

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Work Package	WP 4, Platform Design and Development
Lead Author	Boris Rozenberg, Ron Shmelkin (IBM)
Contributing Author(s)	Beyza Bozdemir, Orhan Ermis, Melek Önen (EURC) Sebastien Canard, Bastien Vialla (ORA) Angel Palomares Perez (ATOS) Tobias Pulls (KAU)
Reviewers	Sebastien Canard (ORA) Angel Palomares Perez (ATOS)
Due date	30.04.2020
Date	30.04.2020
Version	1.0
Dissemination Level	PU (Public)



The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Programme, through the PAPAYA project, under Grant Agreement No. 786767. The content and results of this deliverable reflect the view of the consortium only. The Research Executive Agency is not responsible for any use that may be made of the information it contains.

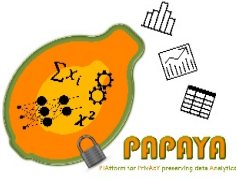


Project No. 786767

**D4.2 – PROGRESS REPORT ON
PLATFORM IMPLEMENTATION AND
PETS INTEGRATION**
Dissemination Level – PU

Revision History

Revision	Date	Editor	Notes
0.1	03.04.2020	Boris Rozenberg (IBM)	ToC
0.2	04.04.2020	All contributing authors	A version for the 1 st internal review
0.3	20.04.2020	All contributing authors	A version after the 1 st internal review
0.4	20.04.2020	All contributing authors	A version after the 2 nd internal review
0.5	27.04.2020	Orhan Ermis (EURC)	Quality Check completed



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Table of Contents

Executive Summary	7
Glossary of Terms	9
1 Introduction	11
1.1 Purpose and Scope	11
1.2 Structure of the Document	11
2 PAPAYA FRAMEWORK	12
2.1 Main Stakeholders	12
2.2 PAPAYA framework architecture	12
3 Platform Core Services	15
3.1 Apply Neural Network Model	15
3.1.1 Privacy-preserving NN classification based on 2PC	15
3.1.2 Privacy-preserving NN classification based on PHE	19
3.1.3 Solution based on Homomorphic Encryption	24
3.1.4 Privacy-preserving NN classification based on hybrid approach	24
3.2 Collaborative Training of Neural Network	26
3.2.1 Main components and their relationships	26
3.2.2 Behavioral analysis:	29
3.2.3 Deployment and configuration	30
3.2.4 Implementation constraints	30
3.2.5 APIs	31
3.2.6 Service integration evaluation	32
3.3 Clustering	32
3.3.1 Privacy-preserving clustering based on 2PC	32
3.3.2 Privacy-preserving clustering based on MinHash	35
3.4 Basic Statistics	41
3.4.1 Privacy-preserving statistics based on Functional Encryption	41
3.4.2 Privacy-preserving Counting using Bloom Filters	41
4 Platform Security and Transparency	44



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

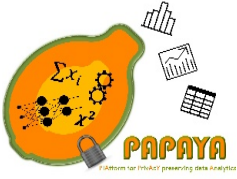
4.1	IAM.....	44
4.1.1	Main components and their relationships.....	44
4.1.2	Deployment and configuration	45
4.1.3	Implementation constraints.....	47
4.1.4	APIs.....	47
4.1.5	Service integration evaluation.....	47
4.2	Auditing	47
4.2.1	Platform auditing.....	47
4.2.2	Agent auditing.....	49
4.3	Key Manager.....	50
4.3.1	Main components and their relationships.....	50
4.3.2	Deployment and configuration	50
4.3.3	Implementation constraints.....	50
4.3.4	APIs.....	50
4.3.5	Integration evaluation	62
5	PAPAYA Dashboards	63
5.1	Platform Dashboard.....	63
5.1.1	Main components and their relationships.....	63
5.1.2	Deployment and configuration	63
5.1.3	Implementation constraints.....	63
5.1.4	APIs.....	63
5.1.5	Integration evaluation	64
5.2	Agent Dashboard	64
5.2.1	Overview	64
5.2.2	Integration evaluation	64
6	Data Subject Toolbox.....	65
6.1	Explaining Privacy-preserving Analytics	65
6.1.1	Overview	65
6.1.2	Integration evaluation	65
6.2	Data Disclosure Visualization Tool	66
6.2.1	Overview	66



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

6.2.2	Integration evaluation	66
6.3	Annotated Log View Tool.....	66
6.3.1	Overview	66
6.3.2	Integration evaluation	67
6.4	Privacy Engine	67
6.4.1	Main components and their relationships.....	67
6.4.2	Deployment and configuration	67
6.4.3	Implementation constraints.....	68
6.4.4	APIs.....	68
6.4.5	Integration evaluation	68
7	Platform Deployment	69
7.1	Platform initialization and platform dashboard deployment	69
7.2	Service upload and deployment.....	69
8	Conclusions	72
9	References	73



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

List of Figures

Figure 1 PAPAYA Framework architecture.....	12
Figure 2 Service Integration Evaluation for PP NN Classification based on 2PC.....	19
Figure 3 Service integration evaluation	25
Figure 4: The client agent components	27
Figure 5: The server components.....	28
Figure 6: Collaborative training - sequence diagram.....	29
Figure 7 Server-side component	36
Figure 8 Client-side component	37
Figure 9 Trajectory clustering analysis	38
Figure 10 Privacy-preserving statistics with Bloom Filters.....	43
Figure 11 IAM and Security Proxy in the PAPAYA Platform.....	44
Figure 12 Deployment steps	45
Figure 13: Platform auditing components and the flow of logs.	48



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

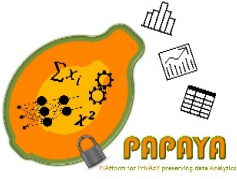
Executive Summary

Rather than several standalone modules, the PAPAYA project aims at developing an integrated platform for privacy preserving data analytics to make them available in a broad spectrum of products and services, with usable, friendly and accessible safeguards options. The main goal of the platform is to be used by service developers to deploy and run privacy-preserving services, and by service consumers, that are interested to employ privacy-preserving analytics. In the previous deliverable (i.e. D4.1 [1]) we presented a first version of the platform functional design, architecture, and deployment. In particular, (1) we described the design of the main platform components that were elicited based on the requirements presented in D2.2 [2], (2) we explained how different privacy preserving primitives being developed in WP3 (see D3.1 [3] and D3.3 [4]) are integrated into the platform in a way that they will be interoperable/compatible with each other and could work together in the integrated platform; and (3) we presented the design of platform dashboards that provide the UI, configuration functionality, and visualization functionality. In this deliverable we describe changes to the services specified in D4.1 [1] and provide description of several new services being created on top of the approaches developed in WP3 and described in detail in D3.3 [4].

As already mentioned in D4.1 [1], PAPAYA platform services are specified based on the four generic usage scenarios, namely upload model, create model, apply model and collaborative training. In upload model, an already trained machine learning (ML) model can be uploaded to the PAPAYA platform when the client wants to delegate the computationally intensive task (which is applying a model on the client's sensitive data) in a privacy-preserving manner. The create model is used when the client is not able to create the ML model; therefore, the PAPAYA platform generates a model on the protected data shared by the client. Apply model is the use case where already uploaded or created model is applied on the client's protected data in a privacy-preserving manner. Finally, in collaborative training, two or more participants perform a ML training collaboratively while preserving the privacy of the training data.

The PAPAYA platform consists of the following groups of services:

- Privacy-preserving analytics defined in the deliverables D3.1 [3] and D3.3 [5] such as classification on Neural Networks, privacy-preserving clustering, privacy-preserving statistics, and privacy-preserving collaborative training of Neural Networks
- Security and transparency services, including the identity access management (IAM) for authentication and authorization services to the different components that will be integrated in the PAPAYA platform, auditing to support auditing and towards being able to hold stakeholders accountable for their use of PAPAYA, and key manager for managing the cryptographic material during the whole lifecycle of the PAPAYA project in the cases where it will be required.
- The PAPAYA platform provides two dashboards for configuration and visualization, namely the platform dashboard and the agent dashboard. The platform dashboard is used for configuration and monitoring the services provided by the platform whereas the agent



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

dashboard is used for viewing the data processing logs from an agent and for showing the configuration of the agent.

- Data subject toolbox services provide a number of mostly independent tools (Explaining Privacy-preserving Analytics, Data Disclosure Visualization, Annotated Log View and Privacy Engine) which provide versatile services related to the data subject privacy. Moreover, the PAPAYA platform provides means to its clients to integrate one or more tools from the provided toolbox to generate an integrated *data subject dashboard*.

As a proof of concept usage, the platform will be deployed on IBM Kubernetes cloud service. In addition to that, all services are designed to be generic in order to be deployed on any other cloud platforms.

The final platform implementation and PETs integration after validation on project use cases, including final version of the dashboard will be given in deliverable D4.3 [6].



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

Glossary of Terms

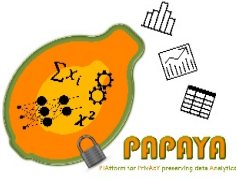
2PC	Two-Party Computation
AA	Auditing Agent
ABY	Arithmetic sharing, Boolean sharing and Yao's garbled circuits framework
AC	Auditing Collector
ALT	Annotated Log view Tool
API	Application Programming Interface
BF	Bloom Filters
BGV	Brakerski-Gentry-Vaikuntanathan homomorphic encryption scheme [7]
BFV	Brakerski/Fan-Vercauteren fully homomorphic encryption scheme [8, 9]
CA	Certificate Authority
CKKS	Cheon-Kim-Kim-Song fully homomorphic encryption scheme [10]
CLI	Command Line Interface
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CR	Container Registry
DB	Data Base
DC	Data Controller
DP	Differential Privacy
DNN	Deep Neural Network
DS	Data Subject
DSRM	Data Subject Rights Manager
DVT	Disclosure Visualization Tool
ECG	Electro cardiogram
ES	ElasticSearch
FC	Fully Connected
FE	Functional Encryption
FHE	Fully Homomorphic Encryption
GRU	Gated Recurrent Unit
HE	Homomorphic Encryption
IAM	Identity Access Manager
KM	Key Manager
ML	Machine Learning
MLP	Multilayer Perceptron
NN	Neural Network
PE	Privacy Engine
PHE	Partially Homomorphic Encryption
PoC	Proof of Concept
PPM	Privacy Preferences Manager
RAM	Random Access Memory



Project No. 786767

**D4.2 – PROGRESS REPORT ON
PLATFORM IMPLEMENTATION AND
PETS INTEGRATION
Dissemination Level – PU**

REST	Representational State Transfer
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SIMD	Single Instruction Multiple Data



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

1 Introduction

1.1 Purpose and Scope

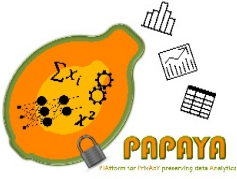
The purpose of this deliverable is to provide a comprehensive description of the PAPAYA platform architecture. This deliverable extends the platform design and architecture already presented in D4.1 [1]. In particular, it provides modifications to already described services as well as specification of several new services developed on top of the analytics developed in WP3 and presented in D3.3 [5]. The intended audience for this document consists of two main groups: (1) service developers, who could use the document to understand the design of the existing services and to develop and deploy their own services; (2) service users, who could use the document to understand how to employ the services available on the platform and how to incorporate them in their applications.

It is important to note that the description of the underlying algorithms employed by the services presented in this document is not in the scope of this deliverable (they are described in D3.1 [3] and in D3.3 [5]).

1.2 Structure of the Document

The rest of the document is organized as follows:

- **Section 2** provides a high-level overview of the PAPAYA framework, including description of main stakeholders.
- **Section 3** describes in detail the core PAPAYA services, which are defined in the deliverable D3.1 [3] and deliverable D3.3 [5]. Provided services in the scope of this deliverable are: (1) Privacy-preserving classification on Neural Networks; (2) Privacy-preserving collaborative training of Neural Networks; (3) Privacy-preserving clustering; and (4) Privacy-preserving statistics.
- **Section 4** gives details about how security and transparency is achieved in the platform, including authentication, authorization, auditing and key management for cryptographic tools (if it is required).
- **Section 5** presents PAPAYA dashboards, namely platform dashboard and agent dashboard.
- **Section 6** dedicated to Data Subject Toolbox, which provides versatile services (Explaining Privacy-preserving Analytics, Data Disclosure Visualization, Annotated Log View and Privacy Engine) related to the data subject privacy. Tools presented in this section can then be used to form a *data subject dashboard*.
- In **Section 7** we describe how to deploy and run all platform components.
- We conclude the deliverable in **Section 8**.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

2 PAPAYA FRAMEWORK

2.1 Main Stakeholders

On a high-level, there are four main stakeholders in the PAPAYA framework:

1. **Platform clients:** stakeholders who wish to perform some analytics in a privacy preserving manner. **Platform clients** can be considered as Data Controllers or external queriers who are allowed to request some analytics results while not being the actual owners of the data.
2. **Platform administrators:** responsible for platform administration purposes such as resource allocation or monitoring.
3. **Service providers:** the author of the services available on the platform.
4. **Data Subjects: end-users (e.g. application user)** of the **Platform clients**.

Details about PAPAYA usage scenarios can be found in D4.1 [1].

2.2 PAPAYA framework architecture

The PAPAYA framework (see Figure 1) is composed of two main groups of components: (1) the platform-side components that will be running on the (non-trusted, but semi honest) Kubernetes cloud server; and (2) client side components, that will be running on trusted client environment.

