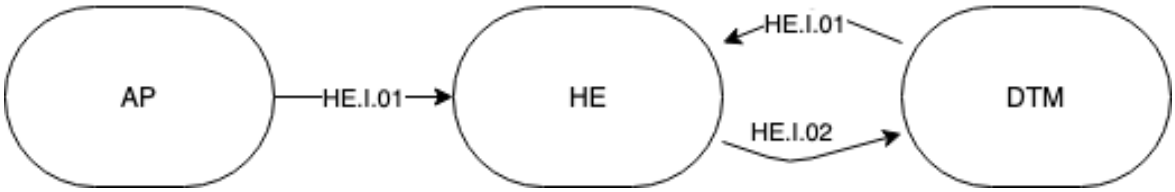

Figure 3: SPHINX HE Component Diagram

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Detailed Technical Specifications

Based on the VOLERE methodology as adapted by the SPHINX Action, the technical requirements/specifications for the HE component are as follows.

HE shall enable storing sensitive data in encrypted format.	
Requirement ID	HE-F-010
Requirement Type	Functional Specifications
Use Cases	-
Dependencies	-
Customer Value	5
Description and Rationale	Information to be stored in SPHINX may contain sensitive and personal data. To protect this data from unauthorised or unnecessary access, sensitive information is be stored in encrypted format using homomorphic encryption.

HE shall provide a secure mechanism to perform searches in and retrieve results from sensitive repositories.	
Requirement ID	HE-F-020
Requirement Type	Functional Specifications
Use Cases	-
Dependencies	-
Customer Value	5
Description and Rationale	Instead of granting unnecessary access to whole data repositories, HE allows that a search is performed on the encrypted stored data and only the data/files containing the desired content are downloaded for further processing.

Table 1: HE Specifications

Interface Specifications

The interfaces applicable to the SPHINX HE component are:

- **HE.I.01: HE Anonymisation Interface**
  This interface is provided by the HE in order to allow homomorphic encryption operations on sensitive data.
  - **Input**: Sensitive Data;
  - **Output**: Encrypted data.

  Related Interfaces: DTM.I.04; AP.I.04.
- **HE.I.02: HE Search Operations Interface**
  This interface is provided by the HE in order to allow homomorphic encryption searches on repositories containing sensitive data.
  - **Input**: Search query;
  - **Output**: List of files (matching query).

  Related Interface: DTM.I.04.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Component Interfaces			
Interface ID	**Involved Components**	**Components Relation**	**Interface Content**
HE.I.01	HE and DTM	The HE provides an encryption service that allows to encrypt the sensitive traffic data of the DTM component.	Sensitive traffic data.
HE.I.01	HE and AP	The HE provides an encryption service that allows to encrypt the sensitive personal data of the AP component.	Personal data.
HE.I.02	HE and DTM	The HE provides a query service that allows to search encrypted traffic data of the DTM component.	Encrypted traffic data.

Table 2: SPHINX HE Interface Specifications

Third-party APIs

The following third-party APIs will be made accessible:

- **HE.API.01: HE Anonymisation Interface**
  This interface is provided by the HE in order to allow homomorphic encryption operations on sensitive data.
  - **Input**: Sensitive Data;
  - **Output**: Encrypted data.

  Interface Relation: HE.I.01.
- **HE.02: HE Search Operations Interface**
  This interface is provided by the HE in order to allow homomorphic encryption searches on repositories containing sensitive data.
  - **Input**: Search query;
  - **Output**: List of files (matching query).

  Interface Relation: HE.I.02.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

4 Design & Development

4.1.1 Interfaces and integration

The Homomorphic Encryption Embedded Engine consists of three interfaces, the dashboard, the client-side module and the healthcare database module. The healthcare database is also referred to as the cloud module in the document. All these components are interlinked and have their own defined characteristics.

4.1.1.1 Dashboard

The dashboard acts as the control panel for the HE tool. SPHINX users interact with this dashboard for the purpose of performing encryption, decryption, search and data anonymization tasks. The dashboard is running off the client-side module and is a web-based tool. The web-based nature of the dashboard makes it operating system and device independent, so that anyone having access to a web browser can use the tool. The dashboard is accessible via the web.

4.1.1.2 Client

The Client module is the key component of the HE tool. This is where all the tasks are executed. The client module itself has two major components: the searchable encryption module and the data anonymization module.

- Searchable Encryption Module:

The searchable encryption module consists of a Springboot-based Java Servlet programming implementation coupled with a C-based code using a Java Native Interface. This searchable encryption module is then dockerized for ease of execution. This module consists of the different set of functions that are needed for performing search in the encrypted domain and the double-sided blinded process. The functions performed at the client module are shown in Fig 4 and a brief description of all these modules is as follows:

- Key Generation:

Whenever a new user registers with the HE tool, the key generation module is executed and it generates a set of public and private keys. These keys are stored on the client hardware with the help of key hashing. Whenever a registered user logs into the system, these keys are used for the purpose of encryption, decryption and search execution.

- Searchable Cipher:

This module involves the creation of a searchable cipher. Whenever a user wants to encrypt a file, that file is sent to this module where it exploits the homomorphic nature of the RSA algorithm and creates a searchable cipher.

