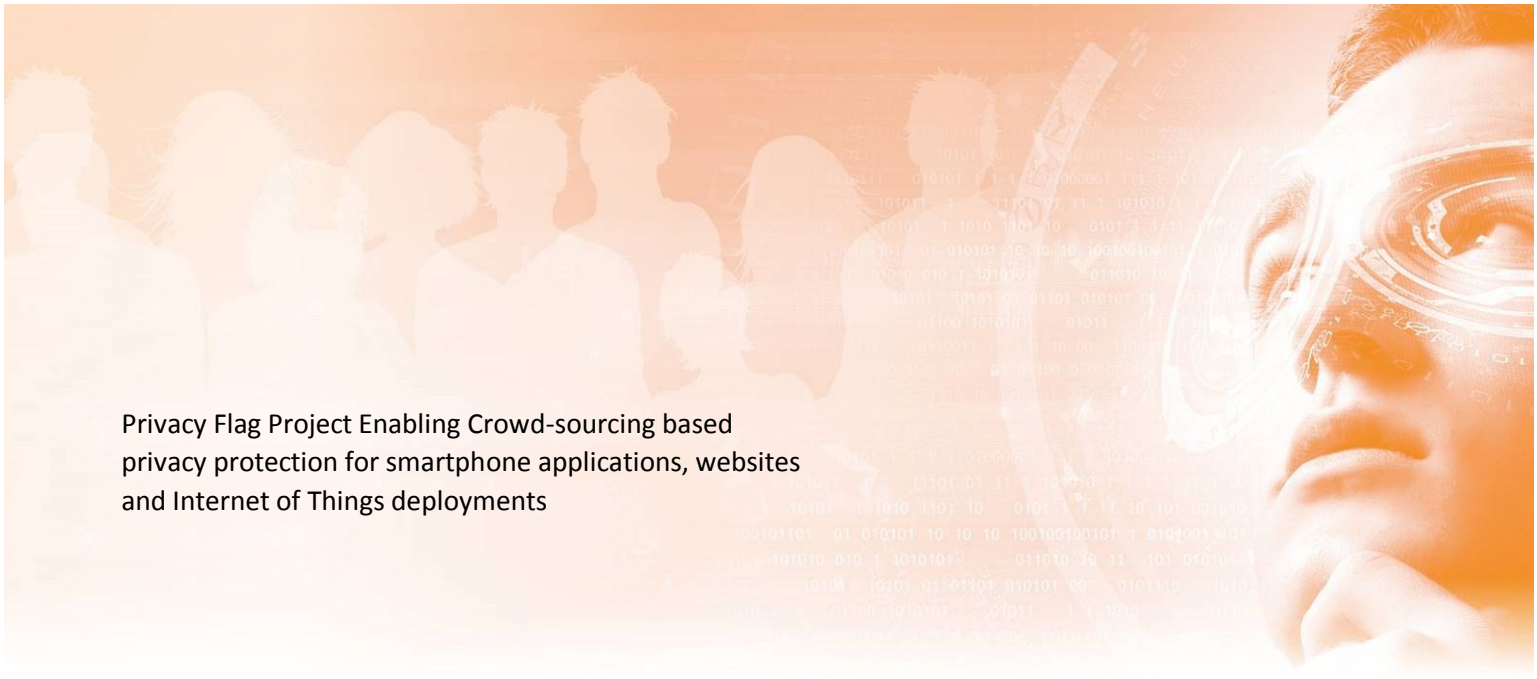


Privacy Flag Project Enabling Crowd-sourcing based privacy protection for smartphone applications, websites and Internet of Things deployments

Grant Agreement No.653426
Topic: DS-01-2014 (Privacy)
Innovation Action

Privacy Flag D5.1
Integration, tests and validation plan

Document Number:	D5.1
Contractual Date of Delivery:	01.05.2016
Editors:	Nenad Gligorić (DNET), Yannis Stamatiou (CTI)
Work-package:	WP5
Distribution / Type:	Public (PU) / Report (R)
Version:	1.0
Total Number of Pages:	50

Co-funded by the European Union
Co-funded by the Swiss Confederation

This deliverable has been written in the context of the Privacy Flag Horizon 2020 European research project, which is supported by the European Commission and the Swiss State Secretariat for Education, Research and Innovation. The opinions expressed and arguments employed do not engage the supporting parties.

Abstract

Privacy Flag combines crowd sourcing, ICT technology and legal expertise to protect citizen privacy when visiting websites, using smart-phone applications, or living in a smart city leveraging user friendly solutions provided as a smart phone application, a web browser add-on and a public website. It will:

1. Develop a highly scalable privacy monitoring and protection solution with: - Crowd sourcing mechanisms to identify, monitor and assess privacy-related risks;

- Privacy monitoring agents to identify suspicious activities and applications;
- Universal Privacy Risk Area Assessment Tool (UPRAAT) and methodology tailored to European norms on personal data protection;
- Personal Data Valuation mechanism;
- Privacy enablers against traffic monitoring and finger printing;
- User friendly interface informing about the privacy risks when using an application or website.

2. Develop a global knowledge database of identified privacy risks, together with online services to support companies and other stakeholders in becoming privacy-friendly, including:

- In-depth privacy risk analytical tool and services;
- Voluntary legally binding mechanism for companies located outside of Europe to align with and abide to European standards in terms of personal data protection;
- Services for companies interested in being privacy friendly;
- Labelling and certification process.

3. Collaborate with standardization bodies and actively disseminate towards the public and specialized communities, such as ICT lawyers, policy makers and academics. Eleven (-11-) European partners, including SMEs and a large telco operator (OTE), bring their complementary technical, legal, societal and business expertise; Privacy-Flag intends to establish strong links with standardization bodies and international fora international fora and it also intends to assess and incorporate outcomes from over 20 related research projects. It will build and ensure a long term sustainability and growth.

Executive Summary

The purpose of this deliverable is to present an initial plan for the testing of the Privacy Flag platform's modules as well as about the integrated platform. These modules are the "output" of WP4 ("Technical Enablers") effort, whose development is still in progress during the composition of this deliverable. The focus of this deliverable is upon WP5 ("*Integration, test and validation*") and, more specifically, upon Task T5.1 ("*Integration and technical validation*").

This deliverable describes, in the best possible detail given the fact that the modules to be tested are not yet fully developed, the envisaged tests that we plan to perform in order to verify that all of the modules work and interact properly, as well as that they have been properly integrated. Thus, the level of details provided for testing will be updated as the modules become available and the integration process starts. For this reason, this deliverable will be followed by a new (updated) version in which these details will be provided along with the test results and corrective measures taken, where relevant.

With respect to the deliverable content, Section 1 discusses the purpose and scope of the deliverable, namely the goal of WP5 as well as that of Task T5.1.

Section 2 provides the initial plan for the scheduling of the testing phase in connection with the other WPs which develop the modules that will be integrated and tested within the full scope of WP5. It also briefly reviews the main available testing methodologies and discusses the partners' choices with respect to selecting the "optimal" methodology for the goals of the entire Privacy Flag (PF) project. Moreover, it describes the use case template that the PF consortium will use in order to describe the use cases for the tests to be performed on the platform modules, before and during the integration phase. The templates contain information that will be populated when the integrated modules are delivered.