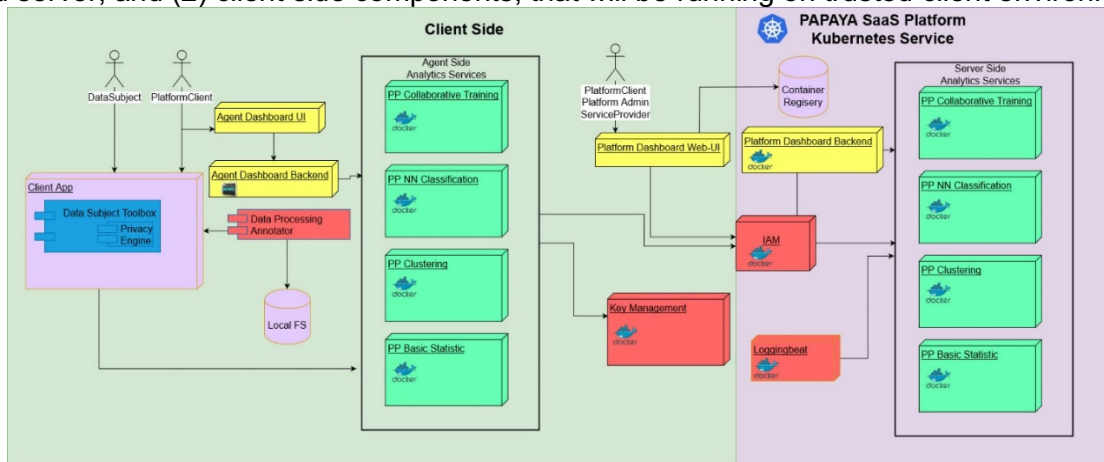


Figure 1 PAPAYA Framework architecture

From the functional perspective, the PAPAYA framework components can be regrouped into the following categories (more details about each category can be found in deliverables D4.1 [1] and D3.1 [3]):

¹ <https://kubernetes.io/>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

- **The privacy-preserving analytics services** (colored with green in Figure 1) which will allow platform clients to perform analytics of interest, in a privacy-preserving manner. In the first version of PAPAYA we will support the following analytics:
 1. Privacy-preserving NN classification. We will provide four services for applying neural network for the purpose of classification in a privacy-preserving manner: (1) 2PC-based; (2) PHE-based; (3) FHE-based; and (4) Hybrid approach. Each one could be preferable than others in different settings, mainly depending on NN architecture
 2. Privacy-preserving collaborative training of NN. The service allows multiple participants to perform a ML training collaboratively, while preserving the privacy of the training data.
 3. Privacy-preserving trajectory clustering. The service provides means to cluster trajectories in a privacy preserving manner.
 4. Privacy-preserving basic statistics. The service provides means for privacy-preserving computation of statistics using functional encryption and privacy-preserving counting using Bloom Filters.

The users of these services are platform clients who can either be Data Controllers or external queriers (in the case for privacy preserving trajectory clustering who are authorized by Data Controllers to request some analytics results. Each service is divided into two parts:

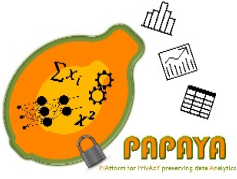
1. Server – responsible for performing analytics of interest on encrypted data and will run on a PAPAYA's Kubernetes cluster.
 2. Agent – responsible for communication with the appropriate server-side component and responsible for managing cryptographic operations for the client. The agent will be downloaded from the Container Registry (CR) as a Docker image and deployed on a client side. Deployment and execution will be under the responsibility of the platform client.
- **The platform security and transparency services** (colored with red in Figure 1) which will provide platform authorization, authentication (Identity and Access Management - IAM), auditing and cryptographic Key Manager (if needed).
 - **The PAPAYA dashboards** (colored with yellow in Figure 1):
 1. **Platform dashboard** will allow: (1) **service providers** to deploy privacy-preserving services; (2) **platform clients** to choose/run privacy preserving services and review operational and auditing logs; and (3) **platform administrators** to configure and manage the platform.
 2. **Agent dashboard** will allow **platform clients** to visualize the configuration of the agent running on the client side and review operational and auditing logs.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

- **Data Subject Toolbox** (colored with blue in Figure 1) consists of a number of mostly independent tools (DS Tool 1: Explaining Privacy-preserving Analytic, DS Tool 2: Data Disclosure Visualization, DS Tool 3: Annotated Log View and DS Tool 4: Privacy Engine), which provide versatile tools for data protection by design by platform clients (acting as data controllers) towards data subjects whose personal data is processed in their services.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

3 Platform Core Services

3.1 Apply Neural Network Model

In this section, we describe several services for applying neural network for the purpose of classification in a privacy-preserving manner. The detailed description of these services can be found in deliverables D3.1 [3] and D3.3 [5].

Prior to using any of the services described in this section, a Neural Network model should be trained locally based on the clear text data with all the required optimization and the dimensionality reduction. After achieving the desired accuracy, the trained model (i.e. architecture and weights) should be saved in a supported format and passed to the service later as described in each of the following subsections.

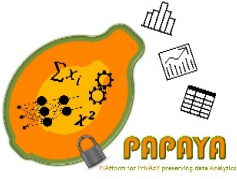
3.1.1 Privacy-preserving NN classification based on 2PC

In this section, updates on APIs and details on the deployment, configuration and implementation of the privacy preserving NN classification based on 2PC service are presented. Since there is no update on the main components, relationships between these components and the behavioral analysis of this service subsections, they remain the same as introduced in D4.1 [1]. Finally, the evaluation of the service integration is overviewed in Section 3.1.1.4.

3.1.1.1 Deployment and configuration

Deployment. As stated in D4.1 [1], the 2PC-based privacy-preserving NN classification based on two-party computation (2PC) service involves two components, namely the client-side and server-side components as stated in D4.1 [1]. Both of these components are deployed as Docker containers. The server-side component uses a particular NN model for arrhythmia classification. The model is also installed during the deployment of server component. Before its deployment, the server component should receive the information of the local IP address (which should be 0.0.0.0 by default because of the use of Flask framework as a RESTful service), local HTTP port number and the TCP port number (any available port number, except the reserved ports by Ubuntu operating system). Note that the local IP address and the port number is required to run ABY server in the server component.

Configuration. This service does not require any specific configuration for the installed libraries, tools and the operating system that will be used for accomplishing the 2PC-based PP NN classification. On the other hand, once the server-side component is deployed to the cloud environment, it will be assigned a new public IP address and a port number. This newly assigned communication information should be stored in the client-side component to execute 2PC computation. Therefore, the one and only configuration that is required for the client-side component, is accomplished by using “init” API call.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.1.1.2 Implementation constraints

The 2PC-based Privacy-preserving NN classification service uses a slightly modified version of Arithmetic-Boolean-Yao (ABY) as a secure 2PC framework. RESTful Web-service requires Python v3 and Flask v1.0.0 (or higher versions).

3.1.1.3 APIs

Server-side Component APIs:

There is only one RESTful API call at the server-side component in this service and this for initiating the classification in the server side (classify/). The rest of the communication between the server-side component and the client-side component is realized by using ABY sockets.

POST classify/

POST /**classify** Classify - Server-side Component

This API call is used for initiating the server-side component for classification.

Parameters

Try it out

Name	Description
Number of signals * required	Number of signals to be classified
integer (path)	<input type="text" value="Number of signals - Number of signals to be classified"/>

Responses

Code	Description	Links
201	Classification has been initiated in the server-side component	No links
400	invalid input, object invalid	No links



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

Client-side Component APIs:

There are two RESTful API calls at the client-side component. The first one is used for initializing the public IP address and the port number of the server (init/) and the second one is used for initiating the classification at the client side. The rest of the communication between the server-side component and the client-side component is realized using ABY sockets.

POST init/

POST
/init
Init - Client-side Component

This API call is used for initializing the port number and the IP address for client-side component.

Parameters
Try it out

Name	Description
IP address * required string (path)	IP address of the server <input type="text" value="IP address - IP address of the server"/>
Port number * required string (path)	port number of the server <input type="text" value="Port number - port number of the server"/>

Responses

Code	Description	Links
201	Server's IP address and the port number has been recorded.	No links
400	invalid input, object invalid	No links



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

POST classify/

POST
/classify
Classify - Client-side Component

This API call is used for uploading the file to be classified.

Parameters
Try it out

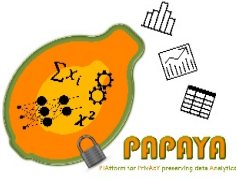
Name	Description
Input file ★ required string (path)	Input file consist ECG signals to be processed. <div> Choose File No file chosen </div>

Responses

Code	Description	Links
201	File has been uploaded for classification.	No links
400	invalid input, object invalid	No links

3.1.1.4 Service integration evaluation

We have implemented each component as a Docker container. The client-side component is deployed on the local machine and the server-side component is deployed on the IBM cloud. After the server-side component is deployed to IBM's cloud, the server-side component's public IP address and the port number are stored in the client-side component via the init/ API call. Later, the file to be processed is uploaded to the client-side component using classify/ API call of the client. Then, the client-side component sends the classification request to the server-side component. Once two components are ready for classification, both parties execute classification



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

using the ABY framework. We have tested our integration by following the steps illustrated in Figure 2 and the modules were executed correctly and output the expected classification result.

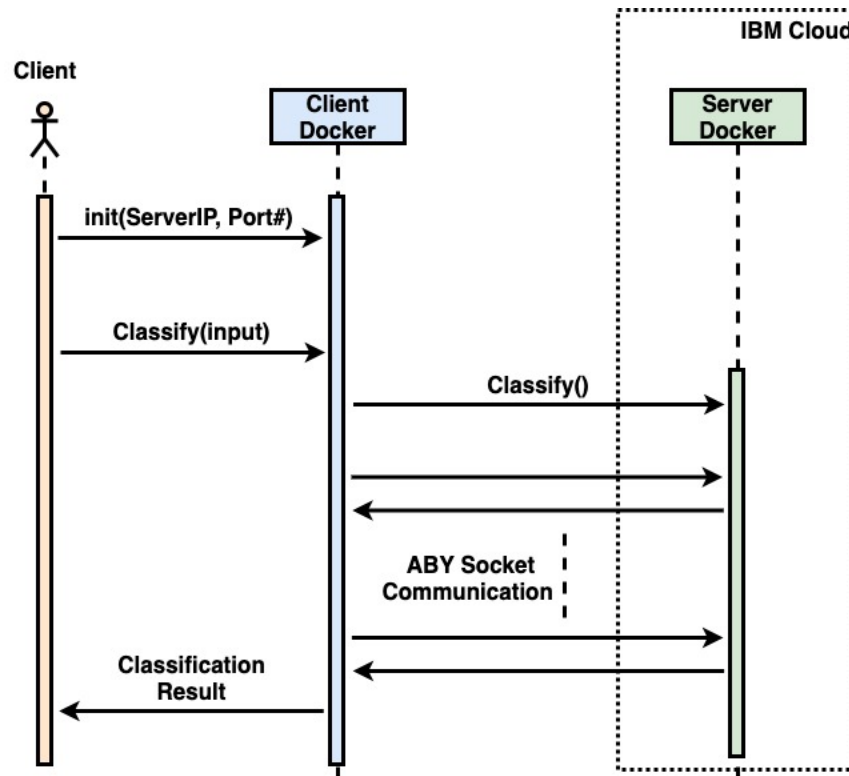


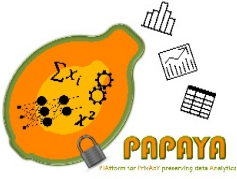
Figure 2 Service Integration Evaluation for PP NN Classification based on 2PC

3.1.2 Privacy-preserving NN classification based on PHE

In this section, updates on APIs and details for the deployment, configuration and implementation of the PHE-based privacy-preserving NN classification-based service are presented. Since there is no update on the main components, relationships between these components and the behavioral analysis of this service subsections, these sections remain the same as introduced in D4.1 [1]. Finally, the evaluation of the service integration for a particular NN model is overviewed in the service integration evaluation Section 3.1.2.4.

3.1.2.1 Deployment and configuration

Deployment. As stated in D4.1 [1], the PHE-based privacy-preserving NN classification service involves two components: the client-side and server-side components. Both of these components are deployed as docker containers. The current version of this service works on particular NN models (ECG classification and MNIST dataset classification). The model is also installed during the deployment of the server-side component. Before its deployment, both the server and the client components may need the information of the local IP address, local HTTP port and local



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

TCP port. Note that the local IP address and the local TCP port number are required to ensure communication between these two parties achieved through socket programming.

Configuration. This service does not require any specific configuration for the installed libraries, tools and the operating system that will be used for accomplishing the PHE-based PP NN classification. On the other hand, once the server-side component is deployed to a cloud environment, it will have a new public IP address and a port number. This newly assigned communication information should be also stored in the client-side component to execute classification. Therefore, the one and only configuration is required for the client-side component, which is realized by using “init/” API call.

3.1.2.2 *Implementation constraints*

The PHE-based privacy-preserving NN classification service uses the Paillier library. Socket programming is used to ensure the communication between the two parties. The REST-full web-service requires Python v3 and Flask v1.0.0 (or higher versions).

3.1.2.3 *APIs*

Server-side component APIs:


There is only one REST-full API call for the server-side component in this service, namely “classify/”. This API call is used for initiating the classification. Once the classification is initiated, then all necessary communications for executing classification are accomplished through socket calls between the server-side and client-side components.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

POST classify/

POST **/classify** Classify - Server-side Component 

This API call is used for initiating the server-side component for classification

Parameters

Try it out

Name	Description
NN model selection * required string (<i>path</i>)	ID of the predefined NN model <div>NN model selection - ID of the predefined NN model</div>

Responses

Code	Description	Links
201	Server has been initiated for [name] NN Model.	No links
400	invalid input, object invalid	No links



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

Client-side component APIs:

There are two REST-full API calls at the client-side component. The first one is used for initializing the public IP address and the port number of the server (init/) and the second one is used to initiate the classification at the client side. Socket calls are used for communication purposes between the server-side and client-side components

POST classify/

POST
/classify Classify - Client-side Component
←

This API call is used for sending the NN model choice and the input file for the classification.

Parameters
Try it out

Name	Description
NN model selection * required string (path)	ID of the predefined NN model <div style="border: 1px solid #ccc; padding: 5px; margin-top: 5px;"> NN model selection - ID of the predefined NN model </div>
Input * required string (path)	ID of the predefined NN model <div style="border: 1px solid #ccc; padding: 5px; margin-top: 5px;"> <div style="display: flex; align-items: center;"> <div style="border: 1px solid #ccc; padding: 2px 5px; margin-right: 5px;">Choose File</div> <div>No file chosen</div> </div> </div>

Responses

Code	Description	Links
201	Client has been initiated for classification.	<i>No links</i>
400	invalid input, object invalid	<i>No links</i>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

POST init/

POST `/init` Init - Client-side Component

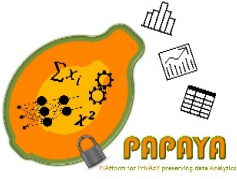
This API call is used for initializing the port number and the IP address for client-side component.