- AES Encryption:

In parallel to the searchable cipher creation, the HE tool uses AES encryption to encrypt all incoming files. This ensures that when a search is performed there is content to be downloaded. This AES encryption feature helps reduce the computational burden on the HE tool. Both the AES encrypted content and the searchable cipher are sent to the healthcare database, where they are stored for later use.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

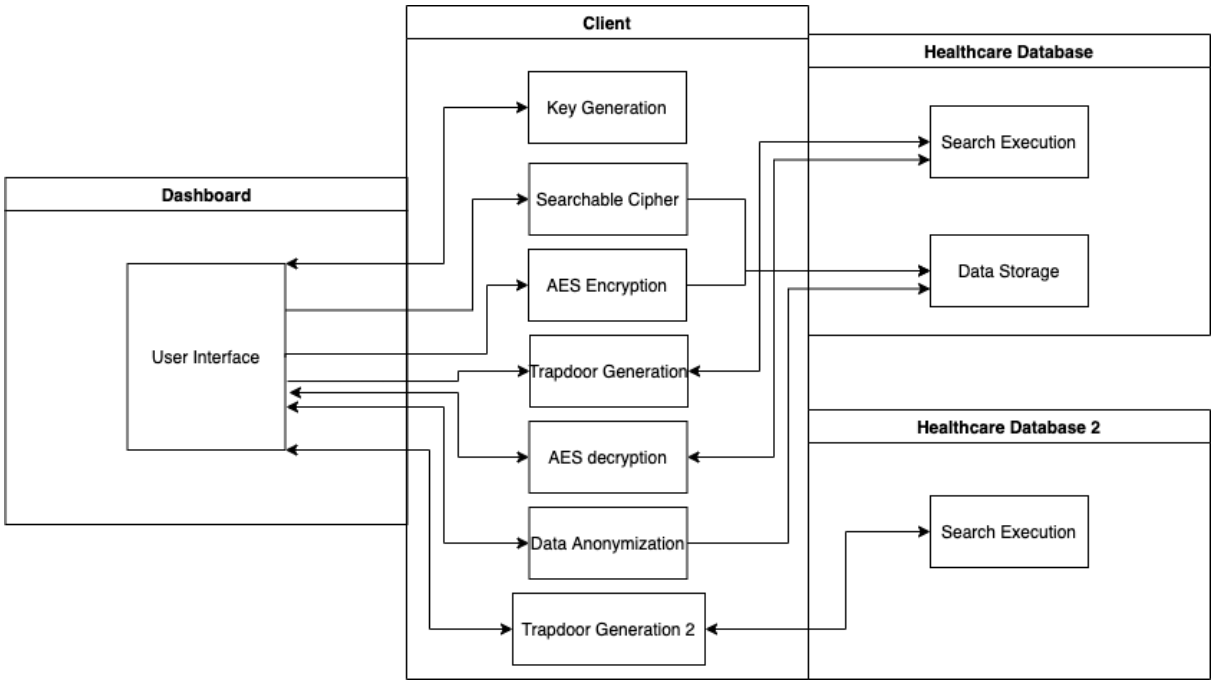

Figure 4:HE module division

- Trapdoor Generation

The trapdoor generation module is needed to perform encrypted searches. Whenever a user wants to search for some content in the encrypted domain, the search query is shared with the Trapdoor Generation module. This module then uses the private key of the user and the search query to create an executable search query. This is shared with the cloud module where a search is performed on the database.

- AES Decryption

When the search query is sent to the healthcare database, it returns the name of the file that contains the desired content. Once the search query is received, the AES decryption module can be used to decrypt the file. This decryption works seamlessly and is computationally less expensive.

- Trapdoor Generation 2

The second trapdoor generation module is there for creation of search queries for the double-sided blinded process. When a user wants to search in a different repository, this module takes the public key associated with the secondary repository and uses it to create a new trapdoor. Similarly to a local search, this search also returns the name of the file which has the search content, thus informing the user if the query exists or not.

- Data Anonymization module

The data anonymization module is a Java Maven based Rest-API implementation. This module is then dockerized to ensure ease of deployment/use. The data anonymization module has three sub-components, and all of these are executed on the client interface. When a user wants to anonymize any network traffic information, they opt for the anonymization module. This module takes network information and then anonymizes all personal information. The data anonymization module allows users to search the content using the actual IP address and also allows one to search using the surrogate values.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

4.1.1.3 Healthcare database

The healthcare database provides the right facility for the users to store information. Once data is encrypted all data is stored on this database. This is later used for search execution. When a user searches for some content, the trapdoor is sent to the database, where it is executed, and the response of the search is then returned to the client interface. This is later shared with the user through the dashboard. This database also executes the searches for the double-sided blinded process.

4.1.2 Swagger definitions

The Homomorphic Encryption tool is employed with the Swagger-API for consistence and ease of use. The Swagger-API provides the right set of documentation that facilitates other modules of the SPHINX tool to identify the inputs and outputs of the HE tool. The API's responses are uniform and structured in Json format. The Swagger-API definition for the HE tool is presented in Fig 5.