Section 3 presents the use cases for testing of the individual modules as well as their integration. The definition of use cases has been based on the requirements of these modules.

Section 4 specifies the small scale pilot planned to be used to perform an overall testing of the integrated platform in real-life conditions.

Finally, in Section 5 we conclude the deliverable and summarize its main points.

Version History

Table 1: Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	17.12.2015	First Draft	Vasileios Vlachos - CTI
0.2	22.12.2015	Second Draft	Yannis Stamatiou - CTI
0.2b	24.01.2016	Website and Backend Management Platform use cases added	Mirjana Nikolic - DNET
0.3	28.01.2016	Third Draft	Nenad Gligoric - DNET
0.3b	07.03.2016	Database and Server Implementation use cases added	Yannis Stamatiou - CTI
0.3c	08.03.2016	Distributed Agents use cases added	Yannis Stamatiou - CTI
0.4	21.03.2016	Fourth Draft	Yannis Stamatiou - CTI
0.4b	24.03.2016	Brower Add-on and Smartphone Application use cases added	Andreas Drakos - Velti
0.5	27.03.2016	Fifth Draft	Yannis Stamatiou - CTI
0.5b	08.04.2016	Security and Privacy Enablers use cases added	Daniel Forster - UL
0.6	12.04.2016	Sixth Draft	Yannis Stamatiou - CTI
0.6b	15.04.2016	Comments from involved partners	All involved partners
0.7	19.04.2016	Seventh Draft - submitted for internal review	Yannis Stamatiou - CTI
0.8	28.04.2016	First revised Draft based on internal reviewers' comments	Nenad Gligoric - DNET
0.9	29.04.2016	Second revised Draft based on internal reviewers' comments	Yannis Stamatiou - CTI
0.10	30.04.2016	Final version of the Deliverable	All involved partners
0.11	09.05.2016	Final Conceptual and Editorial Review	Sebastien Ziegler - MI Ioannis Chochliouros - OTE
1.0	11.05.2016	Final Conceptual and Editorial Review - document ready for submission	Ioannis Chochliouros - OTE

Contributors

Table 2: Contributors

First name	Last name	Partner	E-mail
Nenad	Gligoric	DNET	nenad.gligoric@dunavnet.eu
Mirjana	Nikolic	DNET	mirjana.nikolic@dunavnet.eu
Andreas	Drakos	VELTI	adrakos@velti.com
Daniel	Forster	UL	Daniel.Forster@rwth-aachen.de
Vasileios	Vlachos	CTI	vsvlachos@gmail.com
Yannis	Stamatiou	CTI	stamatiu@ceid.upatras.gr
Sebastien	Ziegler	MI	sziegler@mandint.org
Ioannis	Chochliouros	OTE	ichochliouros@oteresearch.gr

Glossary

ACRONYMS	MEANING
ACID	Atomicity, Consistency, Isolation, Durability
CMS	Content Management System
CRUD	Create, Retrieve, Update, Delete
DA	Distributed Agent
DB	Data Base
DBMS	Data Base Management System
DLV	Deliverable
DoW	Description of Work
EU	European Union
GA	Grant Agreement
GUI	Graphical User Interface
ICT	Information and Communication Technology
ID, id	Identifier
IoT	Internet of Things
IP	Internet Protocol
MS	Milestone
PF	Privacy Flag
QoS	Quality of Service
SQL	Structured Query Language
SSL	Secure Sockets Layer
TBD	To Be Determined
UC	Use Case
UPRAAT	Universal Privacy Risk Area Assessment Tool
WP	Work Package

Table of Contents

List of Figures

List of Tables

1. Introduction

The process of *testing* is the evaluation of a system's components as well as of the whole system taking into account specified functional platform requirements. The goal is to identify and rectify bugs, hardware deficiencies, unexpected behavior and to timely provide corrective measures. We should stress the fact that the testing procedures described in this deliverable will be performed on a technical level, by the development teams of the project themselves and not by real users. Testing with real users, in a pilot operation of the platform, falls within the scope of Task T5.2 and will be elaborated there. However, we plan a small scale pilot operation in the context of Task T5.1, involving members of the CTI team, which we describe in this deliverable.

Testing is, usually, performed on two levels: i) Individual module, and; ii) Integrated system testing. The former refers to the testing of the individual system components with respect to the requirements and the latter refers to the testing of the integrated system itself by testing the interfaces among the modules and the involved data flow and the expected behavior of the system "as a whole".

In order to handle the complexity of the final system, testing will be initially performed on an individual module basis, as detailed in Section 3, before the integration begins. After all modules are verified, then the integration will begin and when it is completed, a final set of tests will be performed to verify the final Privacy Flag platform, mainly in the form of a small scale pilot operation.

Our initial test plan is based on the following general considerations:

- *The adoption of a strategy to use when testing the integrated modules and how the tests will be conducted (i.e., use cases)*. Our consortium has decided on the adopted methodology which is explained in this deliverable.
- *What will be tested (e.g. software modules against set requirements)*. The modules that will be tested are described within the context of WP4.
- *What will be the time frame of the testing (e.g. when it starts and ends)*. We have followed the task and WP sequencing as described in the DoW (WP4 and WP5).
- *Testing responsibilities among involved parties*. These responsibilities have already been assigned and follow the corresponding module design and development responsibilities as described in the DoW, especially in WP4.
- *Pass and fail conditions for each performed test*. The partners have proposed themselves the described tests for the modules they developed and thus, have identified accurately these conditions.
- *Potential risks involved when a test is performed*. Actually, no privacy or other risk is foreseen for the described tests as they will be conducted in a restricted context and controlled environment by the involved partners. Moreover, the small scale pilot will

involve, again, ICT professionals, from CTI. All involved individuals are privacy aware while, *in addition*, no real private data will be stored and exchanged.

- *Approval of the testing plan from all involved parties*. All parties have agreed on the tests and the testing plan as described in this deliverable.

These are some of the most important points to settle for a test plan. However, as the process will progress later in the project duration, more issues may be considered or some already considered issues may be modified, *accordingly*.

The proposed methodology is based on the definition of Use Case Templates that describe use cases to which the individual, as well as the integrated, systems will be subjected. Each use case template includes the information for guiding the testing and the results gathering procedures.

Caveats: We should remark, at this point, that the write-up and delivery of this deliverable occurred *before* the modules described here were fully developed and integrated. In each of the sections that follow, which describe the tests for each integrated module, we have included two subsections titled "Outcomes" and "Corrective measures". As the deliverable has to be submitted by the end of April 2016, when the modules will be unavailable for testing and integration, the deliverable will have many later versions (as a "heartbeat/living" document sequences), in which these two sections will be, appropriately, being filled in with the test results and corrective measures taken as the tests are performed. Then all this information will be processed and presented in a unified manner in Deliverable D5.2.

1.1 Purpose and scope of WP5

WP5's goal is handling the testing and integration of the enablers which will be developed within the other WPs (mostly WP4 – see [6] as well as [5]). WP5 activities will test, interface and integrate the developed building blocks into the final Privacy Flag platform. These blocks include the database and the server on which it runs, the website and backend management platforms, as well as the smartphone application and distributed data agents which collect information, in a distributed fashion, from users' mobile phones.