Parameters Try it out

Name	Description
IP address * required string (path)	IP address of the server <input type="text" value="IP address - IP address of the server"/>
Port number * required string (path)	port number of the server <input type="text" value="Port number - port number of the server"/>

Responses

Code	Description	Links
201	Server's IP address and the port number has been recorded.	No links
400	invalid input, object invalid	No links



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.1.2.4 *Service integration evaluation*

Service integration evaluation will be presented in D4.3 [6].

3.1.3 Solution based on Homomorphic Encryption

This solution uses Homomorphic Encryption (HE) in order to build a privacy-preserving neural network inference solution. This solution uses the CKKS scheme [10] implemented in Microsoft, open source, SEAL library [11]. The solution takes a pre-trained neural network composed of the supported layers (Dense, Convolution), and activation functions and provide an encrypted version for secure inference. The detailed design of this service is presented in D4.1 [1].

3.1.3.1 *Service integration evaluation*

Service integration evaluation will be presented in D4.3 [6].

3.1.4 Privacy-preserving NN classification based on hybrid approach

The hybrid solution uses both, the HE and 2PC, in order to build a privacy preserving NN classification framework and maximize the efficiency of classification on deep NN. The solution is *practically* generic, namely, it supports different types of DNN (i.e., MLP, CNN, and RNN) with any number of layers, any number of neurons in each layer and any activation function (from the set of supported activation functions), while the performance still acceptable (grows linearly with the DNN's depth).

We presented a detailed design in D4.1 [1]. In the following subsection we describe how we integrated and evaluated the service.

3.1.4.1 *Service integration evaluation*

We evaluated the service integration as depicted in Figure 3. In particular, we developed: (1) an application (ClientApp) which simulates client's application (in our case it reads ECG samples from a file and classify them using the PAPAYA service); (2) the agent (provides basic cryptographic functionality and service interfaces); and (3) the service (which performs the privacy preserving classification). Each of this component is implemented as a docker container. ClientApp and the agent are deployed on a local machine, while the service is deployed on IBM cloud. More details about the API of each component could be found in D4.1 [1]. We run the classification flow for all the inputs and everything works as expected.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

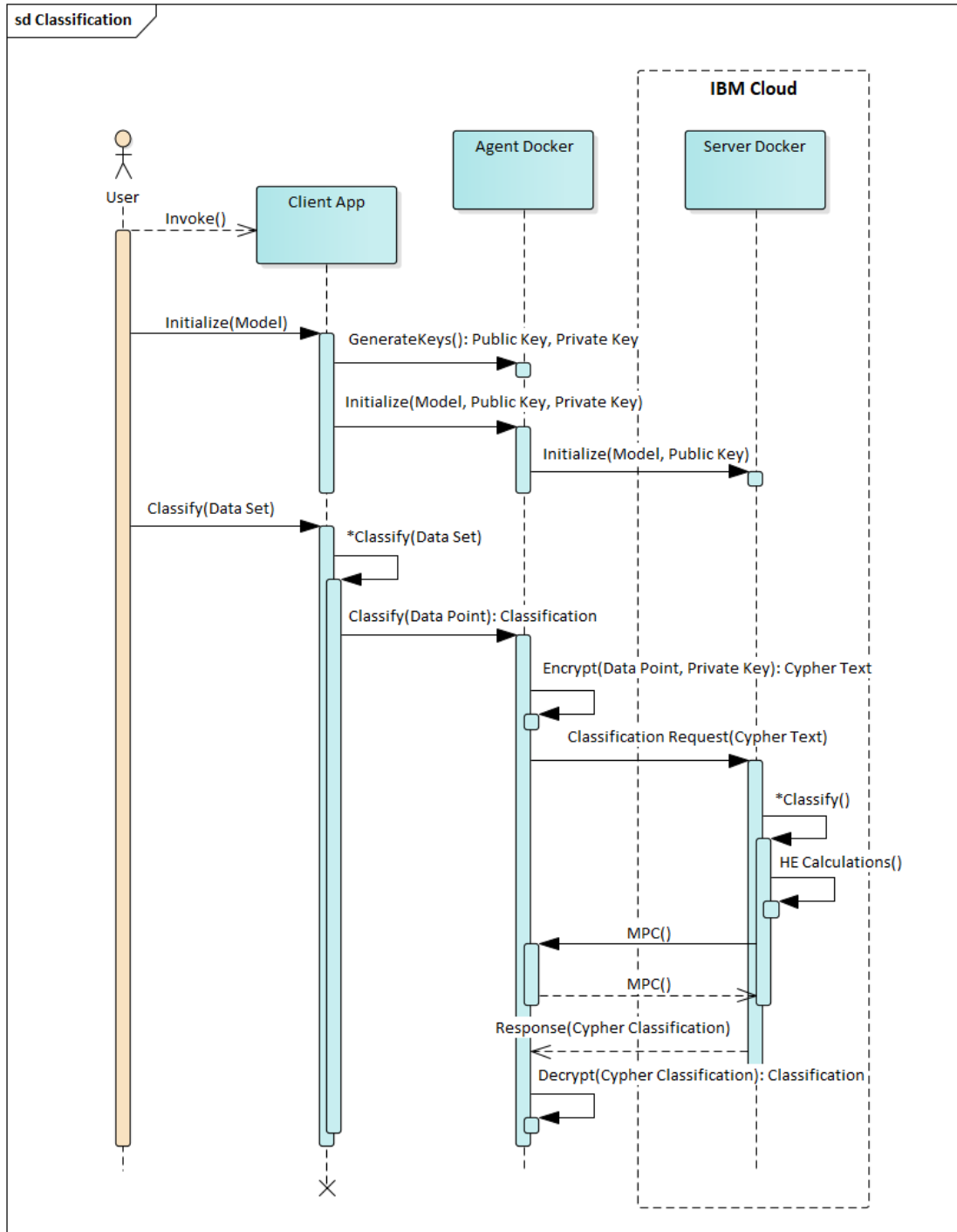
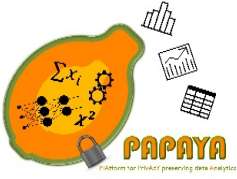


Figure 3 Service integration evaluation



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

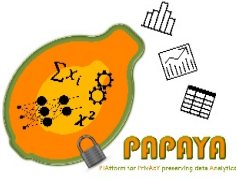
Project No. 786767

3.2 Collaborative Training of Neural Network

Collaborative training of NN allows multiple participants to perform a ML training collaboratively, while preserving the privacy of the training data. In the previous version of this document (D4.1 [1]) we presented a complete specification of the service based on Shokri and Shmatikov approach [12]. Since that, in a course of WP3, we have implemented an additional approach for privacy preserving training of DNN based on the method presented by Abadi et. al. [13] (presented in D3.3 [5]). We integrated this solution into the service as well. There are minor changes in almost all previously presented (in D4.1 [1]) components and APIs. So, we provide a complete specification of the service in this section.

3.2.1 Main components and their relationships

1. The client side – responsible for performing the following functionalities:
 - a. NN training
 - b. Adding Differential Privacy (DP) noise
 - i. Train local DP model
 - ii. Noise updates sent to the server
 - c. Uploading noised gradients or DP weights
 - d. Downloading the model parameters to/from the centralized server.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

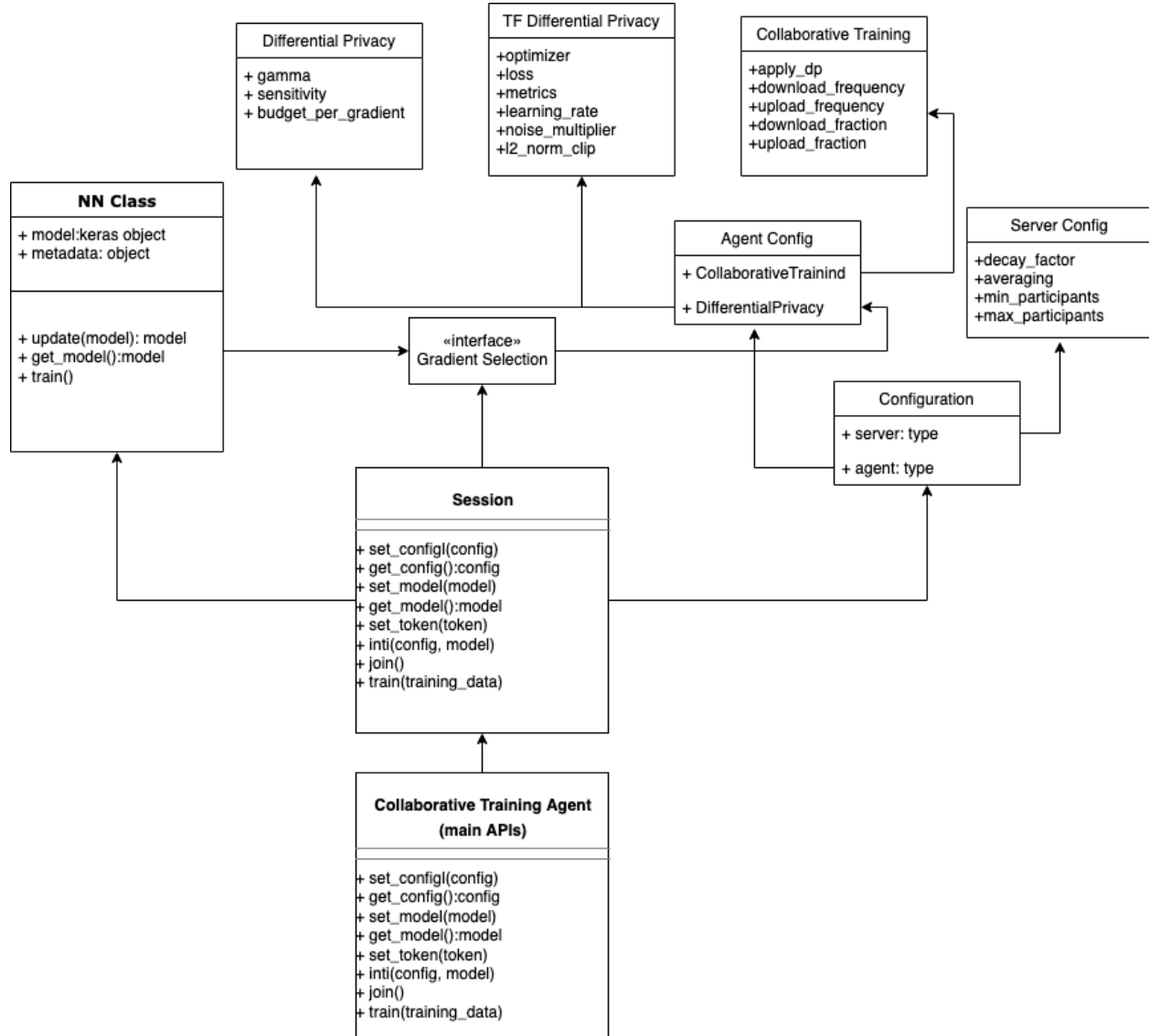


Figure 4: The client agent components

Figure 4 presents the client agent components that will run on the client side (trusted environment).

2. The centralized server (server side) – will provide the following functionality:
 - a. Allow participants to define and download the initial model.
 - b. Aggregate the gradients from the collaborative training participants.
 - i. Manage updates statistics and decay the less relevant updates
 - ii. Updates averaging



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

- c. Allow participants to download the model parameters during the training phase.

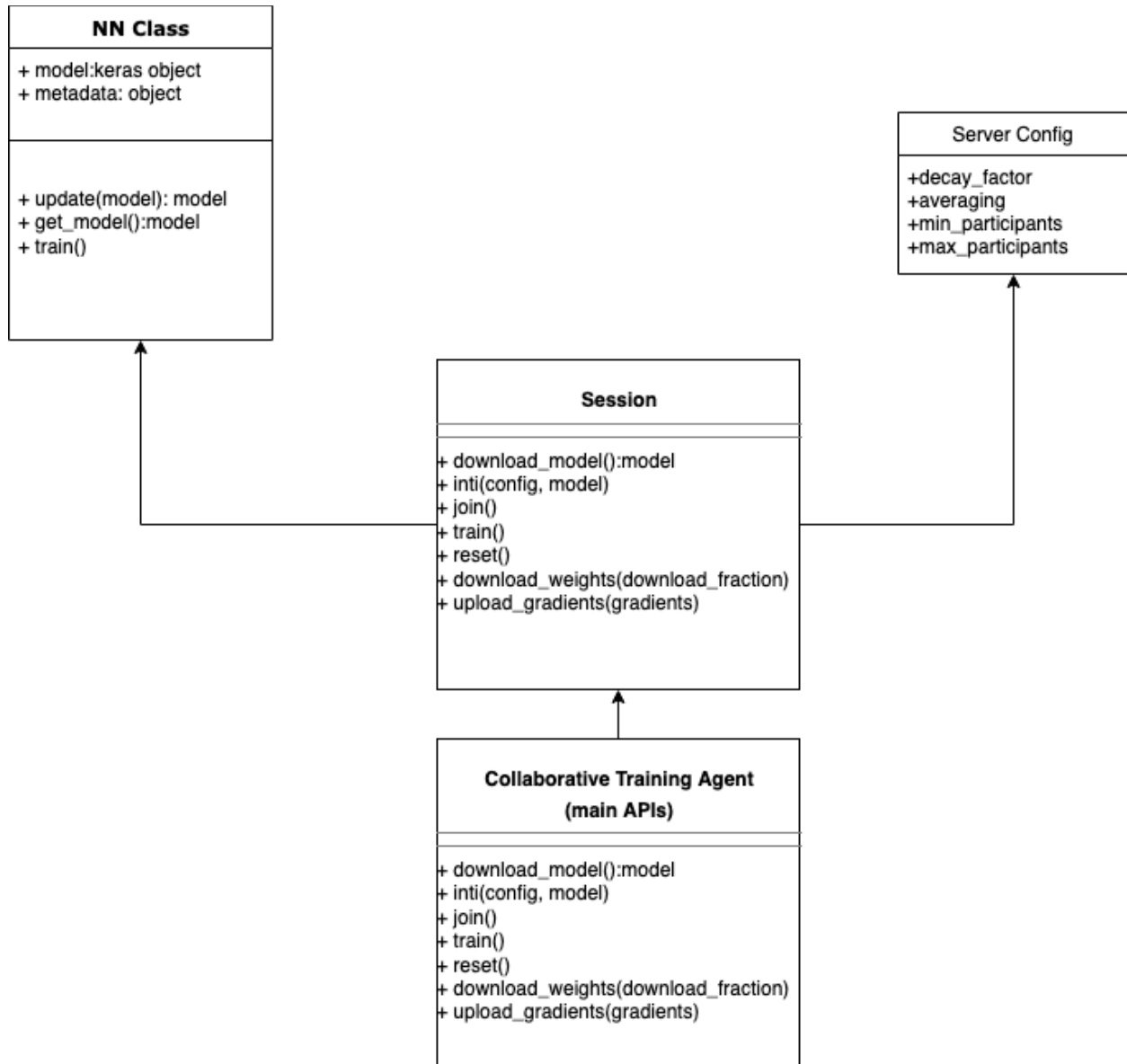
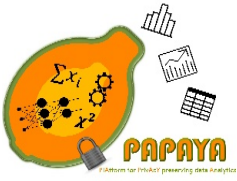