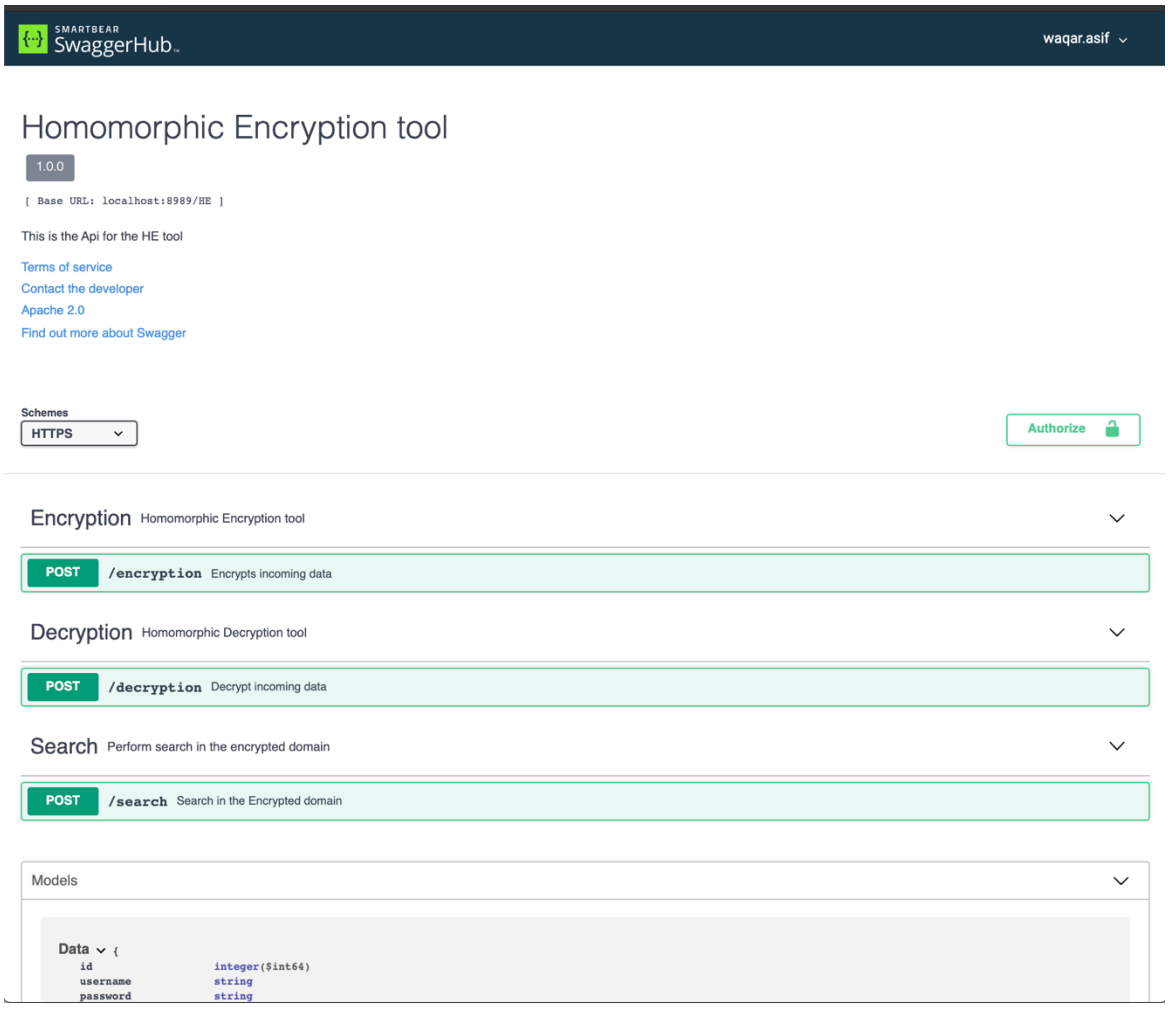

Figure 5: Swagger-Api definition for the HE tool

4.1.3 Test cases

In this section, we explain in detail how the component works. The demo in this demonstrator contains screenshots of an example running in the SPHINX solution. The user can access the HE solution using a web browser with the IP address of the client with an extension of SPHINX added to it. In our deployment, the HE tool is accessed using 192.168.1.189:8888/sphinx/ as shown in Fig 6. The main page of the interface provides

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

the user with an introduction of the tool. It provides options to the user of accessing the different services, reading blogs relevant to the HE tool, accessing the Twitter handle associated with the SPHINX project or going to project website by clicking the SPHINX project icon. The main page also allows users to search for the relevant content in the HE website.

Figure 6: Front end

4.1.3.1 Homomorphic Encryption and Local Database Search

The first service that the tool offers is to encrypt, decrypt and then search in the local repository. This service is accessed using the 192.168.1.189:8888/sphinx/it_service_list.jsp as shown in Fig 7.

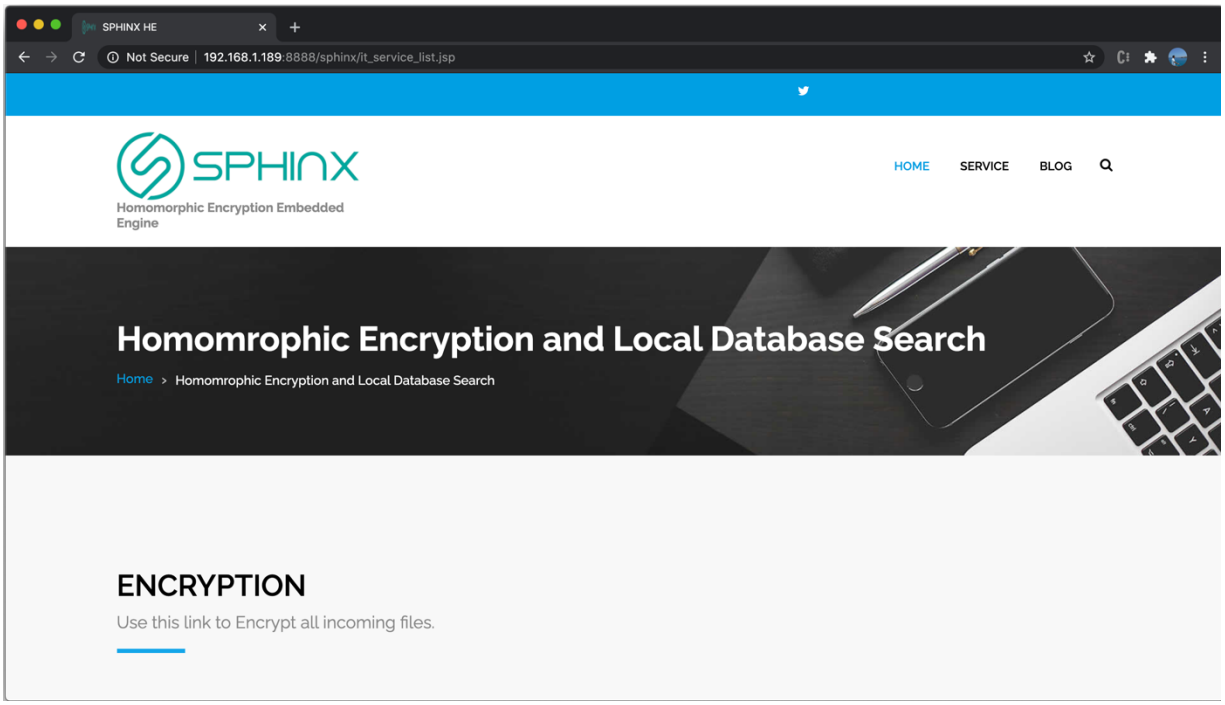

Figure 7: Front end for local search

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

4.1.3.1.1 Encryption

The encryption module allows users to select a number of files and then encrypt them. The encryption module also provides the feature for downloading the encrypted files for viewing purposes as shown in Fig 8.