1.2 Purpose and scope of T5.1

This deliverable is based, mostly, on the work performed within the scope of Task T5.1. This task, according to the DoW, aims at the integration of all the modules and subsystems of which the platform is comprised (mostly implemented within the scope of WP4).

These modules and subsystems will be subjected, within Task T5.1, to individual unit testing. Then, they will be integrated to form the Privacy Flag platform and will be subjected to further tests targeting the correct operation of the platform as a whole. The tests will evaluate whether the platform supports the set services efficiently and without errors.

The processes of individual unit and Integration testing will result in documented test results (accompanied by various analytics based on log files) which will be given, as a feedback to the developers (in WP4), in order to start a second unit development phase for incorporating

corrections and adaptations for the final integration and testing phase implemented by a small scale pilot operation. The final testing will be performed by CTI group members and its goal is to ensure the complete and correct functionality of the first version of the Privacy Flag platform towards the testing phase within the scope of Task T5.3.

1.3 Purpose and scope of the current document

This deliverable presents the initial testing plan for the Privacy Flag modules and their integration. This plan will be, later, refined and made more detailed as the actual module and functionalities will be available along with progress achieved for their integration. In this document, available testing methodologies are presented and tests for the Privacy Flag modules/enablers specified, to evaluate the platform and enable integration into the final platform. We then proceed to describe the use cases as well as the expected module behavior. Our test will be complemented by a small scale pilot, performed by the project partners acting as users, which will be the first actual use of the Privacy Flag platform in a real-life, but sufficiently constrained and controllable, setting to allow test operation and problem identification.

2. Methodology and Scheduling

WP5 depends on the input from other WPs, i.e. the technical enabler modules of the Privacy Flag platform in WP4. Accordingly, the initial specification of the enablers from WP4 was used as an "input" for designing the integration process within WP5.

The important milestone of WP4, which is of utmost importance for the integration and testing plans of WP5, is Milestone MS15 at M18. This milestone signifies the delivery of the first versions of the enablers developed within the context of WP4. Upon reaching the milestone, WP5 partners will start integrating and testing, as described in the sections of this deliverable that follow below.

The time table that follows shows the time sequence of the integration and testing phases (the flag represents the initialization of the corresponding task).

	2016								2017												2018			
	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4
	13	14	15	16	17	18 MS15 (WP4)	19	20	21	22	23	24 MS16 (WP4)	25	26	27	28	29	30 MS17 (WP4)	31	32	33	34	35	36
T5.1																								
T5.2	⚑																							
T5.3													⚑											
MS (WP5)						MS19						MS20						MS21			MS22			
DLV (WP5)												D5.2												D5.3 D5.4

Table 3: Time sequence of the testing phase (MS: milestone, DLV: deliverable)

Formally, the integration and testing phases (MS15) has been achieved. Before MS15, *however*, from month 13 to month 17, testing and initial integration efforts have, already, begun by involved partners using test stubs and drivers while developing their modules within the context of WP4. Initial versions of the modules will become available to WP5 for integration checks and initial interface compatibility testing, already before MS15.

Upon the reach of MS16 at month 24, the results of the tests will have been implemented and the second integration phase of the final versions of the modules will start aiming to start the Task T5.2 large scale pilot at month 25, as expected.

In this timetable, D5.1 and MS18 are not shown because D5.1 is the present deliverable and MS18 is the report to the consortium about the contents of D5.1 and the successful determination of the initial plans for the testing and integration phases.

2.1 The adopted testing methodology

Our consortium has decided to adopt an *agile* testing attitude (see [1]). Briefly, this attitude considers testing and development two intertwined phases and not sequential (i.e. testing following development). This decision was dictated by the tight delivery dates of the modules as well as the requirement for early integration before the large scale pilot of Task T5.2 takes place.

With respect to the practical, separate and integrated module testing, following the generic methodology proposed in [3], in Figure 1 we see a generic perspective of the integration, testing, and validation processes, which will be adopted by our team for the modules that will be integrated to produce the final Privacy Flag platform (the numbers refer to steps, as described below, after the figure):

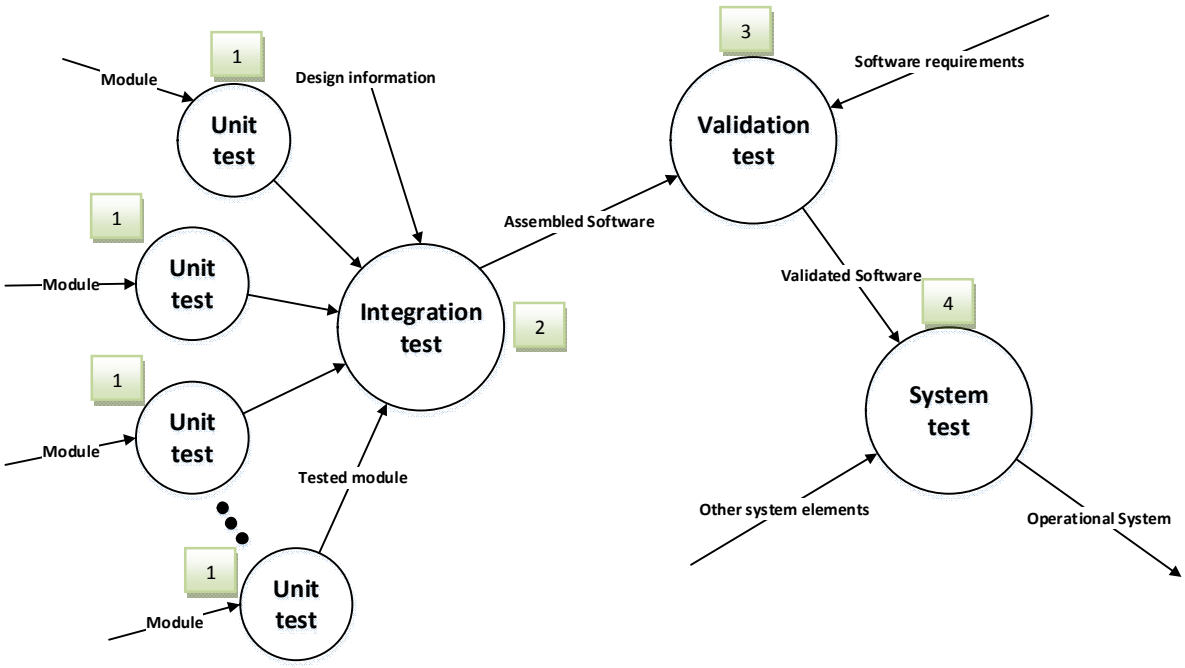


Figure 1: The general testing methodology

According to this generic methodology, our plan is composed of the following major steps:

1. Individual unit testing.
2. Integration of individual units to implement the Privacy Flag platform.
3. Validation test of the integrated platform against the requirements.
4. First round of integrated platform testing.

5. Feedback to developers and implementation of corrective measures – quick individual unit testing against reported problems.
6. Integration of new individual unit modules.
7. Second round of final platform testing.
8. Pilot operation and testing with a small group of real users.