Figure 5: The server components

Figure 5 presents the server-side components that will run on PAPAYA K8s cluster. The service instance will be allocated per training task.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

3.2.2 Behavioral analysis:

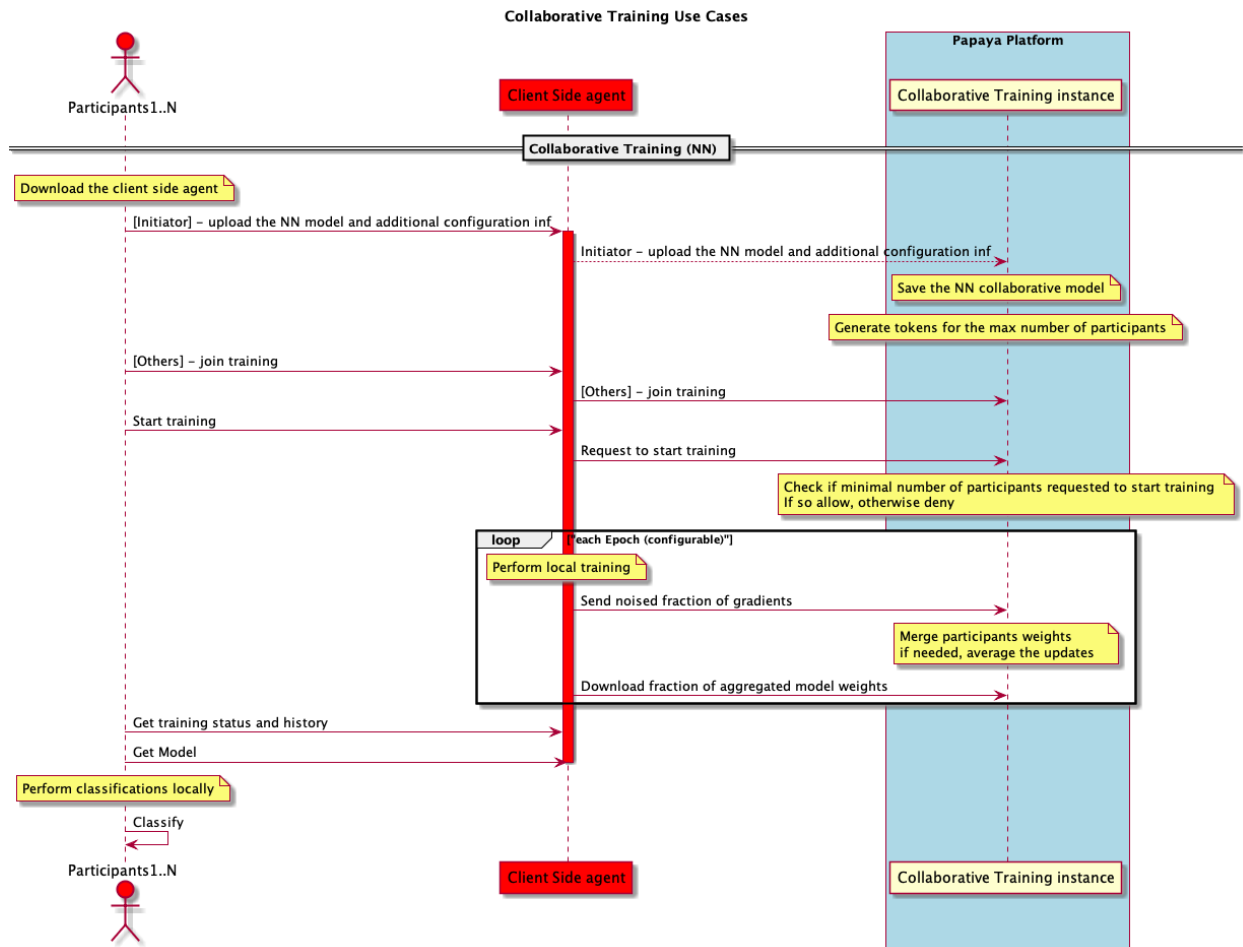
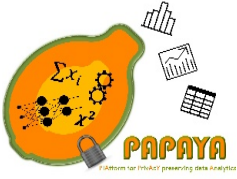


Figure 6: Collaborative training - sequence diagram

As shown in Figure 6, the training phase begins when the required minimal number of participants (as defined in the configuration) requested to join and start the training. Each client will provide a path to the local dataset to the agent. The agent will perform a NN training locally and upload the noised gradients or weights of noised model (depending on the chosen approach and configuration) to the centralized server (located on the PAPAYA cluster). The server will aggregate the updates to the centralized model. The aggregation techniques can be configured. The server allows agents to download the updated model. Client agents download the model and overwrite the local one. The agent proceeds to the training phase on the updated model and so on. The client agent communicates with the server via REST-API calls. The client app should communicate with the client agent via REST-API calls.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.2.3 Deployment and configuration

The necessary information regarding the deployment and configuration of this service is described in D4.1 [1].

3.2.4 Implementation constraints

The service (both the client-side and server-side components) is implemented as a REST-full web service using python² version 3 and Flask framework. Both client and server document the APIs via Swagger³.

The server stores the model in a mounted volume within the K8s server.

As mentioned above, the server side will allocate legal tokens per training task and will expect to receive this token on every request.

The client-side agent expects to receive compiled Keras model.

If the client app chooses to use a differential private local training (based on M. Abadi) approach, then the service supports only the following optimizers:

- SGD
- adam
- adagrad

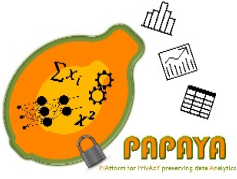
In addition, in this approach, the client app should provide the model's compilation configuration:

- loss
- metrics
- optimizer
- learning rate

These configurations are used in order to recompile the model to privacy preserving model (using tensorflow/privacy library).

² <https://github.com/encryptogroup/ABY> - Release: Oct 19, 2019

³ <https://github.com/encryptogroup/ABY> - Release: Oct 19, 2019



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.2.5 APIs

Server-side APIs:

Servers

server Privacy preserving collaborative training server APIs

GET	/download_model	download the most recent model
GET	/download_weights	download the most frequently updated model's weights
POST	/init	the API to initiate and to join the collaborative training
GET	/join	join collaborative training
POST	/reset	server reset
GET	/train	join to the training process
POST	/upload_gradients	update the centralized model with participant's gradients

Client agent APIs:

Servers

agent Collaborative training client side agent APIs

GET	/get_config	retrieve dp and anomaly detection configuration
GET	/get_model	download a NN model
GET	/get_status	get training status
POST	/init	initialize registration with the server.
GET	/join	join to the collaborative training
POST	/set_config	upload configuration of differential privacy and anomaly detection
POST	/set_model	load NN model and relevant for collaborative training information
POST	/set_token	set authentication token
POST	/train	start a NN training



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.2.6 Service integration evaluation

We executed the complete deployment and training flow and evaluated the entire privacy preserving training flow on the Swell dataset⁴ (Kaggle, based on use case 2). In particular, we (1) uploaded both, the server-side and the client-side containers to the K8s container registry on IBM cloud; (2) deployed the server-side container on IBM cloud using Papaya platform dashboard; (3) deployed client-side agent and an application which simulates client's application on local machine; and finally (4) executed the entire flow, as depicted in Figure 6. The evaluation results with respect to accuracy/privacy are presented in D3.3 [5].

3.3 Clustering

3.3.1 Privacy-preserving clustering based on 2PC

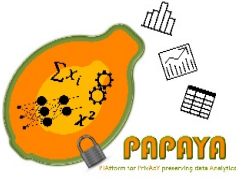
In D4.1 [1], Privacy-preserving trajectory clustering based on 2PC was briefly introduced. There are no further updates for the main components, relationships between main components and the behavioral analysis subsections in this version. This document consists of updates on deployment, configuration, implementation constraints and APIs sections. Service integration evaluation will be presented in D4.3 [6].

3.3.1.1 *Deployment and configuration*

Deployment. As stated D4.1 [1], the privacy-preserving trajectory clustering based on two-party computation (2PC) service and consists of two components, namely the client-side and server-side components. Both components are deployed as Docker containers. Before the deployment, the server component should have the information of the local IP address (which should be 0.0.0.0 by default because of the use of Flask framework as a RESTful service), local HTTP port and the local TCP port (any available port numbers, except the reserved ports by Ubuntu operating system). Note that the local IP address and the TCP port number is required to run ABY server in the server component.

Configuration. This service does not require any specific configuration for the installed, libraries, tools and the operating system that will be used for accomplishing the 2PC-based PP clustering. On the other hand, once the server-side component is deployed to a cloud environment, it will have a new public IP address and a port number. This newly assigned communication information should be also stored in the client-side component to execute 2PC computation. Therefore, the one and only configuration is required for the client-side component, which is realized by using "init" API call.

⁴ <https://github.com/encryptogroup/ABY> - Release: Oct 19, 2019



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

3.3.1.2 Implementation constraints

The privacy-preserving trajectory clustering based on 2PC service uses the latest version of the ABY framework as a 2PC library. For RESTful Web-service, Python v3 and Flask v1.0.0 (or higher versions) are used.

3.3.1.3 APIs

Server-side component APIs:

There is only one RESTful API in the server-side component. This API is used for initiating the ABY based server in the server-side component. All other communications between the client-side component and the server-side component are realized by using ABY sockets.

POST cluster/

POST
/cluster
Cluster - Server-side Component

This API call is used for initiating the server-side component for clustering.

Parameters
Try it out

Name	Description
Request ★ required string (path)	Request for initiating the clustering <div>Request - Request for initiating the clustering</div>

Responses

Code	Description	Links
201	Clustering has been initiated in the server-side component	No links
400	invalid input, object invalid	No links



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Client-side component APIs:

There are two RESTful API calls in the client-side component. The first one is used for initializing the public IP address and the port number of the server (init/) and the second one is used for initiating the clustering in the client side. While initiating the clustering, the to-be clustered line segments obtained from the trajectories are uploaded.

POST init/

POST
/ `init` Init - Client-side Component

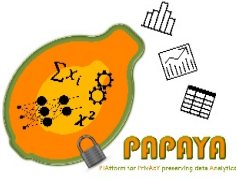
This API call is used for initializing the port number and the IP address for client-side component.

Parameters
Try it out

Name	Description
IP address * required string (path)	IP address of the server <input type="text" value="IP address - IP address of the server"/>
Port number * required string (path)	port number of the server <input type="text" value="Port number - port number of the server"/>

Responses

Code	Description	Links
201	Server's IP address and the port number has been saved	No links
400	invalid input, object invalid	No links



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

POST cluster/

POST
/cluster
Cluster - Client-side Component

This API call is used for initiating the client-side component for clustering.

Parameters
Try it out

Name	Description
Line segments * required string (path)	File that consists of the line segments to be clustered. <div> Choose File No file chosen </div>

Responses

Code	Description	Links
201	Clustering has been initiated in the client-side component	No links
400	invalid input, object invalid	No links

3.3.1.4 Service integration evaluation

Service integration evaluation will be presented in D4.3 [6].

3.3.2 Privacy-preserving clustering based on MinHash

In this section we provide the design of privacy preserving clustering based on 2PC described in D3.3, based on the MinHash clustering algorithm [14] [15]. The solution implemented uses Facebook Crypten open source library.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.3.2.1 Main components and their relationships

In this scenario there are three actors. First, the client that want information about people travels. Second, the Orange Business Service (OBS) that host the service' server. Finally, the Orange Mobile Network (OMN) that provides the requested data to the service.

This solution has two main components:

1. Server-side component
2. Client-side component

Server-side component:

It consists in the following two modules (Figure 7):

- Agent module. This module is used to communicate with the client agent and the antenna. Also, the module supervises the computation by invoking the cryptographic module.
- Crypto module. The crypto module provides the 2PC framework for the computations.

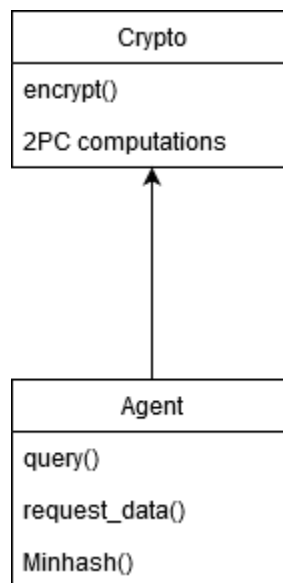


Figure 7 Server-side component

Client-side component:

It consists in the following modules (Figure 8Error! Reference source not found.):

- Agent module. The agent module deals with the communications with the server agent and OMN. It also supervises the computations by invoking the cryptographic module.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

- Crypto module. This module provides the 2PC framework that will be used during the computations of the clustering.

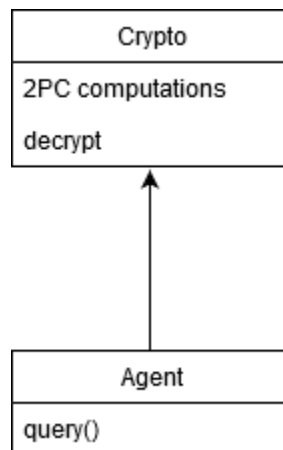
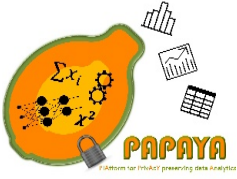


Figure 8 Client-side component

3.3.2.2 Behavioral Analysis.

The service operates as Figure 9 shows. The client starts by sending a clustering query to the server (step 1). The server analyses the query and then make the relevant request to OMN that gather the necessary data (steps 2, 3). The server agent computes the MinHash of the received data (step 4). Then the server agent encrypts the hashes and proceeds with the computations of the clusters with 2PC (steps 6, 7). Once the clusters are computed, a check for k-anonymity is done (steps 8, 9). Finally, the client decrypts and recovers the result (steps 10, 11).