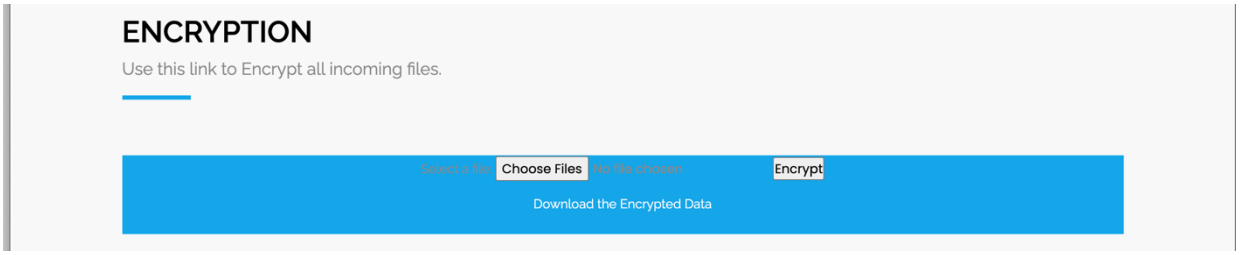

Figure 8: File Encryption and Download option

The file selection option, as shown in Fig 9, allows users to select only .txt files, as the tool currently only supports text files. The dialog box allows users to select multiple files, which are later sent for encryption.

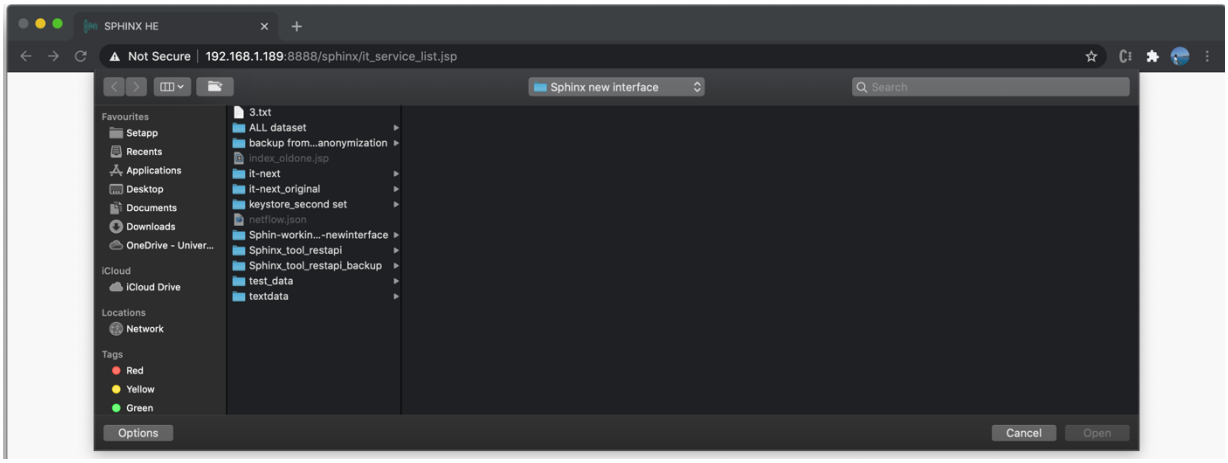

Figure 9: File selection dialog box

As mentioned earlier, the encryption tool produces an AES encrypted file and a searchable cipher. Both these files are stored onto the cloud, which is latter used for search and decryption processes. Fig 10 shows a sample plain text file which is alphanumeric in nature. Fig 11 and Fig 12 present the respective searchable cipher and the AES encrypted files.