In Figure 2 we see the testing process for each integrated module:

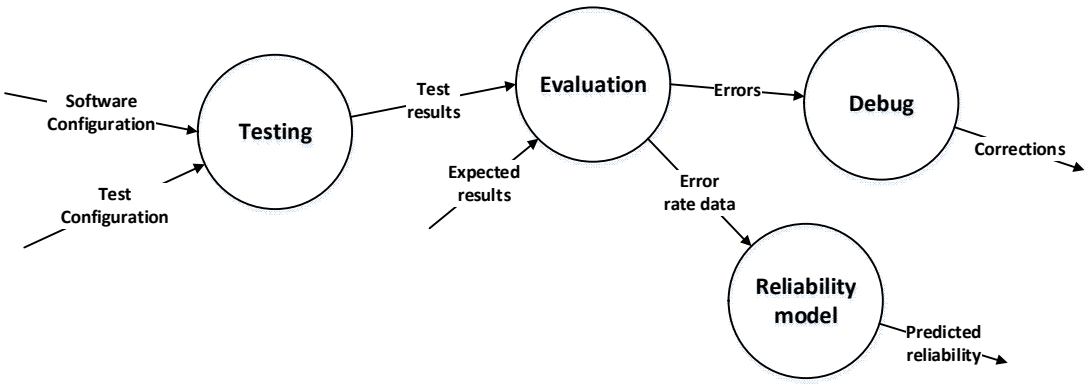


Figure 2: Testing a module

With respect to system testing, there are generally two main methodologies: (i) Incremental testing, and; (ii) Non-incremental testing. We, briefly, discuss these two methodologies in what follows and state, with justification, our adoption decision. In the context of our description, the term *low-level component* refer to self-contained modules which perform a specific, relatively simple, computation and the term *high-level component* refer to modules which perform more complex operations and require other modules for their operation.

1) Incremental system integration

- *Top-Down testing*: This approach requires the integration and testing to start from the highest-level components. This enables the testing of high-level system parts and the involved data flows and interfaces. This approach, *usually*, minimizes the need for *drivers*, i.e. test modules that invoke others, since most modules are in place (at least the high level ones). However, since lower level modules are missing, *stubs* are necessary (perhaps numerous) to feed with inputs the higher level component which are under testing, until the real lower level components become available. In addition, the lower level components are tested late in the integration phase leaving limited time for system level corrective actions. The exact form and functionality of the stubs and drivers, since they form very low code-level components of the testing phase, depend heavily on the actual code of the tested modules, which is not

available at the time of the writing of this deliverable, will be decided later by the involved partners.

- *Bottom-Up testing*: Contrary to top-down testing, this testing strategy starts from the lower level system components. This, also, minimizes the need of stubs and identifies low level problems early, before the integration begins. However, more drivers are needed to invoke, appropriately, the lower level components because the higher level components (which invoke the lower level ones) are missing.
- *Sandwich Testing*: This is also called *Hybrid Integration Testing*, and combines the two approaches discussed above. According to this approach, lower level modules are tested in parallel, while their integration also proceeds along with testing and is tested itself too. Accordingly, the need for stubs and drivers is minimized. However, this approach is a little less systematic than the top-down and bottom-up approaches and may need more coordination effort.

2) Non-Incremental system testing:

- *Big-bang testing*: In this approach all (or a large portion) of the modules that compose the system are integrated and tested. According to this testing methodology, the testing actually starts from the final system and not the individual components level. However, in this testing strategy, if an erroneous behaviour is detected while a test is performed, it is very difficult to isolate the components that fail and lead to this kind of behaviour, since attention is not paid at the component operation or component interface level during the testing.

After consideration of these options, our consortium decided to proceed with the “sandwich approach”. This methodology is more “suitable” due to the fact that partners, in parallel, develop their own modules according to the specifications set in the architecture deliverables (most notably D1.2). Thus, the module developers use the bottom-up approach to test their low-level modules (Privacy Flag platform enablers) and the CTI team, who is responsible for Task T5.1 (the platform integration task) use the bottom-up approach to test the server and the database. In the end, CTI will perform the integration test, especially at the interfaces between the different modules.

2.2 Use Case Template

The proposed template has two sections: (i) Use Case Identification, and; (ii) Use Case Definition. These are explained in the two subsections that follow and need to be defined for the test procedures followed for each of the systems described in Sections 3 to 8.

The adopted use case template is the following:

Part A: *Use case identification*

Use Case ID:			
Use Case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	

Part B: *Use case definition*

Actors:	
Description:	
Trigger:	
Preconditions:	
Postconditions:	
Normal Flow:	
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

In what follows, we explain the fields of the template.

2.2.1 Part A: Use Case Identification

2.2.1.1 Use Case ID

Give each use case a unique integer sequence number identifier, in an incremental format. For instance, we can write CTI_01 or Velti_01.

2.2.1.2 Use Case Name

State a concise, results-oriented name for the use case. These reflect the tasks the user needs to be able to accomplish using the system. Include an action verb and a noun. Examples include the following:

- User registration to the Privacy Flag platform
- Privacy Flag platform setup

2.2.1.3 Use Case History:

2.2.1.3.1 Created By

Supply the name of the person who initially documented this use case.

2.2.1.3.2 Date Created

Enter the date on which the use case was initially documented.

2.2.1.3.3 Last Updated By

Supply the name of the person who performed the most recent update to the use case description.

2.2.1.3.4 Date Last Updated

Enter the date on which the use case was most recently updated.

2.2.2 Part B: Use Case Definition

2.2.2.1 Actors

An actor is a person -or other entity- external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor that will be initiating this use case and any other actors who will participate in completing the use case.

2.2.2.2 Description

Provide a brief description of the reason for and outcome of this use case, or a high level description of the sequence of actions and the outcome of executing the use case.

2.2.2.3 Trigger

Identify the event that initiates the use case. This could be an external event or system event that causes the use case to begin, or it could be the first step in the normal flow.

2.2.2.4 Preconditions

List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each precondition. Examples:

1. The user has been, successfully, authenticated.
2. The user's mobile device has sufficient protection against malware.

2.2.2.5 Postconditions

Describe the state of the system at the conclusion of the use case execution. Number each postcondition. Examples of such preconditions are the following:

1. The user's device has detected the existence of a privacy risk at a site and has sent an alert to the server.

2. The user has submitted a UPRAAT (i.e. user assessment) questionnaire to the platform through his/her mobile device.

2.2.2.6 Normal Flow

Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and description. This description may be written as an answer to the hypothetical question, "How do I <accomplish the task stated in the use case name>?" This is best done as a numbered list of actions performed by the actor, alternating with responses provided by the system. The normal flow is numbered "X.0", where "X" is the Use Case ID.

2.2.2.7 Alternative Flows

Document other, legitimate usage scenarios that can take place within this use case separately in this section. State the alternative flow, and describe any differences in the sequence of steps that take place. Number each alternative flow in the form "X.Y", where "X" is the Use Case ID and Y is a sequence number for the alternative flow. For example, "CTI_5.3" would indicate the third alternative flow for use case number CTI_5.