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

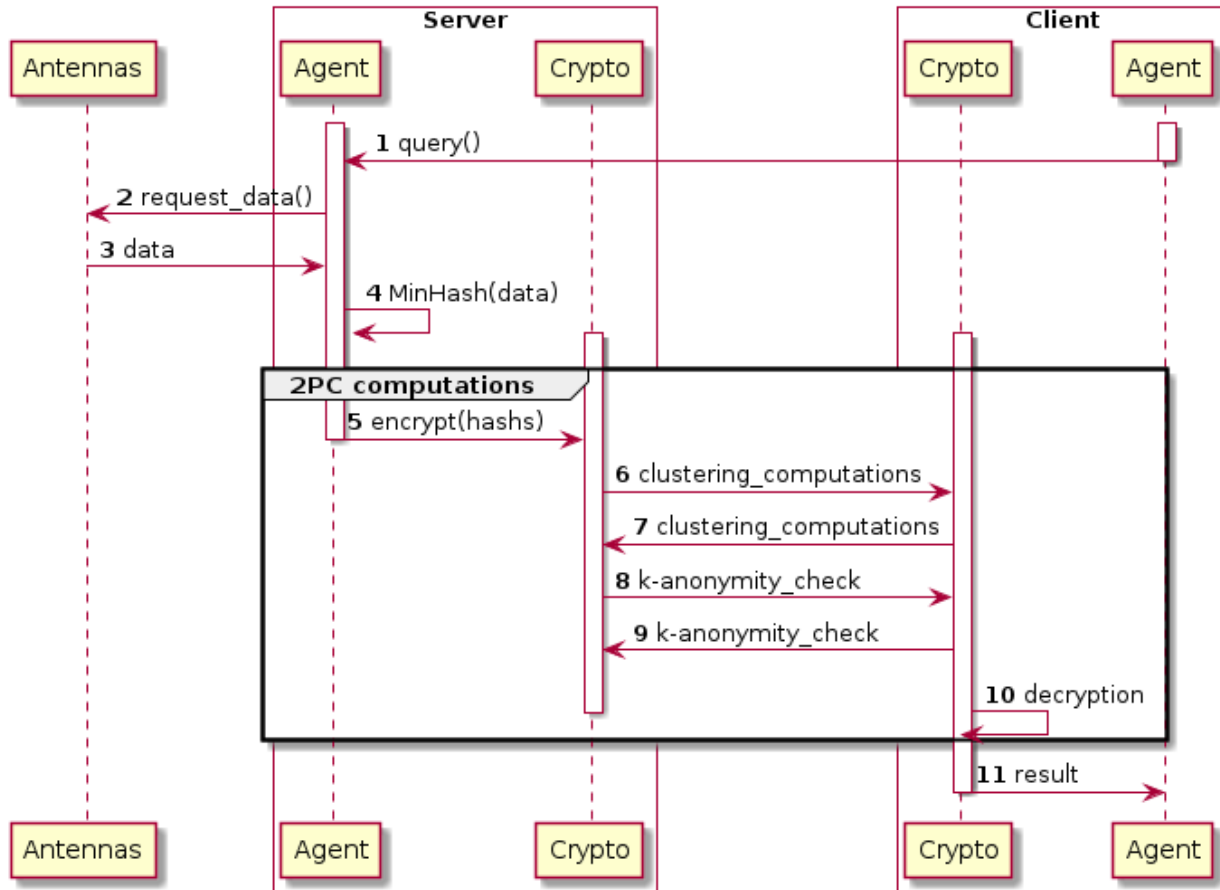


Figure 9 Trajectory clustering analysis

3.3.2.3 Deployment and configuration

The components described above will be deployed within the platform as Docker containers.

Configuration parameters of the server include: the IP addresses and port numbers of the two types of clients. In turn, the client should have the server's IP address and port number. Requestor and client do not interact with each other.

3.3.2.4 Implementation constraints

The agents are implemented in Python programming language and communicate using a REST API. The 2PC computations are done using the Facebook's CrypTen⁵ open source library.

⁵ <https://github.com/facebookresearch/CrypTen>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

3.3.2.5 APIs

Server API:

There are two REST API calls supported by the Server-side component: (1) *query*; and (2) *request_data*. The *query* call receives the query file from the client and processes it. The *request_data* call requests data from the providers. Following is a detailed description of the calls.

POST `/request_data` Request the data to the provider.

Request the data to the provider.

Parameters [Try it out](#)

Name	Description
<code>request</code>	Request for the data to the provider in json file.
<code>()</code>	<div>request - Request for the data to the provider in json file.</div>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

GET `/query` Get the query from the client.

Get the query from the client and process it.

Parameters Try it out

Name	Description
query * required (query)	Query from the client in json file.

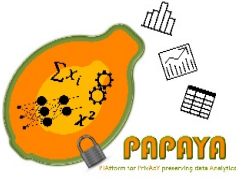
query - Query from the client in json file.

Responses Response content type **application/json**

Code	Description
405	Invalid input

Client API:

There is only one REST API call on the client side. The call is *query*, it is used to generate and send a query file to the server.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

POST /query Send a query for clustering to the server.

Send a query for clustering to the server.

Parameters
Try it out

Name	Description
query * required (query)	Query to the server for clustering. <input type="text" value="query - Query to the server for clustering."/>

Responses
Response content type application/json

Code	Description
405	Invalid input

3.3.2.6 Service integration evaluation

Service integration evaluation will be presented in D4.3 [6].

3.4 Basic Statistics

3.4.1 Privacy-preserving statistics based on Functional Encryption

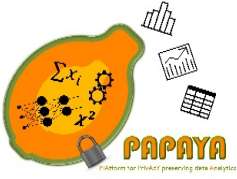
This solution uses functional encryption to allow a requestor to compute statistics on users' mobile usage. Functional encryption ensures data confidentiality and also that only the computation agreed by the user can be done. A detailed description of the design is given in D4.1 [1].

3.4.1.1 Service integration evaluation

Service integration evaluation will be presented in D4.3 [6].

3.4.2 Privacy-preserving Counting using Bloom Filters

In this section, we provide the design of the solution for privacy preserving counting using Bloom filters as described in D3.3 [4]. The implementation of this solution uses Microsoft SEAL for the



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

homomorphic computations and Facebook Crypten for the 2PC computations: both are open source libraries. The main update from D4.1 [1] is the introduction of 2PC for the computation of the k-anonymity.

3.4.2.1 Main components and their relationships

We give a brief overview of the components to understand the update in the next sections. Full details are available in D4.1 [1].

The solution consists of three main components:

1. **Requestor-side component.** This component is composed of two modules:
 - a. Agent module. The purpose of the agent module is to communicate with server component.
 - b. Crypto module. The goal of the crypto module is to provide cryptographic primitives for the requestor.
2. **Server-side component.** This component is composed of three modules:
 - a. Storing module. The purpose of the storing module is to store on the server the requests sent by the requestor.
 - b. Operation module. This module compute bloom filters intersections/unions in the encrypted domain, depending on the request from the requestor.
 - c. Interface. The goal of the interface module is to deal with communications from the requestor and the client.
3. **Client-side component.** This component is composed of two modules:
 - a. Agent module. The agent module interfaces with the server, collects data from antennas and invokes the crypto module to encrypt the data.
 - b. Crypto module. The crypto module provides the cryptographic primitives for the client.

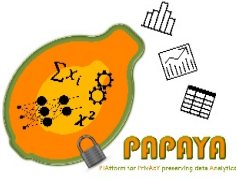
3.4.2.2 Behavioral analysis

We only give an overview of the behavioral analysis to be able to explain the update. The full details of the behavioral analysis are available in D4.1 [1].

The service operates in three main phases (see Figure 10):

1. System initialization, including the analytics initialization and the key generation;
2. Encryption phase - used to encrypt the Bloom Filters;
3. Statistics phase, including the computation on encrypted data and the decryption of the result.

The main update in the new version comes during the Statistics phase. In the previous version, there was no k-anonymity check before the requestor decryption. Now, before sending the result to the requestor for decryption, we apply a mask (Step 11). Then the ciphertext is sent to the



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

requestor (Step 13). The requestor decrypts it (Step 14). After that, the server and the requestor check if the value is greater than the anonymity threshold (Step 15, 16), using 2PC. Finally, the requestor decrypts the final result (Step 17).

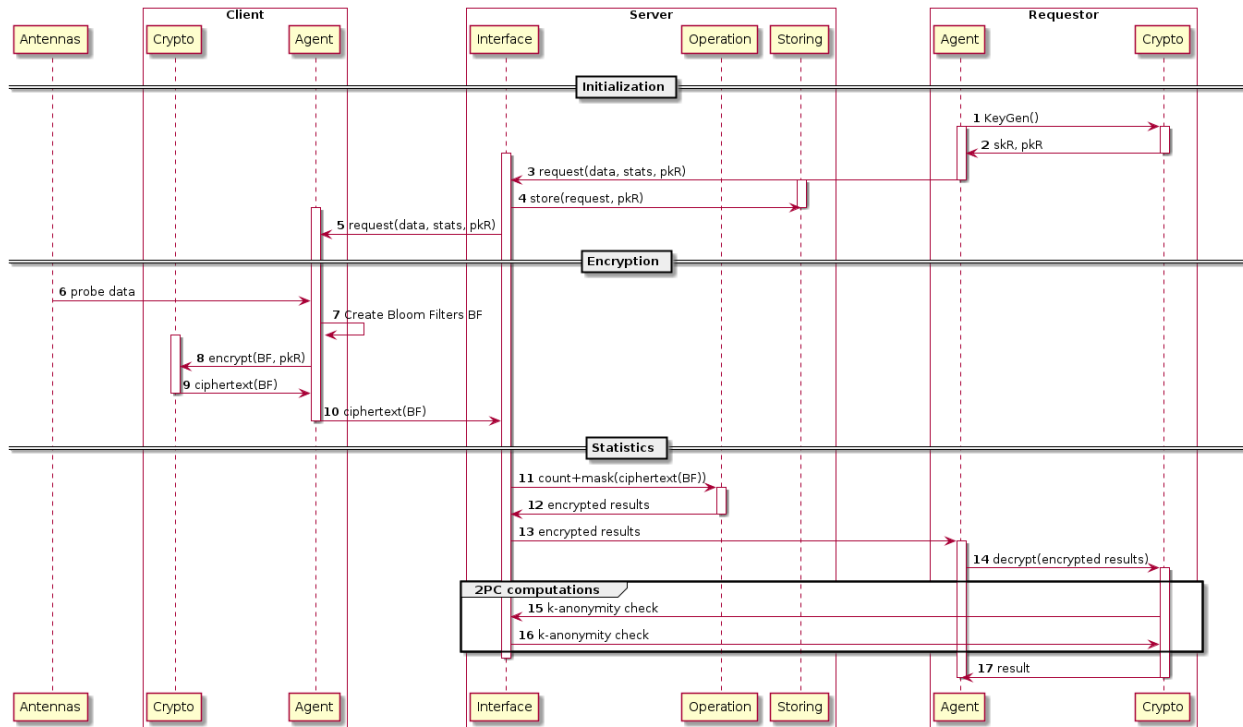


Figure 10 Privacy-preserving statistics with Bloom Filters

3.4.2.3 Deployment and configuration

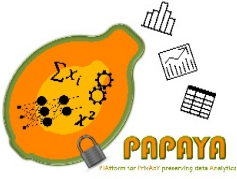
The above-described components will be deployed within the platform as Docker containers.

Configuration parameters of the server include: the IP addresses and port numbers of the two types of clients, and in turn the clients should have the server's IP address and port number. Requestor and client do not interact with each other.

3.4.2.4 Implementation constraints

During the collection of encrypted Bloom filters provided by the client, the server may wait for the reception of several Bloom Filters (at different times) before applying the statistics operation.

The input Bloom filters must have all the same size so that operations such as intersection could be defined.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

3.4.2.5 APIs

The APIs are the same, so we refer to D4.1 [1] for the description of this section.

3.4.2.6 Service integration evaluation

Service integration evaluation will be presented in D4.3 [6].

4 Platform Security and Transparency

4.1 IAM

The Identity Access Manager (IAM) carries out the Authentication and Authorization services to protect the accesses to the PAPAYA platform. In order to do so, a traditional bastion-host pattern has been applied, which implies the implementation and deployment of different components, not only for the IAM server itself. Hence this chapter is devoted to describing the IAM component but also those other complementary components necessary to provide these services. In contrast with the previous version of this deliverable (D4.1), the following sections of this report focus on the deployment details of the different components necessary for providing Authentication and Authorization services within the Papaya platform.

4.1.1 Main components and their relationships

The main IAM components and their relationships with other components of the PAPAYA platform is described in detail within deliverable D4.1. Just as a reminder on what is described in the previous version of this document, as shown in the Figure 11, all access to the PAPAYA framework will be done towards the Security Proxy component, which will contact the IAM to verify if every access is authenticated and authorized and, if the access is granted, it will be redirected to the corresponding Computation Component.