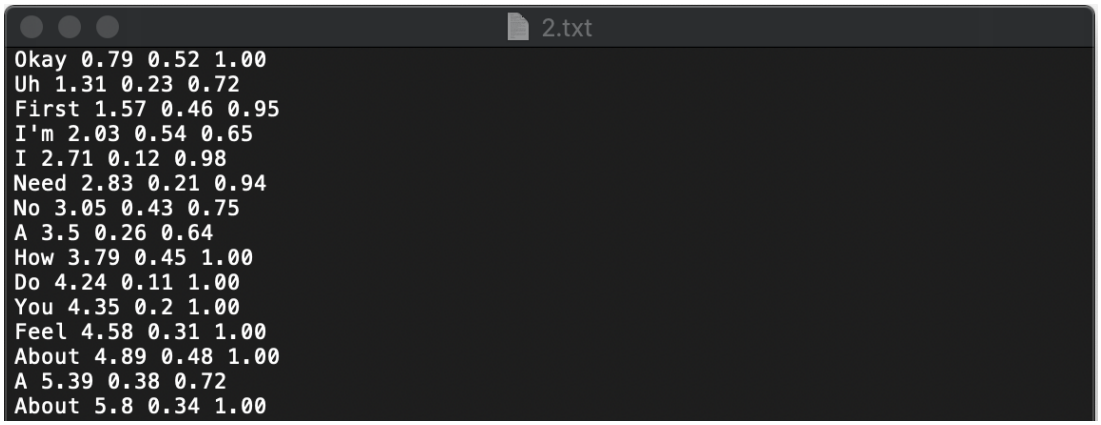

Figure 10: Sample, alphanumeric plain text file

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

Figure 11: Searchable Cipher created after HE encryption

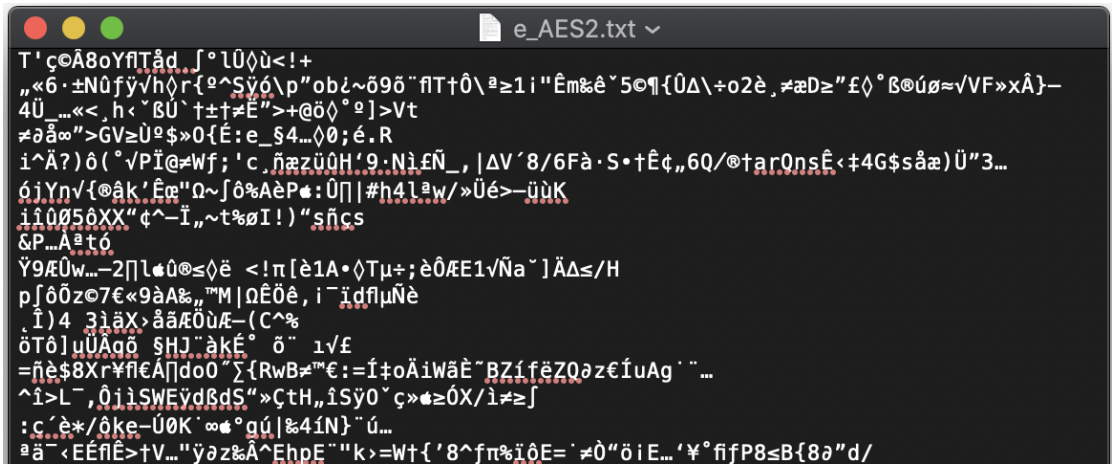

Figure 12: AES encrypted file

Upon successful encryption, the user is shown a success message by the tool as shown in Fig 13.

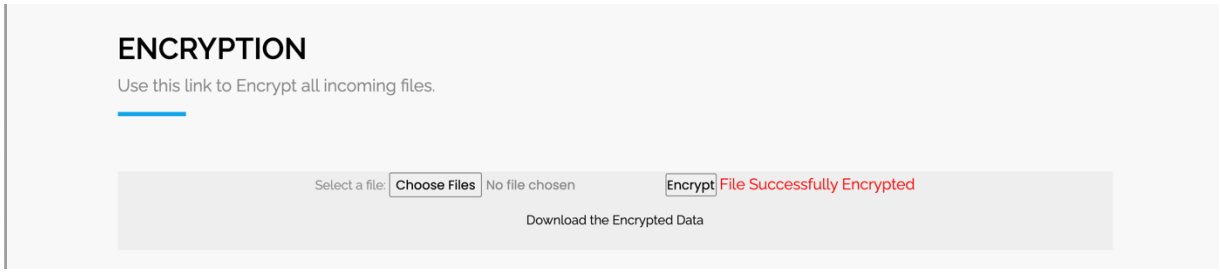

Figure 13: Encryption successful message

4.1.3.1.2 Search

The search option, as shown in Fig 14, allows users to input a search query which is then transformed into a trapdoor at the backend. The trapdoor is then shared with the healthcare database where the search query is executed.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

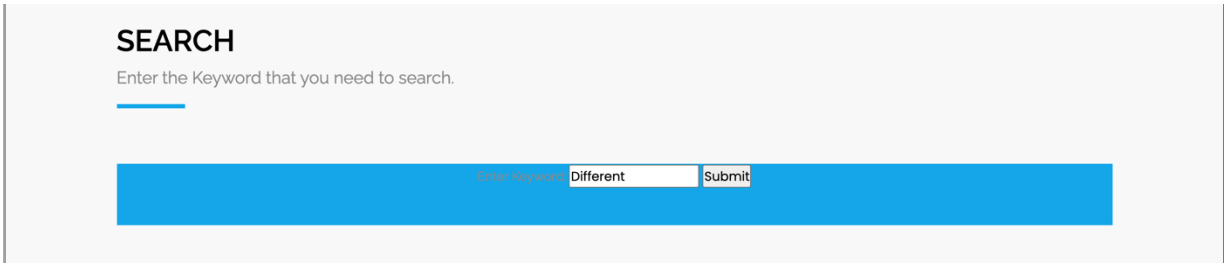

Figure 14: Search query input

Upon successful execution of a search, the search response is received in the form of a file name. This appears in red along with a search completion message, as shown in Fig 15. In case the search query does not exist in the database, the tool returns a null message stating that the search content does not exist.

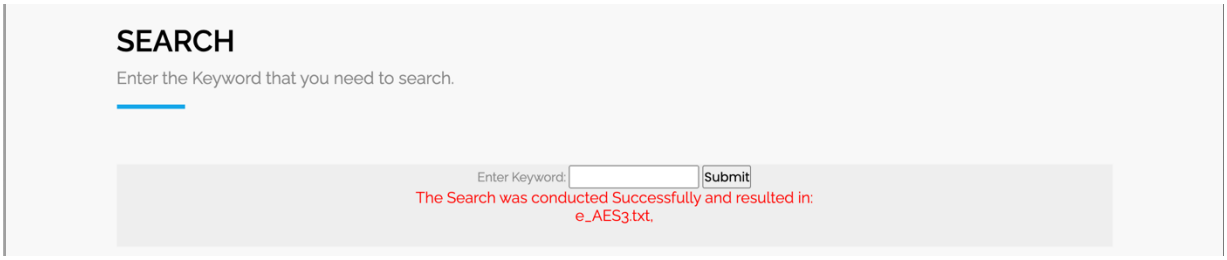

Figure 15: Search response

4.1.3.1.3 Decrypt

The decrypt option allows the users to input the file name that needs to be decrypted. Upon successful search, the search end point returns a file name. This filename, when plugged into the decrypt endpoint, would result in the right file being decrypted. This file can then be downloaded using the "Download the Decrypted Data" URL as shown in Fig 16.