2.2.2.8 Exceptions

Describe any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions. Also, describe how the system is to respond if the use case execution fails for some unanticipated reason. If the use case results in a durable state change in a database or the outside world, state whether the change is rolled back, completed correctly, partially completed with a known state, or left in an undetermined state as a result of the exception. Number each alternative flow in the form "X.Y.E.Z", where "X" is the Use Case ID, Y indicates the normal (0) or alternative (>0) flow during which this exception could take place, "E" indicates an exception, and "Z" is a

sequence number for the exceptions. For example “CTI_5.0.E.2” would indicate the second exception for the normal flow for use case SE_5.

2.2.2.9 Includes

List any other use cases that are included (“called”) by this use case. Common functionality that appears in multiple use cases can be split out into a separate use case that is included by the ones that need that common functionality.

2.2.2.10 Special Requirements

Identify any additional requirements, such as non-functional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.

2.2.2.11 Legal Considerations

Identify legal impacts which must be taken into account.

2.2.2.12 Assumptions

List any assumptions that were made in the analysis that led to accepting this use case into the product description and writing the use case description.

2.2.2.13 Notes and Issues

List any additional comments about this use case or any remaining open issues or TBDs (To Be Determined) that must be resolved. Identify who will resolve each issue, the due date, and what the resolution ultimately was.

We would like to remark that it may not be necessary, applicable or even possible to fill in all the fields mentioned above for all use cases at this point of time, as the tested modules are not available yet. However, we have provided these fields for completeness and possible later use, after the modules are ready and the testing phase begins. In a later version of this deliverable, after the tests have been performed and the results documented, we will update accordingly the use case definition tables described in the present version of the deliverable.

2.2.3 Test outcomes and corrective measures

The outcomes of the use cases will be employed towards taking corrective measures towards the tested systems, before the integration begins. After each use case description two sections follow, one that documents the results of the test and one that documents the taken corrective measures.

3.Test Use Cases

3.1 Security and privacy enablers

3.1.1 Test planning

The objective of these tests is to verify the anonymity of the people using the PrivacyFlag tool, as well as that the Quality of Service (QoS) requirements (such as latency, available bandwidth and the maximum tolerated risk of signal/data jitter, etc.) are met.

Use Case ID:	UL_PE_01		
Use Case Name:	**Privacy Flag – Privacy Enabler**		
Created By:	Daniel Forster	Last Updated By:	Daniel Forster
Date Created:	08/04/2016	Date Last Updated:	08/04/2016

Actors:	Privacy Enabler – Network Proxy, User
Description:	This test should ensure that the user's connections are routed through a privacy enhancing system that hides the user's IP address from the connection endpoint.
Trigger:	The Privacy Enabler Network Proxy gets activated.
Preconditions:	Connection endpoints can see the user's IP address.
Postconditions:	Connection endpoints cannot see the user's IP address.
Normal Flow:	The user activates the Privacy Enabler Network Proxy. His/her requests are routed through the privacy enhancing system and his/her IP address changes from the view of the connection endpoints.
Alternative Flows:	
Exceptions:	The user activates the Privacy Enabler Network Proxy, but the privacy enhancing system is not able to establish a suitable proxy server route through the network. The system should present the user an error message that the anonymous connection failed. The system should *never* fall back to a non-anonymous connection while the Privacy Enabler Network Proxy is activated and the user may think he/she is using a privacy enhanced connection.
Includes:	
Special Requirements:	
Legal Considerations:	

Assumptions:	
Notes and Issues:	

Use Case ID:	UL_PE_02		
Use Case Name:	**Privacy Flag – Privacy Enabler – Router Selection**		
Created By:	Daniel Forster	Last Updated By:	Daniel Forster
Date Created:	08/04/2016	Date Last Updated:	08/04/2016

Actors:	Privacy Enabler – Network Proxy, Tester
Description:	This test should ensure that the Network Proxy uses only anonymity proxies that are located inside of Europe.
Trigger:	The Privacy Enabler Network Proxy gets activated.
Preconditions:	
Postconditions:	
Normal Flow:	After enabling the Network Proxy, a request for the location of the set of used proxies from the privacy enhancing system should state that all proxy server locations are inside of the EU.
Alternative Flows:	
Exceptions:	The set contains a proxy server that is not located inside of Europe.
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UL_PE_03		
Use Case Name:	**Privacy Flag – Privacy Enabler – QoS requirements**		
Created By:	Daniel Forster	Last Updated By:	Daniel Forster

Date Created:	08/04/2016	Date Last Updated:	08/04/2016

Actors:	Privacy Enabler – Network Proxy, User
Description:	This test should try to ensure, that the bandwidth and latency overhead for a privacy enhanced connection is still usable.
Trigger:	The user activates the Network Proxy
Preconditions:	
Postconditions:	
Normal Flow:	The system still offers a usable user experience.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UL_PE_04		
Use Case Name:	**Privacy Flag – Privacy Enabler – No connection leaks**		
Created By:	Daniel Forster	Last Updated By:	Daniel Forster
Date Created:	08/04/2016	Date Last Updated:	08/04/2016

Actors:	Privacy Enabler – Network Proxy, Browser Add-on, Tester
Description:	This test should ensure that the browser add-on takes care of the fact that no connection from inside the browser bypasses the privacy proxy and makes a direct connection.
Trigger:	Tester activates the privacy enabler.
Preconditions:	
Postconditions:	
Normal Flow:	After activation, the tester ensures (e.g., via packet inspection)

	that the browser does not make any attempts for direct connections that bypass the privacy enhancing proxy.
Alternative Flows:	
Exceptions:	Still requests to different IP addresses than the IP address of the first proxy server in the row.
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

3.1.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the privacy enabler modules.

3.1.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the privacy enabler modules.

3.2 Crowd sourcing monitoring of privacy risks with distributed agents

The primary objective of the crowd sourcing monitoring of privacy risks with distributed agents is to provide a technical analysis of the websites visited by users or their smart phone applications based on UPRAAT questionnaire and retrieve data from the database in order to detect outliers.

3.2.1 Test planning

The goals of these tests are to observe and analyze the performance of the distributed agents. More specifically the tests planned include the verification of the correctness of answers given by the distributed agents to the UPRAAT questionnaire as well as the robustness of the scoring system. Another goal of these tests is to evaluate that the Distributed Agents properly communicate with the other PrivacyFlag tools (such as the database, UPRAAT etc.).

Based on the set requirements from Deliverable D1.2, we propose the following test cases:

Use Case ID:	CTI_DA_1		
Use Case Name:	Privacy Flag Smart Phone Application - Distributed Agent communication.		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	08/03/2016	Date Last Updated:	08/03/2016

Actors:	Distributed Agent
Description:	Test whether the Distributed Agent opens a secure communications channel with the Smart Phone application.
Trigger:	
Preconditions:	
Postconditions:	
Normal Flow:	
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	

Notes and Issues:	

Use Case ID:	CTI_DA_2		
Use Case Name:	Activation/deactivation of the automatic data collection process.		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	08/03/2016	Date Last Updated:	08/03/2016

Actors:	Distributed Agent
Description:	Test whether the Distributed Agent activates and deactivates, correctly, the automatic data collection process.
Trigger:	Activation by the user or self-activation upon detection of a threat.
Preconditions:	The distributed agent is running on the user's device.
Postconditions:	The distributed agent has sent an identified incident.
Normal Flow:	
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	CTI_DA_3		
Use Case Name:	Identification/authentication/authorization mechanisms		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	08/03/2016	Date Last Updated:	08/03/2016

Actors:	Distributed Agent
Description:	Test whether the identification, authentication and authorization mechanisms for the involved actors (PrivacyFlag Smart Phone Application and Distributed Agent) are operational.
Trigger:	The identification, authentication, and authorization functionalities are activated.
Preconditions:	The mobile phone application is connected to the Privacy Flag platform.
Postconditions:	The identification, authentication, and authorization operations have been successfully executed.
Normal Flow:	
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

3.2.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the distributed agent modules.