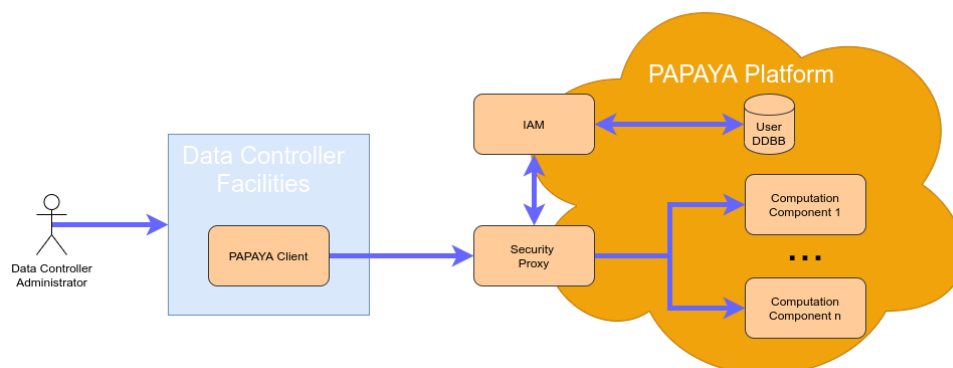


Figure 11 IAM and Security Proxy in the PAPAYA Platform



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

4.1.2 Deployment and configuration

As mentioned before, this section is devoted to describing the deployment of a set of different components, not just the IAM server itself. In order to provide the Authentication and Authorization services for the PAPAYA framework it is necessary to follow each of these steps:

- Step 1. IAM deployment (including the DDBB associated)
- Step 2. OAuth 2 client configuration
- Step 3. Computation Components deployment
- Step 4. Security-Proxy deployment

The Figure 12 shows these steps in place:

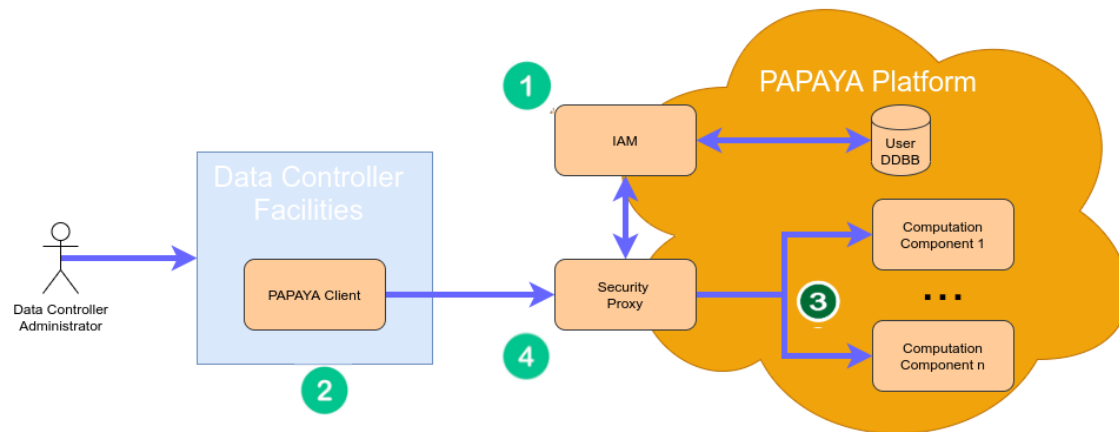


Figure 12 Deployment steps

4.1.2.1 Step 1 IAM deployment

As the IAM selected is based on a broadly used open source solution, the deployment of the IAM server can be performed easily using the following command:

```
$helm install --name keycloak -f keycloak-values.yaml codecentric/keycloak
NAME: calico-macaw
LAST DEPLOYED: Thu Mar24 10:32:36 2020
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/ConfigMap
NAME          DATA AGE
calico-macaw-keycloa-sh  1 0s
calico-macaw-keycloa-startup 1 0s
calico-macaw-keycloa-test  1 0s
```



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

```

==> v1/Pod(related)
NAME          READY STATUS    RESTARTS AGE
calico-macaw-keycloa-0 0/1   ContainerCreating 0    0s

==> v1/Secret
NAME          TYPE  DATA AGE
calico-macaw-keycloa-db  Opaque 1  0s
calico-macaw-keycloa-http  Opaque 1  0s

==> v1/Service
NAME          TYPE    CLUSTER-IP  EXTERNAL-IP  PORT(S)    AGE
calico-macaw-keycloa-headless ClusterIP None      <none>      80/TCP,8443/TCP 0s
calico-macaw-keycloa-http  ClusterIP 10.98.100.117 <none>      80/TCP,8443/TCP 0s

==> v1/StatefulSet
NAME          READY AGE
calico-macaw-keycloa 0/1  0s
NOTES:
Keycloak can be accessed:
* Within your cluster, at the following DNS name at port 80:
calico-macaw-keycloa-http.default.svc.cluster.local
* From outside the cluster, run these commands in the same shell:
export POD_NAME=$(kubectl get pods --namespace default -l app.kubernetes.io/instance=calico-macaw -o
jsonpath='{.items[0].metadata.name}')
echo "Visit http://127.0.0.1:8080 to use Keycloak"
kubectl port-forward --namespace default $POD_NAME 8080

Login with the following credentials:
Username: keycloak

To retrieve the initial user password run:
kubectl get secret --namespace default calico-macaw-keycloa-http -o jsonpath='{.data.password}' | base64 --decode;
echo

```

As result of the execution of the previous command, the whole information or the containers created and how to access to the IAM is displayed.

4.1.2.2 Step 2. Configure the client

The access from the client side to the platform will be done following an OAuth 2 standard as a client. Therefore, it is necessary to configure the access of the client on the IAM platform beforehand. Although in the PAPAYA deployment, this configuration will be automatized using the appropriate scripts and the client's configuration details can be consulted in the following link: https://www.keycloak.org/docs/9.0/server_admin/#_clients



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

4.1.2.3 Step 3. Computation Components Deployment

Once the IAM server is in place and the client has been configured properly it is then the turn of the deployment of all the PAPAYA platform services. Each service deployed can be associated with one or several entry points that need to be exposed to the exterior of the platform. Although how Computation Components are deployed within the platform is not the focus of this section (each Computation Components will have their own deployment plan), the entry points associated with them will be important. This list of entry points will be used in the following configuration steps.

4.1.2.4 Step 4. Security-Proxy deployment

At the last step, using the list of entry points obtained previously, the Security-Proxy can be configured and deployed. In order to do so, it is only necessary to execute the following command:

```
$helm install --name security-proxy -f security-proxy-values.yaml .
```

4.1.3 Implementation constraints

All issues and constraints associated with this set of components are described in the previous version of this document D4.1.

4.1.4 APIs

The API definition of all the services described in this section can be consulted in the following link: <https://www.keycloak.org/docs-api/9.0/rest-api/index.html>

4.1.5 Service integration evaluation

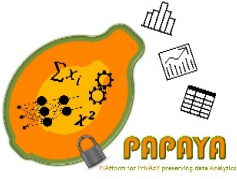
The deployment of the whole set of components for providing the Authentication and Authorization services has been deployed successfully locally but also within the cloud environment provided for the deployment of the project. Service integration evaluation will be presented in D4.3 [6].

4.2 Auditing

Towards being able to hold stakeholders accountable for their use of the PAPAYA platform and services, data processing is logged both as part of the platform and locally at agents.

4.2.1 Platform auditing

Our approach to platform auditing is described in Section 5.2.1 of D4.1 [1]. Here, we first provide an overview and then describe how to evaluate the integration of the platform auditing into the PAPAYA platform.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Project No. 786767

4.2.1.1 Overview

We plan to deploy platform auditing in two stages:

1. In the first stage (due M30, as part of the intermediate version of the platform milestone), we deploy the commonly used Elastic stack in the platform's K8s cluster.
2. In the second stage due M36, we improve the security of the setup by making the resulting logs (i) tamper proof and (ii) verifiable—in terms of authenticity and time—by third parties.

As part of the Elastic stack, we will in the first stage use the Filebeat⁶, Elasticsearch⁷, and Kibana⁸ components. Containers that run the analytics services (Section 3) log their operations to standard output, and Filebeat is configured in K8s (as part of pods) to collect all that output and send it to the Elasticsearch instances we run as part of the platform for storage. From the platform dashboard, administrators and platform users will be able to access an instance of Kibana to view their logs stored in Elasticsearch. One significant open question is how to deal with access control to logs in Kibana⁹. Our goal is to have this operational as part of the intermediate version of the platform in M30.

The second stage will involve one additional component of the Elastic stack: Logstash¹⁰. Logstash is a log pre-processor that receives logs from Filebeat, performs some processing, and then forwards them to Elasticsearch. We will create a custom Logstash component based on an existing secure logging scheme built for the Elastic stack [16]. The component will efficiently sign the logs such that they can be shared and verified by third parties. The figure below summarizes the flow of logging data for platform auditing and the involved components.

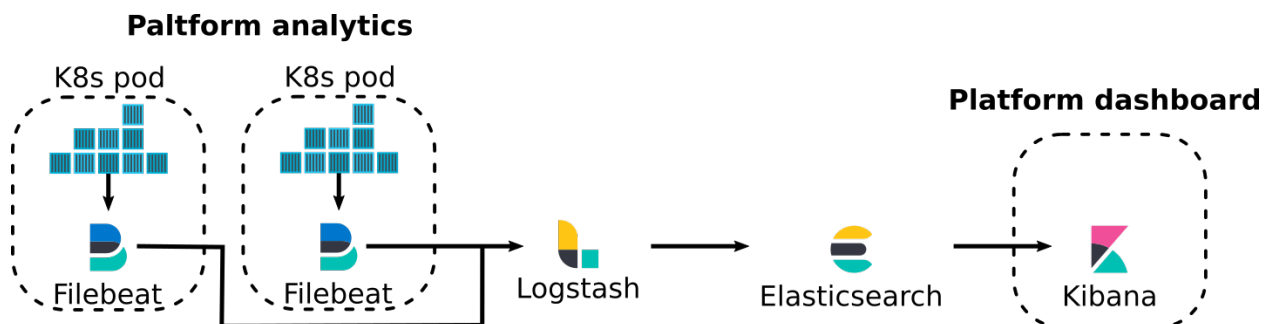


Figure 13: Platform auditing components and the flow of logs.

⁶ <https://www.elastic.co/beats/filebeat>

⁷ <https://www.elastic.co/elasticsearch/>

⁸ <https://www.elastic.co/kibana>

⁹ This used to be a paid feature of the Elastic stack but rudimentary functionality is now part of the open source components, see <https://www.elastic.co/blog/security-for-elasticsearch-is-now-free>. Before M30, we need to decide on using the limited free functionality of paying for the full functionality.

¹⁰ <https://www.elastic.co/logstash>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

4.2.1.2 Integration evaluation

We will validate the integration of platform auditing as follows for the first stage:

1. Create a new user for the platform.
2. Create one or more instances of analytic services.
3. For at least one service, run its agent and perform some analytics.
4. As the newly created user, access Kibana through the platform dashboard. Verify that all expected logs are available.
5. Perform the same verification as in 4, but for another user, verifying that it cannot see the newly generated logs.
6. Perform the same verification as in 4, but for the platform administrator, ensuring that it can see all logs.

For the second stage, there are two additional steps:

7. Repeat steps 4 and 6, but verify the completeness of the logs: do the logged events capture all relevant information for the logs to be useful for auditing and ultimately accountability purposes?
8. For a subset of logs in steps 4 and 6, extract the authenticity data and verify it.

4.2.2 Agent auditing

Our approach to platform auditing is described in Section 5.2.2 of D4.1 [1]. Here, we first provide an overview and then describe how to evaluate the integration of the agent auditing into the PAPAYA platform.

4.2.2.1 Overview

Agents log their processing to standard out, allowing the container environment (such as Docker) to capture them. Quickly viewing agent logs generated by the container running an analytics agent is possible through the Agent Dashboard, see Section 5.2. The use the Agent Dashboard is mainly intended during the development phase of using the PAPAYA platform. During the operational phase, agent logs can use the same Elastic stack as the platform auditing.

4.2.2.2 Integration evaluation

See the Agent Dashboard in Section 5.2 for the development stage. For the re-use of the Elastic components used for platform auditing, we plan to produce a small proof-of-concept where logs generated by the agent is shown in an instance of Kibana running on localhost. Further evaluation is pointless; most PAPAYA users will already be operating some form of logging infrastructure. By showing that the widely used Elastic stack can be used we demonstrate that the PAPAYA platform can be operationalized as part of a larger existing environment.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

4.3 Key Manager

The Key Manager (KM) component is devoted to providing a collateral service of the use of a privacy-preserving platform, carrying out the management of the cryptographic material during the whole lifecycle of the PAPAYA project in the cases where it will be required. In the previous version of this report, it was described the architecture and a detailed design of the Key Manager. Nevertheless, in the following sections of this document are focus on providing the deployment details necessary to use this service and updating that information that have changed since the last report.

4.3.1 Main components and their relationships

The main components and the integration of them within the PAPAYA platform is described in the section 1.1 of the deliverable D4.1 [1].

4.3.2 Deployment and configuration

In order to facilitate the deployment of the Key Manager server, it is available in a docker image. Therefore, to deploy a KM container within the PAPAYA framework it will only be necessary to execute the following command:

```
$ docker run -p 9311:9311 --name key-manager -t de.icr.io/papaya-de/key-manager
```

In addition, the KM client developed to facilitate the integration with other components is available for different programming languages and frameworks, such as: C Sharp, Java, JavaScript, Python, etc.

4.3.3 Implementation constraints

Section 1.1.18 of deliverable D4.1 [1] describes in detail the implementation constraints associated with the implementation of this component.