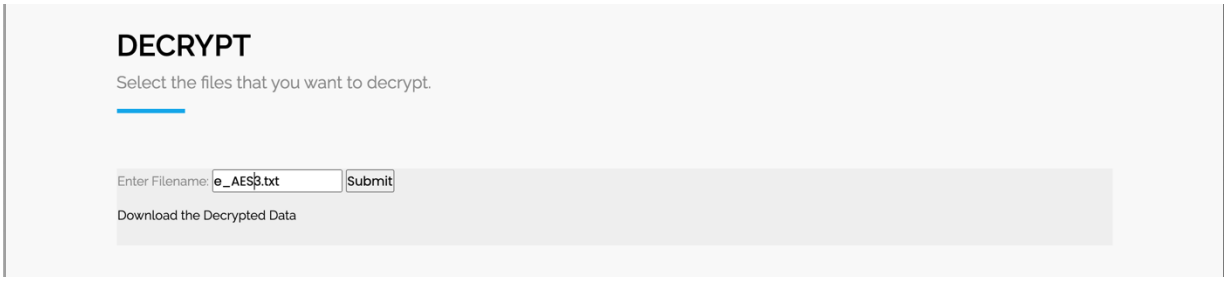

Figure 16: File Decryption

4.1.3.2 Search in External Databases

The search in external databases end point allows users to use the double-sided blinded search capability. This allows entities to search in each other database without revealing what else is present in the database. This component can be accessed through the homepage of the tool or from the URL link 192.168.1.189:8888/sphinx/ExternalDatabase.jsp as shown in Fig 17.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

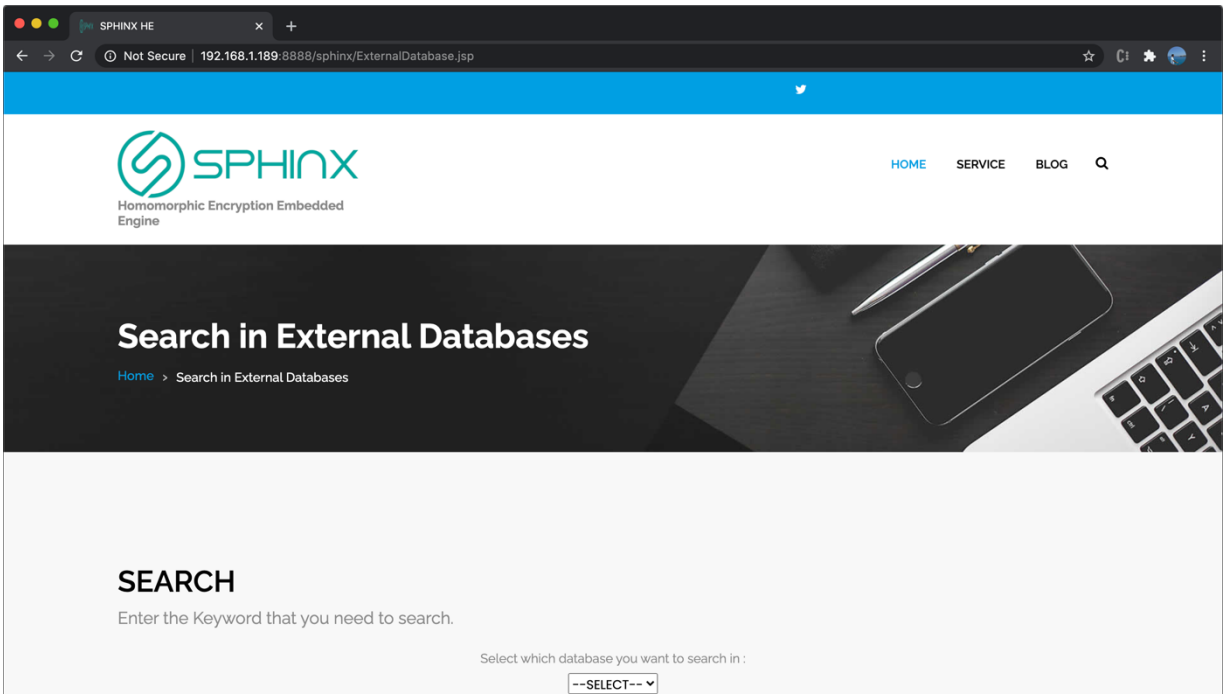

Figure 17: Search in External Databases

4.1.3.2.1 Search

This interface allows users to perform search operations in external databases. In order for this to be executed, the user needs to select what other repository they need to look into. A drop-down menu provides a list of all the available repositories that one can search in. Once the repository is selected, the user can enter the search query and the tool responds in the form of a file name that contains the search query as shown in Fig 18.

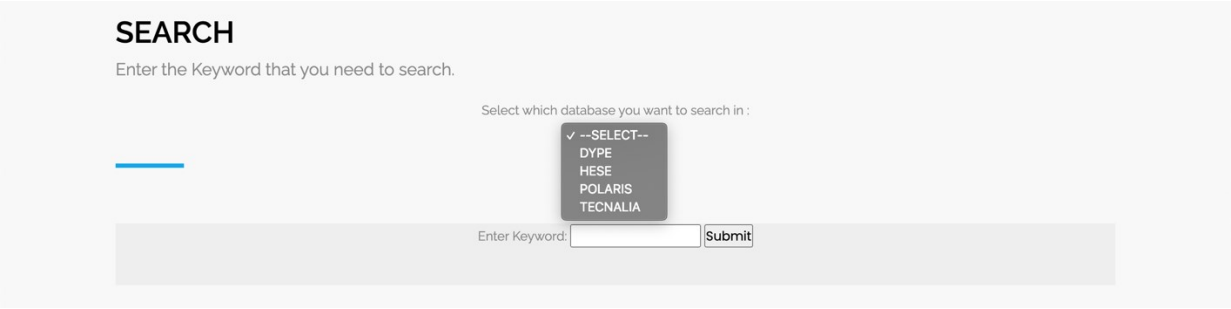

Figure 18: Search in external databases

4.1.3.2.2 Decrypt:

The Decrypt end point is conditional on the capability of decryption. If the external database provides the capability of decryption, then this end point would allow the users to plug in the file name that they want to decrypt. This end point decrypts the file and prints a “File successfully Decrypted” message as shown in Fig 19.