3.2.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the distributed agent modules.

3.3 Browser add-on

The Privacy Flag browser add-on informs users on the privacy risks when visiting a web site. The risk analysis is twofold coming both from crowd sourcing information and a set of automated tests (enablers). The goals of the tests bellow are to observe and analyze the functionality and the performance of the overall browser add-on in the terms of the response time to user requests (Quality of Service – QoS – requirement).

3.3.1 Test planning

Use Case ID:	PF_BA_01		
Use Case Name:	**Privacy flag browser add-on open response time**		
Created By:	Andreas Drakos	Last Updated By:	Andreas Drakos
Date Created:	24/03/2016	Date Last Updated:	24/03/2016

Actors:	Any user
Description:	The time for the browser add-on to load when a user opens the plugin and system responds promptly.
Trigger:	User visits a web page and opens the add-on
Preconditions:	Communication with PF server is working correctly
Postconditions:	
Normal Flow:	Once the user opens the add-on the information should be shown
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	PF_BA_02
Use Case Name:	**Privacy flag browser add-on submit response time**

Created By:	Andreas Drakos	Last Updated By:	Andreas Drakos
Date Created:	24/03/2016	Date Last Updated:	24/03/2016

Actors:	Any user
Description:	The time for the browser add-on to send the evaluation and re-load the stats
Trigger:	User submits his evaluation
Preconditions:	Communication with PF server is working correctly
Postconditions:	
Normal Flow:	After the user submits the evaluation, the main screen of the add-on should be updated with the new information
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

3.3.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the browser add-on module.

3.3.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the browser add-on module.

3.4 Smartphone application

The Privacy Flag application informs users on the privacy risks when visiting a web site. The risk analysis is twofold coming both from crowd sourcing information and a set of automated tests (enablers). The goal of the tests described in what follows is to observe and analyze performance of the overall application in the terms of the response time to user requests.

3.4.1 Test planning

Use Case ID:	PF_BA_01		
Use Case Name:	**Privacy flag application open response time**		
Created By:	Andreas Drakos	Last Updated By:	Andreas Drakos
Date Created:	24/03/2016	Date Last Updated:	24/03/2016

Actors:	Any user
Description:	The time for the application to load when a user opens it, load the needed information and system responds promptly.
Trigger:	User opens the application
Preconditions:	Communication with PF server is working correctly
Postconditions:	
Normal Flow:	Once the user opens the application the information should be shown
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	PF_BA_02
Use Case Name:	**Privacy flag browser add-on submit response time**

Created By:	Andreas Drakos	Last Updated By:	Andreas Drakos
Date Created:	24/03/2016	Date Last Updated:	24/03/2016

Actors:	Any user
Description:	The time for the application to send the evaluation and re-load the stats
Trigger:	User submits his evaluation
Preconditions:	Communication with PF server is working correctly
Postconditions:	
Normal Flow:	After the user submits the evaluation, the main screen of the application should be updated with the new information
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

3.4.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the smartphone application modules.

3.4.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the smartphone application modules.

3.5 Database and server implementation

3.5.1 Test planning

The goals of these tests are to observe and analyze the performance of the database and server infrastructure. More specifically these tests are designed to validate that all the proposed requirements (such as: scalability, responsiveness and others) are met. In what follows, we will use the following acronyms:

- ACID DB properties:
    1. *Atomicity:*
       Atomicity requires that all transactions are either completed in their entirety or fail. That is, each DB transaction should be secured against mishaps such as power failures, errors, and system crashes. To the entity submitted the transaction, the transaction is either successfully committed to or it is aborted due to some failure event.

    2. *Consistency:*
       Consistency requires that each transaction transforms the DB from a valid state into another valid state. This means that all data written to the database must be valid according to the predefined rules as well as DBMS switches and selections (which includes constraints, cascades, triggers, and their combination). This ensures that possible errors are not the result of a violation the predefined rules.

    3. *Isolation:*
       Atomicity refers to the execution of operations and transactions, with respect to data modifications, serially i.e. no operation should be initiated on a DB table before another one, which modifies it, is completed. This ensures that the concurrent execution of transactions results in a valid DB state. The means to achieve atomicity is operation isolation, which is the goal of DBMS concurrency control mechanisms.

    4. *Durability:*
       Durability requires that after a transaction is committed to, the data pertaining to the transaction remain permanent, even in cases of power loss or other system malfunction. With respect to this issue, one must be careful about the system and DBMS cash memory policy. One must ensure that the commitment of a transaction also brings data to a non-volatile memory module or disk (e.g. by imposing a write-through cash memory policy).

- CRUD DB operations:
  C: Create – When a user or other system module initiates a ‘Save’ to perform a new data send transaction, a ‘Create’ DB operation is executed.

  R: Retrieve – When a user or other system module initiates a ‘Search’ or a ‘View’ to read any saved transaction data, the ‘Retrieve’ DB operation is executed.

U: Update – When a user or other system module initiates an 'Edit' or 'Modify' to change stored data, the 'Update' DB operation is executed.

D: Delete – When a user or other system module initiates a 'Remove' for any data item in the system, the 'Delete' DB operation of DB is executed.

More specifically, our team will target the following goals during the testing phase:

1. Confirmation of correct data mapping: our team will ascertain that the data correspondence/mapping between the different information formats, module interfaces and user interfaces with respect to the tables of the DB is, both, accurate and conforms to design documentation. For all CRUD operations, we will verify that respective tables, records and fields are updated when user initiates a 'Save', 'Update', 'Search' or 'Delete' transaction from a GUI or when a modules performs the same operations remotely (e.g. the distributed user agents).

2. Ensure that the ACID DB properties are observed for all types of transactions: As we stated above, the ACID properties of DB Transactions refer to the 'Atomicity', 'Consistency', 'Isolation' and 'Durability'. Testing these properties is a major element of any DB testing plan and our team will perform it for all transactions defined in the Privacy Flag platform.

3. Ensure Data Integrity: we will examine whether the same data appear at users (e.g. on their GUI screens) as well as platform modules. That is, we will examine whether the latest status of data stored in the DB is reflected throughout the platform entities so that all entities access the most recent data on all APIs as well as user GUI screens.

4. Ensure Accuracy of implemented Business Process: we will examine if the implemented DB features such as 'Referential Integrity', relational constrains, triggers and stored procedures reflect the processes and intended data transformations and dependencies described in the architecture document.

5. Ensure that access control and data protection mechanisms are implemented and are operational during the operation of the DB.

The following lists the main tests that will be performed:

1. Creation and execution of test queries: our team will create SQL queries with dummy, but realistic, data as they would come from the agents communicating with the DB. These queries will store and retrieve records carrying dummy threat and alert data from fictitious websites and applications.