4.3.4 APIs

Although in the previous version of this report it was detailed the API interfaces used for the Key Manager, some small changes on the interface has been done since them. Hence, the final REST API that will allow to store and to retrieve the cryptographic material is as follows:



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

POST/secrets

POST

/secrets Creates a Secret entity

Creates a Secret entity. If the payload attribute is not included in the request, then only the metadata for the secret is created, and a subsequent PUT request is required

Parameters

[Try it out](#)

Name

Description

X-Project-ID * required

X-Project-ID

string
(header)

body * required

object
(body)

Secret object that needs to be added

Example Value | Model

```
{
  "name": "string",
  "expiration": "string",
  "algorithm": "string",
  "bit_length": 0,
  "mode": "string",
  "payload": "string",
  "payload_content_type": "application/octet-stream",
  "payload_content_encoding": "string",
  "secret_type": "application/octet-stream"
}
```

Parameter content type

application/json





Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

Dissemination Level – PU

Responses		Response content type
		application/json
Code	Description	
201	Successfully created a Secret	
	Example Value Model	
	<pre>{ "secret_ref": "string" }</pre>	
400	Bad Request	
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource	
403	Forbidden. The user has been authenticated, but is not authorized to create a secret. This can be based on the user's role or the project's quota	
415	Unsupported media-type	

GET/secrets

GET	/secrets	Lists a project's secrets
Lists a project's secrets. The list of secrets can be filtered by the parameters passed in via the URL. The actual secret payload data will not be listed here. Clients must instead make a separate call to retrieve the secret payload data for each individual secret		



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Parameters
Try it out

Name	Description
X-Project-ID * required string (header)	<input type="text" value="X-Project-ID"/>
offset integer (query)	The starting index within the total list of the secrets that you would like to retrieve <input type="text" value="offset - The starting index within the total list of the secrets that you would like to retrieve"/>
limit integer (query)	The maximum number of records to return (up to 100). The default limit is 10 <input type="text" value="limit - The maximum number of records to return (up to 100). The default limit is 10"/>
name string (query)	Selects all secrets with name similar to this value <input type="text" value="name - Selects all secrets with name similar to this value"/>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

alg string (query)	Selects all secrets with algorithm similar to this value
<input type="text" value="alg - Selects all secrets with algorithm similar"/>	
mode string (query)	Selects all secrets with mode similar to this value
<input type="text" value="mode - Selects all secrets with mode similar 1"/>	
bits integer (query)	Selects all secrets with bit_length equal to this value
<input type="text" value="bits - Selects all secrets with bit_length equal"/>	
secret_type string (query)	Selects all secrets with secret_type equal to this value
<input type="text" value="secret_type - Selects all secrets with secret_1"/>	
acl_only boolean (query)	Selects all secrets with an ACL that contains the user. Project scope is ignored
<input type="text" value="--"/>	



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

created

string
(query)

Date filter to select all secrets with created matching the specified criteria. The values for this parameter are comma separated lists of time stamps in ISO 8601 format. The time stamps can be prefixed with any of these comparison operators: gt: (greater-than), gte: (greater-than-or-equal), lt: (less-than), lte: (less-than-or-equal).

created - Date filter to select all secrets with c

updated

string
(query)

Date filter to select all secrets with created matching the specified criteria. The values for this parameter are comma separated lists of time stamps in ISO 8601 format. The time stamps can be prefixed with any of these comparison operators: gt: (greater-than), gte: (greater-than-or-equal), lt: (less-than), lte: (less-than-or-equal).

updated - Date filter to select all secrets with

expiration

string
(query)

Date filter to select all secrets with expiration matching the specified criteria. The values for this parameter are comma separated lists of time stamps in ISO 8601 format. The time stamps can be prefixed with any of these comparison operators: gt: (greater-than), gte: (greater-than-or-equal), lt: (less-than), lte: (less-than-or-equal).

expiration - Date filter to select all secrets wit

sort

string
(query)

Determines the sorted order of the returned list. The value of the sort parameter is a comma-separated list of sort keys. Supported sort keys include created, expiration, mode, name, secret_type, status, and updated. Each sort key may also include a direction. Supported directions are :asc for ascending and :desc for descending. The service will use :asc for every key that does not include a direction.

sort - Determines the sorted order of the retu



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Responses		Response content type application/json
Code	Description	
200	<p>Successfull request</p> <p>Example Value Model</p> <pre>{ "secrets": [{ "name": "string", "expiration": "string", "algorithm": "string", "bit_length": 0, "mode": "string", "payload": "string", "payload_content_type": "application/octet-stream", "payload_content_encoding": "string", "secret_type": "application/octet-stream" }], "total": 0, "next": "string", "previous": "string" }</pre>	
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource	

GET/secrets/{uuid}

GET	/secrets/{uuid} Retrieves a secret's metadata.
Retrieves a secret's metadata.	



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Parameters

Try it out

Name	Description
X-Project-ID * required string (header)	<input type="text" value="X-Project-ID"/>
uuid * required string (path)	<div>The uuid that needs to be fetched. <input type="text" value="uuid - The uuid that needs to be fetched."/></div>



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Responses		Response content type application/json
Code	Description	
200	Successful request	
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource	
	Example Value Model	
		<pre>{ "status": "string", "created": "string", "updated": "string", "expiration": "string", "algorithm": "string", "bit_length": 0, "mode": "string", "name": "string", "secret_ref": "string", "secret_type": "application/octet-stream", "content_types": "string" }</pre>
404	Not Found	
406	Not Acceptable	

PUT/secrets/{uuid}

PUT

/secrets/{uuid} Upload payload to a secret

Add the payload to an existing metadata-only secret, such as one made by sending a POST /v1/secrets request that does not include the payload attribute..



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

Parameters
Try it out

Name	Description
X-Project-ID * required string (header)	<input type="text" value="X-Project-ID"/>
uuid * required string (path)	secret that need to be added the payload <input type="text" value="uuid - secret that need to be added the paylo"/>
body * required object (body)	Updated secret object <div> Example Value Model </div> <pre>{ "name": "string", "expiration": "string", "algorithm": "string", "bit_length": 0, "mode": "string", "payload": "string", "payload_content_type": "application/octet-stream", "payload_content_encoding": "string", "secret_type": "application/octet-stream" }</pre> <div> Parameter content type <input type="text" value="text/plain"/> </div>

Responses
Response content type

Code	Description
204	Successful request
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource
404	Not Found



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

DELETE/secrets/{uuid}

DELETE /secrets/{uuid} Delete a secret by uuid

Delete a secret by uuid.

Parameters

Try it out

Name	Description
X-Project-ID * required string (header)	X-Project-ID
uuid * required string (path)	The uuid of the secret that needs to be deleted

uuid - The uuid of the secret that needs to be

Responses

Response content type application/json

Code	Description
204	Successful request
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource
404	Not Found



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

GET/secrets{uuid}/payload

GET /secrets/{uuid}/payload Retrieve a secret's payload.

Retrieve a secret's payload.

Parameters

Try it out

Name	Description
X-Project-ID * required string (header)	<input type="text" value="X-Project-ID"/>
uuid * required string (path)	<div>The uuid that needs to be fetched. <input type="text" value="uuid - The uuid that needs to be fetched."/></div>

Responses

Response content type

Code	Description
200	successful request
401	Invalid X-Auth-Token or the token doesn't have permissions to this resource
404	Not Found
406	Not Acceptable



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

4.3.5 Integration evaluation

This set of components has been designed and implemented to give support to other components and then help them to be integrated within the PAPAYA platform. Therefore, the Key Manager Service will be integrated in some of the use cases whenever necessary. Hence the evaluation of this component will be included with the specific evaluation plans where this set of components will be integrated.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

5 PAPAYA Dashboards

There are two dashboards in PAPAYA, namely the platform dashboard and the agent dashboard. The platform and agent dashboards are, as the name suggests, tied to the respective architectural components and their respective back-ends provide for the dashboards. The dashboards are accessed through web views in a web browser by their respective target users.

5.1 Platform Dashboard

Platform dashboard implemented as a Web application hosted in a container that runs on the PAPAYA's K8s cluster. The major differences, with respect to D4.1, in this section are the implementation constraints (due to internal security constraints, since the platform are deployed on the internal IBM's account). The dashboard will provide the following functionality:

1. Present a list of services provided by the platform (Service's Catalog)
2. Add/Edit/Delete services.
3. Create/Deploy/Delete application (a dedicated instance of each of the provided services).
4. Allow application's owners to monitor the flow of the application by presenting operational logs.

5.1.1 Main components and their relationships

As described in the previous deliverable (D4.1 [1]).

5.1.2 Deployment and configuration

As described in the previous deliverable (D4.1 [1]).

5.1.3 Implementation constraints

The dashboard assumes that the user that sends a request is already authenticated (by Identity and Access Management (IAM) mechanism described in Section 4.1).

The platform dashboard supports deployment of three kinds of services:

- Rest server – service which communicates via https only
- Rest and TCP server – service which communicates via https and socket
- TCP server – service which communicates via socket only

Due to internal IBM's security constraints the platform dashboard is limited to 50 TCP channels.

5.1.4 APIs

As described in the previous deliverable D4.1.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

5.1.5 Integration evaluation

We deployed the platform dashboard on the PAPAYA's K8s server. Then we registered a few services and deployed them using the dashboard. In particular, we deployed the service for Privacy-preserving NN classification based on hybrid approach described in Section 3.1.4, and the service for Collaborative Training of Neural Network described in the Section 3.2 (both, the server-side and the client-side), and run the entire flow. In this way we evaluated the functionalities 1-3 listed in the Section 5.1. The evaluation of functionality number 4 (related to presenting operational logs to the application owners) will be presented in D4.3 [6].

5.2 Agent Dashboard

The agent dashboard is briefly described in Section 6.2 of D4.1 [1]. Here, we first provide an overview and then describe how to evaluate the integration of the agent dashboard.

5.2.1 Overview

The agent dashboard is primarily intended to be used during the development phase of a platform client integrating the use of an analytics agent into its systems. There may be some limited use also operationally. The agent dashboard consists of a minimal backend (command line executed) intended to be run on the same system as the agent container and a web-based front-end. Once run, the dashboard opens up a browser interface on localhost that shows the configuration of the run agent and its logs (re-using the visualizations from the Data Subject Toolbox, see Section 6).

5.2.2 Integration evaluation

Do the following steps:

1. Download an agent from the PAPAYA platform.
2. Start the agent in a Docker container.
3. Start the agent dashboard, providing the name of the Docker container from step 2.
4. Open the address printed in the terminal in your browser.
5. View the configuration of the agent in the browser.
6. Perform some analytics with the agent.
7. View the logs of the agent in the browser.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

6 Data Subject Toolbox

Data protection by design is a key consideration of any data controller and data processor since the GDPR came into effect. The use of complex cryptographic systems—such as privacy preserving data analytics—pose a challenge for data controllers and processors when dealing with data subjects. In particular, when it comes to the rights to data subjects to be informed about how their personal data is processed and to remain in control of this processing. Towards addressing these challenges, the PAPAYA framework includes a data subject toolbox with a number of small, independent tools that can be used by data controllers and data processors in their interaction with data subjects.

6.1 Explaining Privacy-preserving Analytics

The components for explaining privacy-preserving analytics were described in Section 7.1 of D4.1 [1] as well as in D3.2 [17]. Here, we first provide an overview and then describe how to evaluate the integration of the components.

6.1.1 Overview

The components can be split in two ways. First, there are independent components for explaining risks to data subjects. These components are generic (in the sense that they are applicable for any privacy-preserving analytic) and tied to the output of the CNIL PIA tool¹¹, which we have also improved¹². These components and the modifications are explained in detail in D3.4 [18]. The other type of component aims to explain in easy terms and with usable visualizations how a specific privacy-preserving analytic works. These components are tailored to only one or a particular group of privacy-preserving analytics (e.g., those based on partial or fully homomorphic encryption). These are also explained in detail in D3.4 [18], with a focus on the design of their user interfaces.

Each component mentioned above are independent of each other, enabling a data controller or processor to pick the relevant components for their use of the PAPAYA platform. The components are implemented (some done, others scheduled for M24-M30) as React Native¹³ components. Such components can easily be integrated into existing iOS and Android apps. This should allow the data controller or data processor, as part of a mobile app they provide to data subjects (~users) of their services, to easily integrate the relevant components to explain how the data processing using privacy-preserving analytics works and any associated risks to data subjects.

6.1.2 Integration evaluation

For the components that explain risk:

¹¹ <https://www.cnil.fr/en/privacy-impact-assessment-pia>

¹² <https://github.com/papaya-h2020/pia> , modifications released as open source

¹³ <https://reactnative.dev/>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

1. Using the modified CNIL PIA tool, perform a PIA up to and including the risk assessment with respect to the six risk categories in the modified tool.
2. Export a JSON-file of the PIA from within the CNIL PIA tool.
3. Use the JSON as input to one of the risk components in React Native. Verify that the communicated risk assessments are the same as specified in the PIA from step 1.

The components that explain a privacy-preserving analytic are not integrated with any other component, but rather integrated into an existing mobile app.

6.2 Data Disclosure Visualization Tool

The Data Disclosure Visualization Tool is briefly described in Section 7.2 of D4.1 [1]. Here, we first provide an overview and then describe how to evaluate the integration of the tool in an existing mobile app.

6.2.1 Overview

The goal of the tool is to provide a quick visualization for data subjects of what personal data (by attribute, such as first name, email, and blood type) has been disclosed concerning the data subject to which entities (data processors or data controllers). The design of the visualization is based on the Data Track¹⁴, an open source tool from the EU FP7 project A4Cloud. This visualization should be appropriate for use in a mobile app and will be a single React Native component. The component will be configured by a JSON file that describes the disclosed data attributes and their recipients.

6.2.2 Integration evaluation

There is no integration with other components from PAPAYA. The integration should be evaluated on the ease of integration with an existing mobile app from a developer's perspective.

6.3 Annotated Log View Tool

The Annotated Log View Tool is briefly described in Section 7.3 of D4.1 [1]. Here, we first provide an overview and then describe how to evaluate the integration of the tool.

6.3.1 Overview

The goal of the tool is to make data processing, as it relates to privacy-preserving data analytics, more transparent towards data subjects. The tool consists of two components: one user-interface component and a log annotator component. The log annotator component is used by the data processor or data controller using a platform agent to create annotated logs of its data processing using the agent. The annotated logs are then provided to the user-interface component that visualizes the annotated log. The visualization is based on the “timeline” view of the Data Track,

¹⁴ <https://github.com/pylls/datatrack>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

but ported to a mobile interface. How the annotated log is stored and provided to the UI component is unspecified, it depends on how the Annotated Log View Tool is used. The implementation of the UI component will be as a single React Native component, and the log annotator component will be provided as a function in several programming languages, including Java and Kotlin for mobile development targeting Android.

6.3.2 Integration evaluation

Verify the integration as follows:

1. Setup an analytics agent.
2. In the code that instructs the agent to perform some analytics, add a corresponding call to the log annotator component to produce a log entry. Store all log entries in a data structure that maps data subjects to relevant log entries.

In an existing mobile app, integrate the UI component. When the UI component is triggered, supply it with the relevant log entries for the data subject stored in step 2.

6.4 Privacy Engine

6.4.1 Main components and their relationships

The deliverable D3.2 [17] details the design and architecture description of the PE and its sub-components. No significant changes have occurred on the design or architecture of this components since then. In addition, in the previous version of this document, the relationships between the Privacy Engine components with the rest of the PAPAYA framework components and a full detailed description of the interfaces used by the PE was detailed. Nevertheless, the following sections of this document are focused on the description of the deployment and configuration of the different set of components of the PE.

6.4.2 Deployment and configuration

This section details how to deploy the different sub-components of the Privacy Engine. Following a classical approach these sub-components can be classified into back-end and front-end component classification. In addition and just as a reminder, the functionality of the Privacy Engine can be also divided into: Privacy Preferences Manager (PPM) and Data Subject Rights Manager (DSRM). So, the following sections describe how to deploy the Privacy Engine components classify by: Back-End Servers and Front-End Interfaces.