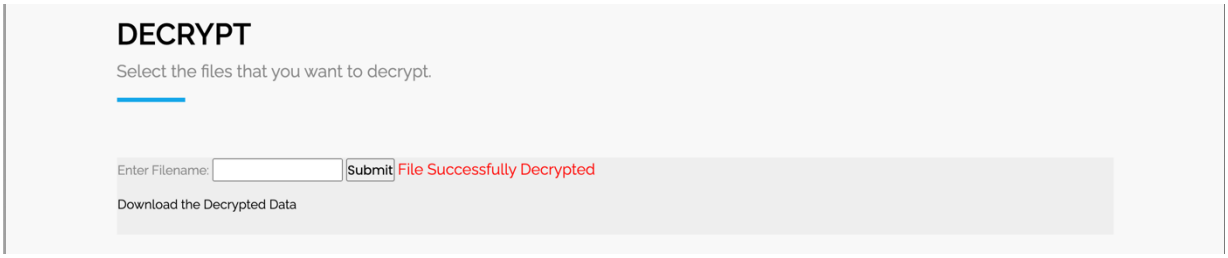

Figure 19: File Decryption for external databases

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

4.1.3.3 Data Anonymization:

The data anonymization end point allows users to anonymize network traffic information. This ensures data privacy. The tool also provides pseudo-anonymization end points, which can be exploited to identify the true identity of the source of the network traffic. This component can be accessed through the homepage of the tool or from the URL link 192.168.1.189:8888/sphinx/anonymization.jsp as shown in Fig 20.

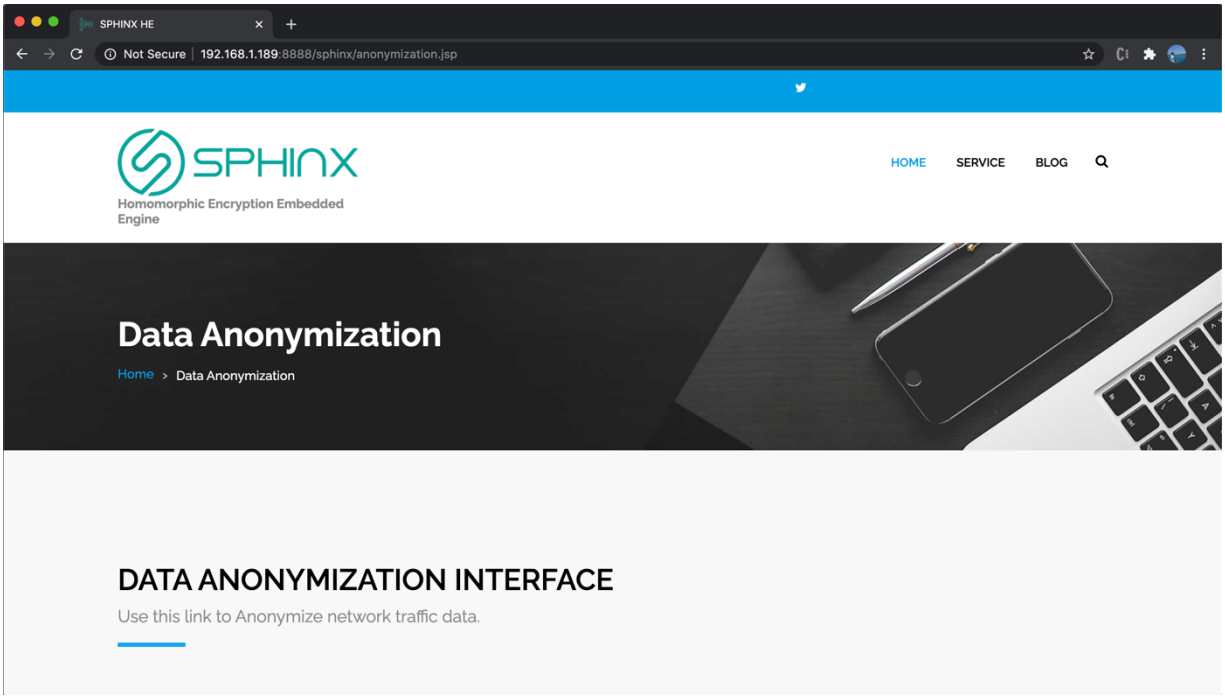

Figure 20: Data anonymization interface

4.1.3.3.1 Data Anonymization Interface

The data anonymization interface takes as input network traffic information and anonymizes all network identifiers. These include source and destination IP addresses. These IP addresses are replaced with surrogate values thus maintaining the structure of network traffic data. The surrogate values are mapped to the encrypted version of the original IP addresses for latter decryption purposes. This tool works as a back-end tool and for the purpose of this demo, the response is printed as a response to the anonymization request as shown in Fig 21.

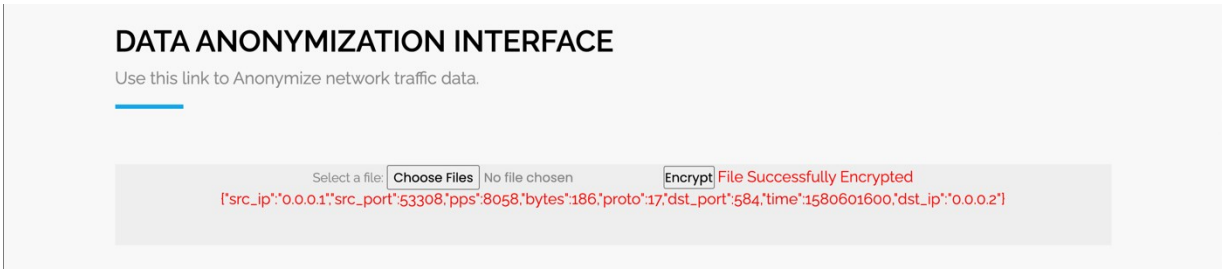

Figure 21: Data Anonymization response

4.1.3.3.2 Search

The search interface in the data anonymization interface allows users to search for the identity of the anonymized IP address. This includes searching from either the surrogate IP or searching using the original IP address. When the user plugs in the original IP address, the tool converts this into an encrypted cipher using Homomorphic Encryption and then maps it to the hashed map of the surrogate IP address. The tool returns the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

surrogate IP address that is used to represent the original IP address. The user can search for this surrogate IP address to identify all network traffic information relevant to this search. The response of the search is shown in Fig 22.