2. Observe data in tables (i.e. records and fields): After the successful commitment of the data manipulated by the queries, our team will examine (directly) the stored data and compare them with expected results in order to identify queries or data which were not properly handled.

3. Obtain test queries from enabler developers: our team will also ask the developers of the enablers (WP4) to produce their own queries for testing the DB, as they would be sent by the enablers themselves.

4. Test the DB responsiveness under heavy workload: our team will create batches of sample SQL queries, simulating heavy DB workload stemming from requests coming from a large crowd. This will identify the threshold point at which the responsiveness of the DB will become intolerable.

5. Ascertain that all communication with the database is secure and appropriate access rights mechanisms are implemented.

With respect to the testing of the server which will host the database, our team will test the server for Quality of Service and, especially, its ability to withstand as many connections as possible within a given time frame. In addition, tests will be performed to see whether the SSL connections are properly initiated and sustained throughout the whole session duration.

According to the above considerations, we have developed the following test cases:

1) Server use cases:

Use Case ID:	CTI_SE_1		
Use Case Name:	Performance		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	07/03/2016	Date Last Updated:	07/03/2016

Actors:	Distributed agents and users.
Description:	This is a stress test for the server according to which the testing team will find the threshold point at which the response time of the server drops significantly. This will test only the ability of the server to sustain an acceptable connection rate without taking into account the database response times (this will be a separate test for the database module).
Trigger:	Connections from the involved actors. Also, if necessary, there will be a use of an automated Webserver Stress Tool (we have identified Paessler's free webserver testing tool at *https://www.paessler.com/tools/webstress*).
Preconditions:	Requests for connections.
Postconditions:	Each request is served within a predetermined time interval (this will be defined and set during the integration phase as it will depend on the employed technologies and the final module's configurations).

Normal Flow:	The connections are served within he predetermined time limit.
Alternative Flows:	
Exceptions:	The service time falls outside the preset time limit.
Includes:	
Special Requirements:	Establishment of a fast connection.
Legal Considerations:	
Assumptions:	The connection between the communicating applications and devices is sufficiently fast.
Notes and Issues:	

Use Case ID:	CTI_SE_2		
Use Case Name:	Data confidentiality		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	07/03/2016	Date Last Updated:	07/03/2016

Actors:	Distributed agents and users.
Description:	This will test whether all data connections between the actors and the database are suitably encrypted, i.e. whether the SSL protocol is activated with the correct connection parameters (e.g. encryption algorithm used and key sizes).
Trigger:	Connections from the involved actors.
Preconditions:	Request for a connection.
Postconditions:	SSL connection is activated with the correct parameters.
Normal Flow:	The exchanged data are in encrypted format based on the encryption algorithm activated by the SSL protocol.
Alternative Flows:	
Exceptions:	Data is not properly encrypted.
Includes:	
Special Requirements:	Firewall connections (if firewalls are activated) do not block the connections requests.
Legal Considerations:	

Assumptions:	The SSL protocol suite is correctly set-up and executes correctly on the involved communicating agents.
Notes and Issues:	

2) Database use cases:

Use Case ID:	CTI_DB_1		
Use Case Name:	Execution of sample queries		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	07/03/2016	Date Last Updated:	07/03/2016

Actors:	Distributed agents and users (through questionnaires).
Description:	This test will evaluate the ability of the database to correctly execute sample queries on sample data.
Trigger:	A connection from an agent or users.
Preconditions:	The query arrives, intact, to the database
Postconditions:	The results of the query match the expected results, as reflected by the database scheme and sample contents.
Normal Flow:	The results are promptly returned and are as expected based on the stored values.
Alternative Flows:	
Exceptions:	Query results are faulty or query results are not returned at all.
Includes:	
Special Requirements:	The database is up and running.
Legal Considerations:	
Assumptions:	The database server is correctly set-up and configured while the database contents are correct.
Notes and Issues:	

Use Case ID:	CTI_DB_2		
Use Case Name:	Data confidentiality		
Created By:	Yannis Stamatiou	Last Updated By:	Yannis Stamatiou
Date Created:	07/03/2016	Date Last Updated:	07/03/2016

Actors:	Data exchanged with other platform modules.
Description:	Test whether the connection with the DB is secure, i.e. data encryption and authentication mechanisms are implemented and enabled.
Trigger:	Initiation of communication between the DB and another module (e.g. Distributed Agents).
Preconditions:	The database and platform modules are correctly configured for communication.
Postconditions:	Data is exchanged between the database and any connecting module in encrypted format.
Normal Flow:	Data is properly encrypted.
Alternative Flows:	
Exceptions:	Data is not in encrypted format.
Includes:	
Special Requirements:	The involved modules and the database are correctly set-up and configured.
Legal Considerations:	
Assumptions:	All modules are in an appropriate operating condition.
Notes and Issues:	

3.5.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the database and server application modules.

3.5.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the database and server application modules.

3.6 Website and backend management platform

The deployment architecture for the website and backend management platform consists of Privacy Flag website deployed in DotNetNuke[1]. The main features of the DotNetNuke can be expanded by adding third-party extensions, i.e. modules from an existing library repository, or by developing custom functionality. The DotNetNuke framework provides basic functionality such as security, user administration, and content management, while modules are used to tailor the web site for specific deployment needs. Development of module can be done in either VB.NET or C#.

3.6.1 Test planning

[Performance tests] The goals of these tests is to observe and analyze performance of the overall web platform in the terms of the response time to user requests, session setup time between a resource user and a platform; and to evaluate the backend platform processes.

Use Case ID:	DNET_01		
Use Case Name:	**Access to Privacy Flag webpage response time**		
Created By:	Mirjana Nikolic	Last Updated By:	Mirjana Nikolic
Date Created:	24/01/2016	Date Last Updated:	24/01/2016

Actors:	Authenticated user
Description:	Authenticated user starts an action on the privacy flag webpage and system responds promptly.
Trigger:	An action is performed by the user.
Preconditions:	User has been authenticated
Postconditions:	
Normal Flow:	Authenticated user starts an action on the privacy flag webpage and system responds as expected. System response time is within the limits.
Alternative Flows:	
Exceptions:	The response time falls outside the set limits.
Includes:	
Special Requirements:	The web server is appropriate configured and set-up.
Legal Considerations:	

[1] *.NetNuke* is a Web CMS Platform that provides rich commercial Websites. More information about the CMS can be found under the following link: *http://www.dnnsoftware.com/*

Assumptions:	The user has been authenticated and the connection between the user's device and the server is sufficiently fast.
Notes and Issues:	

Use Case ID:	DNET_02		
Use Case Name:	**Access to Privacy Flag webpage concurrent users**		
Created By:	Mirjana Nikolic	Last Updated By:	Mirjana Nikolic
Date Created:	24/01/2016	Date Last Updated:	24/01/2016

Actors:	Authenticated user
Description:	The number of authenticated users is performing actions on the privacy flag webpage and system responds promptly.
Trigger:	Actions are performed by the number of users.
Preconditions:	User has been authenticated
Postconditions:	
Normal Flow:	The number of authenticated users starts an action on the privacy flag webpage and system responds as expected. System response time is within the limits for concurrent users.
Alternative Flows:	
Exceptions:	System response time falls outside the preset limits.
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_03		
Use Case Name:	**Privacy Flag webpage session setup time**		
Created By:	Mirjana Nikolic	Last Updated By:	Mirjana Nikolic

Date Created:	24/01/2016	Date Last Updated:	24/01/2016

Actors:	Registered user
Description:	The user starts a session and system responds promptly.
Trigger:	The user opens a web page.
Preconditions:	The user is registered.
Postconditions:	
Normal Flow:	The user starts a session and system responds as expected. Session setup time is within the limits.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_04		
Use Case Name:	**Privacy Flag webpage session concurrent users**		
Created By:	Mirjana Nikolic	Last Updated By:	Mirjana Nikolic
Date Created:	24/01/2016	Date Last Updated:	24/01/2016

Actors:	Registered user
Description:	The number of concurrent user starts a sessions and system responds promptly.
Trigger:	The number of users opens a web page.
Preconditions:	The users are registered and logged in the website.
Postconditions:	
Normal Flow:	The number of concurrent user starts a session and system responds as expected. Session setup time is within the limits.

Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_05		
Use Case Name:	**Privacy Flag backend users authorization**		
Created By:	Nenad Gligoric	Last Updated By:	Nenad Gligoric
Date Created:	2/05/2016	Date Last Updated:	2/05/2016

Actors:	Registered user
Description:	A user should be able to access only resources which he is authorized to access after starting the session.
Trigger:	The user opens backend of the website.
Preconditions:	The user is registered into the platform.
Postconditions:	
Normal Flow:	The number of concurrent users starts a session and system responds as expected and allows access only to authorized resources.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_06		
Use Case Name:	**Privacy Flag webpage session concurrent users**		
Created By:	Nenad Gligoric	Last Updated By:	Nenad Gligoric
Date Created:	02/05/2016	Date Last Updated:	02/05/2016

Actors:	Any user
Description:	The number of concurrent user starts a sessions and system responds promptly.
Trigger:	The number of users opens a web page.
Preconditions:	The users are registered and logged in the website.
Postconditions:	
Normal Flow:	The number of concurrent user starts a session and system responds as expected. Session setup time is within the limits.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_08		
Use Case Name:	**Add-on and smartphone application download**		
Created By:	Nenad Gligoric	Last Updated By:	Nenad Gligoric
Date Created:	02/05/2016	Date Last Updated:	02/05/2016

Actors:	Any user
Description:	A user should be able to download Privacy Flag Add-On and Smartphone application from the website.

Trigger:	The user opens the website.
Preconditions:	The user is registered into the platform.
Postconditions:	
Normal Flow:	The user opens the website and installs the browser Add-On or Smartphone application without any errors.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Legal Considerations:	
Assumptions:	
Notes and Issues:	

Use Case ID:	DNET_09		
Use Case Name:	**Privacy flag user's assessment of the privacy risk**		
Created By:	Nenad Gligoric	Last Updated By:	Nenad Gligoric
Date Created:	02/05/2016	Date Last Updated:	02/05/2016

Actors:	Authenticated and authorized user
Description:	A user should be able to assess the privacy risk using the form and questionnaire populated on the website backend.
Trigger:	The user opens assessment form in the website backend.
Preconditions:	The user is registered into the platform.
Postconditions:	The form is successfully saved into the database
Normal Flow:	The number of concurrent users starts a session and submits the assessment of the privacy risk.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	

Legal Considerations:	
Assumptions:	
Notes and Issues:	

3.6.2 Outcomes

In the next versions of this deliverable, this section will document the outcomes of the tests on the backend management platform modules.

3.6.3 Corrective measures

In the next versions of this deliverable, this section will document the correcting actions (if any were required) on the backend management platform modules.

4.Small Scale Pilot

CTI, which will also supervise the integration process within the context of Task T5.1, will perform a small scale pilot within the area of the University of Patras Campus, in Patras, Greece. In the context of this pilot, CTI team members will roam the campus collecting data from contacted web sites and installed applications.

This small scale pilot will be scheduled after the individual and integration use cases have been performed and the corrective measures have been taken in order to handle the issues that arose. Since the main part of the testing and integration phase will be from M18 to M24, the small scale pilot is estimated to take place during the M24, in preparation of the full scale pilot, with real users, that will start at the M25. In this pilot, members of the IoT Lab of CTI will participate as users, offering also their technical expertise on operational and functionality related aspects of the tested platform.

This small pilot will precede the large scale pilot in the context of the Task T5.2 that will evaluate the platform in real environment with the end-users. A goal is to eliminate any remaining small issues related to the correct operation of the platform as well as its QoS guarantees in preparation for the T5.2 pilot operation. The small pilot users will simulate real users visiting sites and downloading application performing, essentially, the use cases described in this document. This in-field operation of the platform will uncover possibly remaining issues, after the testing and integration phases, leading the involved partners to perform final corrections and modifications in the technical enablers. Then a final integration step will follow to produce the first fully operational version of the platform for the full scale pilot that will begin at M25.

5.Conclusion

In this deliverable we have described the initial integration and testing plan for the individual modules as well as the integrated Privacy Flag platform. CTI will coordinate, within the context of the Task T5.1, the integration and testing of the Privacy Flag modules as described and delivered by WP4, with the assistance of the partners who develop the corresponding modules. Since the modules that will be integrated were not fully developed upon the delivery of this document, we have described initial plans and tests which will be refined later on.

Each partner will perform the tests described in this deliverable and take the corresponding corrective measures for any issues that will arise. Refinements of the tests as well as their results will be continuously added in this document and documented in final form in Deliverable D5.2.

A small scale pilot will be organized to validate the performance of the Privacy Flag platform, followed by the larger scale evaluation and integration that will be performed in the pending tasks.

6.List of References

[1] L. Crispin and J. Gregory. *Agile Testing: A Practical Guide for Testers and Agile Teams*. Addison-Wesley Professional, 2009.

[2] G.J. Myers and C. Sandler. *The Art of Software Testing*. Wiley, 3rd Edition, 2011.

[3] R. Pressman and B. Maxim. *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 8th Edition, 2014.

[4] A. Raza and S. Clyde. *Creating Datasets for Testing Relational Databases: Test-data Exaction and Comparison with Test-data Generation*. LAP LAMBERT Academic Publishing, 2012.

[5] "Privacy Flag" Project, official deliverable D1.2, "Privacy Flag initial architectural design", Grant Agreement No. 653426, 2015.

[6] "Privacy Flag" Project, official deliverable D4.1, "First year report on Technical Enablers development", Grant Agreement No.653426, 2015.

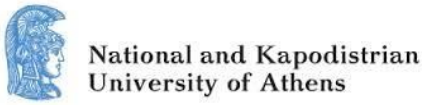
Partners


Archimede Solutions SARL


CTI
"DIOPHANTUS"
computer technology institute & press


DUNAV NET


velti


National and Kapodistrian University of Athens


uni.lu
UNIVERSITÉ DU LUXEMBOURG


OTE


MANDAT INTERNATIONAL


University of BRISTOL


HWCommunications
Cyber Security and Resilience


LULEÅ
UNIVERSITY
OF TECHNOLOGY


Istituto Italiano per la Privacy
e la Valorizzazione dei Dati