6.4.2.1 Back-End Servers

The back-end components have been implemented to be containerized and therefore, easily deployed using dockers. As mentioned before, two different images have been developed, one for the PPM functionality and other for the DSRM. For the PPM deployment it will be necessary to execute the following command:

```
$ docker run -p 8080:8080 de.icr.io/papaya-de/privacy_engine-ppm-server:latest
```



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

While for deploying a DSRM container, it will be as follows:

```
$ docker run -p 8080:8080 de.icr.io/papaya-de/privacy_engine-dsrm-server:latest
```

6.4.2.2 Front-End Interfaces

The front-end interfaces have been developed to be included in mobile applications. In order to maximize the possible platforms where these interfaces can be integrated, they have been developed using the NativeScript framework¹⁵. The use of this popular and broadly used framework facilitates not only the development but also the integration within the final applications used for the PAPAYA pilots. How to integrate these interfaces with the final pilots applications can be consulted in the following link:

<https://www.nativescript.org/faq/how-do-i-add-nativescript-to-an-existing-ios-or-android-app>

6.4.3 Implementation constraints

There are no further implementation constraints since this was reported in the previous version of this document.

6.4.4 APIs

The whole API of the different components of the Privacy Engine can be consulted in the deliverable D3.2 [17].

6.4.5 Integration evaluation

The Privacy Engine will be included in two different Use Cases: *US2 - Privacy-preserving stress management* and *US3 - Privacy-preserving mobility analytics*. Thereby PE evaluation will be included in the plan developed for the evaluation of both Use Cases.

¹⁵ <https://www.nativescript.org/>



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

7 Platform Deployment

There are minor changes to the deployment process (in comparison to D4.1 [1]). In order to make the description self-contained we describe the whole deployment process in this section.

7.1 Platform initialization and platform dashboard deployment

For the proof of concept (PoC) usage, the platform is deployed on IBM's cloud service. The platform administrator (IBM) created an account of K8s service and allocated the required resources for the Kubernetes cluster. In addition, the platform administrator allocated the following required external services:

1. Container Registry (CR). The platform administrator created single namespaces, **papaya-de**.
2. Permissions Provided to the platform users
3. Elasticsearch and Kibana internal instance.

The platform administrator created the following permission groups:

1. Platform administrator
 - a. Allocate resources on the Kubernetes cluster
 - b. Allocate/Create external services
 - c. Admin access to CR
2. Service providers
 - a. Active access to the CR
 - b. If the service requires integration/communication with an external service, appropriate accounts should be created or allocated.
3. Platform Clients
 - a. Should be able to download only the agent side container from the CR

The platform administrator deployed the Platform Dashboard using yaml deployment file. Service providers and platform clients can use it to deploy/use the services.

To support auditing, the platform administrator will execute Loggingbeat as Daemonset on the Kubernetes cluster and connect it to the Elasticsearch instance.

7.2 Service upload and deployment

This section describes the steps needed in order to upload or use a service. For the PoC usage we use a Container Registry Provided by IBM Cloud. The steps are divided into two phases: (1) uploading phase; and (2) usage-execution phase.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

The following steps are common for both phases:

1. Install the IBM Cloud CLI¹⁶
2. Install the Docker CLI¹⁷
3. Install the Container Registry plug-in.
`ibmcloud plugin install container-registry -r Bluemix`
4. Log in to your IBM Cloud account.
`ibmcloud login -a https://cloud.ibm.com`

The uploading phase will be done by the service provider user and it consists of the following steps:

1. Log your local Docker daemon into the IBM Cloud CR `ibmcloud cr login`
2. Create two images: server side and client-side agent. The server-side container will run on papaya's K8s cluster. The client-side agent will run on the client's side. Instructions on how to create the Docker image can be found in the following link:

<https://docs.docker.com/develop/develop-images/baseimages/>

3. Upload each image to the appropriate namespace on the CR for the service images use:

```
docker tag <local container name> de.icr.io/papaya-de/<my_repository>:<my_tag>
docker push de.icr.io/papaya-de/<my_repository>:<my_tag>
```

More details and CR commands can be found in Kubernetes documentation¹⁸.

4. The service provider user (i.e. IBM, ORANGE, EUROCOM) will add a new service to the services catalog on the Platform dashboard (<https://platform.papaya.eu-de.containers.appdomain.cloud>), while specifying all required fields and describing in details the service's flow and requirements.

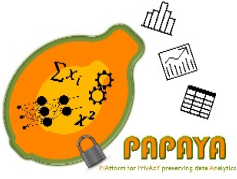
The usage-execution flow will consist of the following steps:

1. The platform client will login to the Platform dashboard (<https://platform.papaya.eu-de.containers.appdomain.cloud/auth/>).
2. The client will select the tab of services catalog and choose the service of interest.
3. The client will provide a configuration file if the service requires so.
4. The client will download the client-side agent image by using the following commands:

¹⁶ <https://cloud.ibm.com/docs/cli?topic=cloud-cli-getting-started#idt-prereq>

¹⁷ <https://docs.docker.com/install/>

¹⁸ <https://cloud.ibm.com/kubernetes/registry/main/start>



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

```
ibmcloud cr login
```

```
docker pull <image name>
```

Where the <image name> is presented in the “services catalog” view on a platform dashboard under the “Agent Side Container” name and will be structured of the following:

```
de.icr.io/papaya-de/<image>:<tag>
```

5. After the services are deployed on the k8s cluster (by the platform dashboard application), the platform will present the url of the deployed service for the https communication or provide a cluster IP and port for TCP (socket) communication, For example:

```
SERVER_URL=https://29dr4d.papaya.eu-de.containers.appdomain.cloud
```

```
SERVER_TCP_PORT=32067
```

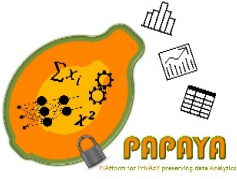
```
SERVER_IP=158.177.182.219
```

In addition, the platform dashboard provides the option to download the file (env file) that will contain these parameters (accordingly to the communication protocol with the service). The client administrator should be able to provide this file to the client agent, based on how the agent expects to receive these parameters. Possible ways of doing that are as follows:

- a. Provide the env file as env. variables for the client agent execution.

```
docker run with an --env-file <file name>
```

- b. Read the env file by the client's application and provide it to the agent side within the initiation step. Meaning that the agent side will provide INIT API and will expect to receive all essential parameters in order to communicate with appropriate server side.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION

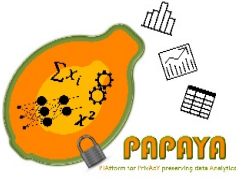
Dissemination Level – PU

8 Conclusions

In this deliverable, we presented modifications and extensions to the platform functional design, architecture and deployment presented in D4.1 [1]. Thus, this deliverable completes the specification presented in D4.1 [1]. We described in detail the core platform services dealing with privacy-preserving computations as well as the services responsible to ensure data privacy, security, and transparency of all the processes while operating the platform. We explained how different services can be integrated into the platform in a way that they will be interoperable/compatible with each other and could work together in the integrated platform. In particular, by using k8s, we show that the analytics are possible to run using modern cloud environments; by adding Identity and Access management (IAM), we ensure that access to the analytics can be properly authenticated and authorized; by adding auditing mechanisms, we ensure that the analytics generate appropriate logging information.

We also presented the design of platform dashboards that will provide UI, configuration, and visualization functionality. Finally, we described how we deployed some of the services and how we evaluated their integration.

In next version of this document (D4.3 [6]) we plan to provide the full description of incomplete services, missing evaluation results, and changes to the current design that maybe be required after the platform evaluation in WP5.



Project No. 786767

D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

9 References

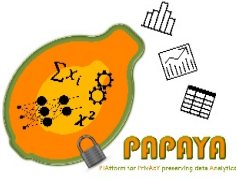
- [1] B. Rozenberg, R. Shmelkin, B. Bozdemir, O. Ermis, M. Önen, M. Barham, M. Azraoui, S. Canard, B. Vialla and T. Pulls, "D4.1-Functional Design and Platform Architecture".
- [2] S. Fischer-Hübner, B. Kane, J. S. Pettersson, T. Pulls, L. Iwaya, L. Fritsch, B. Rozenberg, R. Shmelkin, A. Palomares Perez, N. Ituarte Aranda and J. Carlos, *D2.2 - Requirements Specification*, 2019.
- [3] B. Bozdemir, O. Ermis, M. Önen, M. Barham, M. Azraoui, S. Canard, B. Vialla, B. Rozenberg and R. Shmelkin, *D3.1 - Preliminary Design of Privacy Preserving Data Analytics*, 2019.
- [4] S. Canard, B. Vialla, B. Bozdemir, O. Ermis, M. Önen, M. Barham, B. Rozenberg, R. Shmelkin, I. Adir and R. Masalha, *D3.3 - Complete Specification and Implementation of Privacy preserving Data Analytics*, 2020.
- [5] S. Canard, B. Vialla, B. Bozdemir, O. Ermis, M. Önen, M. Barham, M. Azraoui, B. Rozenberg and R. Shmelkin, *D3.3 - Complete Specification and Implementation of Privacy preserving Data Analytics*, 2020.
- [6] B. Rozenberg, R. Shmelkin, B. Bozdemir, O. Ermis, M. Önen, S. Canard, B. Vialla and T. Pulls, "D4.3 Final report on platform implementation and PETs integration," To be submitted in 2021.
- [7] B. Zvika, G. Craig and V. Vinod, "Fully Homomorphic Encryption without Bootstrapping".
- [8] Z. Brakerski, "Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP," in *Annual Cryptology Conference*, 2012.
- [9] J. Fan and F. Vercauteren, "Somewhat Practical Fully Homomorphic Encryption," *Cryptology ePrint Archive*, vol. Report 2012/144, 2012.
- [10] J. H. Cheon, A. Kim, M. Kim and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International Conference on the Theory and Application of Cryptology and Information Security*, Hong Kong, China, 2017.
- [11] Microsoft Research, Redmond, WA., *Microsoft SEAL*, : <https://github.com/Microsoft/SEAL>, 2018.
- [12] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *53rd Annual Allerton Conference on Communication, Control, and Computing*, Allerton, 2015.
- [13] M. Abadi, A. Chu, I. Goodfellow, B. H. McMahan, I. Mironov, K. Talwar and L. Zhang, "Deep learning with differential privacy," in *the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.
- [14] C. C. a. A. M. F. Tom Bohman, "Min-Wise Independent Linear Permutations," *Electr. J. Comb.*, vol. 7, 2000.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

- [15] Y. Z. a. M. R.-N. Maede Rayatidamavandi, "A Comparison of Hash-Based Methods for Trajectory Clustering," *15th {IEEE} Intl Conf on Dependable, Autonomic and Secure Computing*, pp. 107-112, 2017.
- [16] T. Pulls and D. Rasmus, "Steady: A Simple End-to-End Secure Logging System," <https://eprint.iacr.org/2018/737>, 2018.
- [17] T. Pulls, L. Fritsch, L. Iwaya, F. Karegar, A. Palomares and J. C. Prez Ban, "D3.2 - Risk Management Artefacts for Increased Transparency," PAPAYA report document, 2019.
- [18] S. Fischer-Hübner, M. T. Beckerle, J. S. Pettersson and P. Murmann, "D3.4 - Transparent Privacy preserving Data Analytics," PAPAYA report document, 2020.
- [19] S. Halevi, "HElib," [Online]. Available: <https://github.com/homenc/HElib>.
- [20] "JustGarble," [Online]. Available: <https://github.com/irdan/justGarble>.
- [21] C. r. a. O. S. University, "libOTe," [Online]. Available: <https://github.com/osu-crypto/libOTe>.
- [22] N. Smart and F. Vercauteren, "fully Homomorphic SIMD Operations".
- [23] wiki, "Garbled circuits," [Online]. Available: https://en.wikipedia.org/wiki/Garbled_circuit.
- [24] wiki, "Oblivious transfer," [Online]. Available: https://en.wikipedia.org/wiki/Oblivious_transfer.
- [25] F. Chollet, "Keras," [Online]. Available: <https://keras.io/>.
- [26] M. Abdalla, F. Bourse, A. De Caro and D. Pointcheval, "Simple Functional Encryption Schemes for Inner Products," in *IACR International Workshop on Public Key Cryptography*, 2015.
- [27] M. Abdalla, D. Catalano, D. Fiore, R. Gay and B. Ursu, "Multi-Input Functional Encryption for Inner Products: Function-Hiding Realizations and Constructions Without Pairings," in *Annual International Cryptology Conference*, 2018.
- [28] S. Agrawal, B. Libert and D. Stehlé, "Fully Secure Functional Encryption for Inner Products, from Standard Assumptions," in *Annual International Cryptology Conference*, 2016.
- [29] J. Chotard, E. D. Sans, R. Gay, D. H. Phan and D. Pointcheval, "Decentralized Multi-Client Functional Encryption for Inner Product," in *International Conference on the Theory and Application of Cryptology and Information Security*, 2018.
- [30] E. D. Sans, R. Gay and D. Pointcheval, "Reading in the Dark: Classifying Encrypted Digits with Functional Encryption," IACR Cryptology ePrint Archive, 2018.
- [31] C. E. Z. Baltico, D. Catalano, D. Fiore and R. Gay, "Practical Functional Encryption for Quadratic Functions with Applications to Predicate Encryption," in *Annual International Cryptology Conference*, 2017.



D4.2 – PROGRESS REPORT ON PLATFORM IMPLEMENTATION AND PETS INTEGRATION Dissemination Level – PU

Project No. 786767

- [32] J.-G. L. a. J. H. a. K.-Y. Whang, "Trajectory Clustering: A Partition-and-Group Framework," in *SIGMOD '07 Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, Beijing, China , 2007.
- [33] B. Rozenberg, R. Shmelkin, B. Bozdemir, O. Ermiş, M. Önen, M. Barham, M. Azraoui, S. Canard, B. Vialla and T. Pulls, *D4.2 - Progress report on platform implementation and PETS integration*, 2020.
- [34] Y. Ishai, J. Kilian, K. Nissim and E. Petrank., "Extending Oblivious Transfers Efficiently," in *CRYPTO*, 2003.
- [35] chiraag, "Gazelle MPC," [Online]. Available: https://github.com/chiraag/gazelle_mpc/tree/master/src/lib.
- [36] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics), Springer, 2006.
- [37] S. G. M. M. A. a. S. C. Eleonora Ciceri, *D2.1: Use Cases and Requirements. PAPAYA Deliverable D2.1*, 2019.