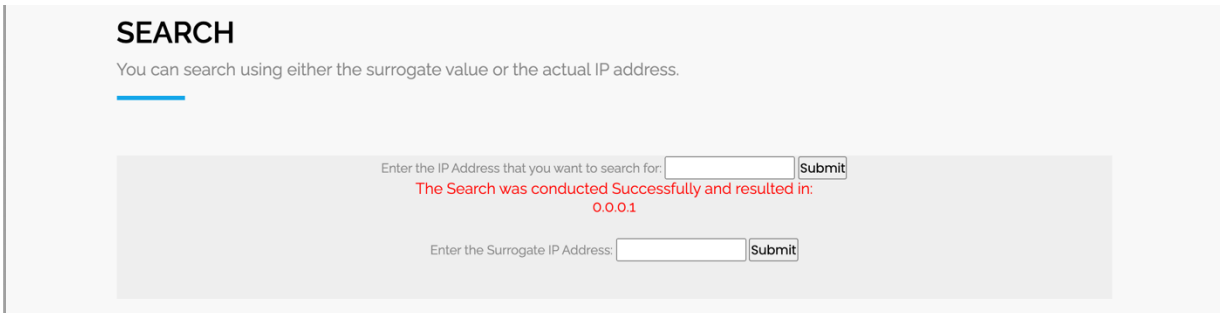

Figure 22: Network IP encrypted search

The tool also allows users to de-anonymize the surrogate IP back to the original IP address. The user can then plug in the surrogate IP address and the tool responds with the original IP as shown in Fig 23.

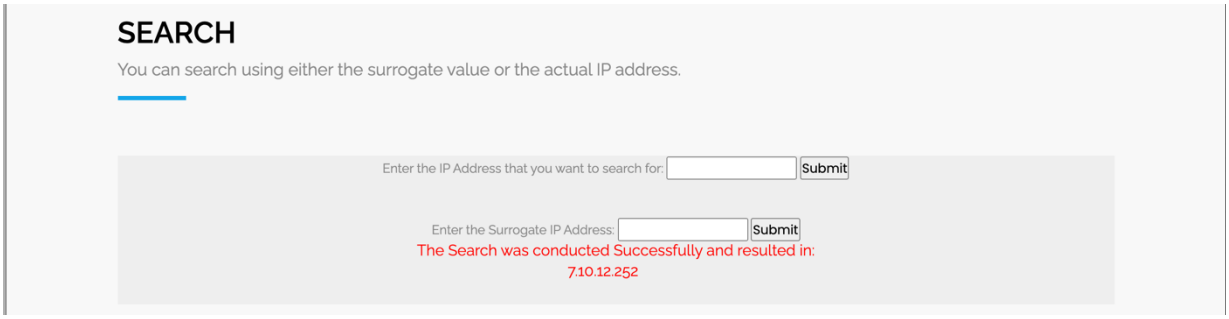

Figure 23: Network IP de-anonymization

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

5 Conclusion

The homomorphic encryption embedded engine provides both security and privacy. The security aspect is provided with the help of a searchable encryption tool that allows users to search in their own encrypted data or perform search in external databases upon getting the necessary privileges. The privacy aspect comes from the data anonymization component which allows users to anonymize network traffic information. The tool itself is rigorously tested and is ready for the next stage of integration. This deliverable covers the current state of the tool and highlights all the features that it offers.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

6 References

[1] Gentry, C. and Boneh, D., 2009. *A fully homomorphic encryption scheme* (Vol. 20, No. 9, pp. 1-209). Stanford: Stanford university.

[2] Boneh, D., Gentry, C., Halevi, S., Wang, F. and Wu, D.J., 2013, June. Private database queries using somewhat homomorphic encryption. In *International Conference on Applied Cryptography and Network Security* (pp. 102-118). Springer, Berlin, Heidelberg.

[3] Sha, P. and Zhu, Z., 2016, August. The modification of RSA algorithm to adapt fully homomorphic encryption algorithm in cloud computing. In *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)* (pp. 388-392). IEEE.

[4] Rani, B., 2016. A novice's perception of partial homomorphic encryption schemes. *Indian J. Sci. Technol*, *9*(37), pp.10-18.

[5] Bösch, C., Hartel, P., Jonker, W. and Peter, A., 2014. A survey of provably secure searchable encryption. *ACM Computing Surveys (CSUR)*, *47*(2), pp.1-51.

[6] Zhang, R., Xue, R. and Liu, L., 2017. Searchable encryption for healthcare clouds: a survey. *IEEE Transactions on Services Computing*, *11*(6), pp.978-996.

[7] Wang, B., Song, W., Lou, W. and Hou, Y.T., 2015, April. Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee. In *2015 IEEE Conference on Computer Communications (INFOCOM)* (pp. 2092-2100). IEEE.

[8] General Data Protection Regulation (GDPR) [Online]: Available at: https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/

[9] Asif, W., Ray, I.G., Tahir, S. and Rajarajan, M., 2018, June. Privacy-preserving Anonymization with Restricted Search (PARS) on Social Network Data for Criminal Investigations. In *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)* (pp. 329-334). IEEE.

[10] Machanavajjhala, A., Kifer, D., Gehrke, J. and Venkitasubramaniam, M., 2007. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *1*(1), pp.3-es.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.